Modern Data Analytics

NetApp Solutions

NetApp
August 01, 2022

Table of Contents

NetApp Modern Data Analytics Solutions

NetApp StorageGRID with Splunk SmartStore

TR-4869: NetApp StorageGRID with Splunk SmartStore

Karthikeyan Nagalingam, Bobby Oommen, Joseph Kandatilparambil

Splunk Enterprise is the market-leading Security Information and Event Management (SIEM) solution that drives outcomes across the Security, IT, and DevOps teams. Data volumes continue to grow at exponential rates, creating massive opportunities for enterprises that can leverage this vast resource. Splunk Enterprise continues to gain adoption across a wider variety of use cases. As the use cases grow, so does the amount of data that Splunk Enterprise ingests and processes. The traditional architecture of Splunk Enterprise is a distributed scale-out design providing excellent data access and availability. However, enterprises using this architecture are faced with growing costs associated with scaling to meet the rapidly growing volume of data.

Splunk SmartStore with NetApp StorageGRID solves this challenge by delivering a new deployment model in which compute and storage is decoupled. This solution also unlocks unmatched scale and elasticity for Splunk Enterprise environments by allowing customers to scale across single and multiple sites, all while reducing costs by allowing compute and storage to scale independently and adding intelligent tiering to cost-effective cloud- based S3 object storage.

The solution optimizes the amount of data in local storage while maintaining search performance, allowing compute and storage to be scaled on demand. SmartStore automatically evaluates data access patterns to determine which data needs to be accessible for real-time analytics and which data should reside in lower-cost S3 object storage.

This technical report outlines the benefit NetApp provides to a Splunk SmartStore solution while demonstrating a framework for designing and sizing Splunk SmartStore in your environment. The result is a simple, scalable, and resilient solution that delivers a compelling TCO. StorageGRID provides the scalable and cost-effective S3 protocol/API-based object storage, also known as remote storage, allowing organizations to scale their Splunk solution at a lower cost while increasing resiliency.

ⓘ | Splunk SmartStore refers to object storage as remote stores or remote storage tiers.

About NetApp StorageGRID

NetApp StorageGRID is a software-defined object storage solution for large archives, media repositories, and web data stores. With StorageGRID, NetApp leverages two decades of experience in delivering industry-leading innovation and data management solutions while helping organizations manage and maximize the value of their information both on-premises and in public, private, or hybrid cloud deployments.

StorageGRID provides secure, durable storage for unstructured data at scale. Integrated, metadata-driven lifecycle management policies optimize where your data lives throughout its life. Content is placed in the right location, at the right time, and on the right storage tier to reduce cost. The single namespace allows the data to be accessed via a single call regardless of geographical location of the StorageGRID storage. Customers can deploy and manage multiple StorageGRID instances between datacenters and in the cloud infrastructure.

A StorageGRID system is composed of globally distributed, redundant, heterogeneous nodes that can be integrated with both existing and next-generation client applications.
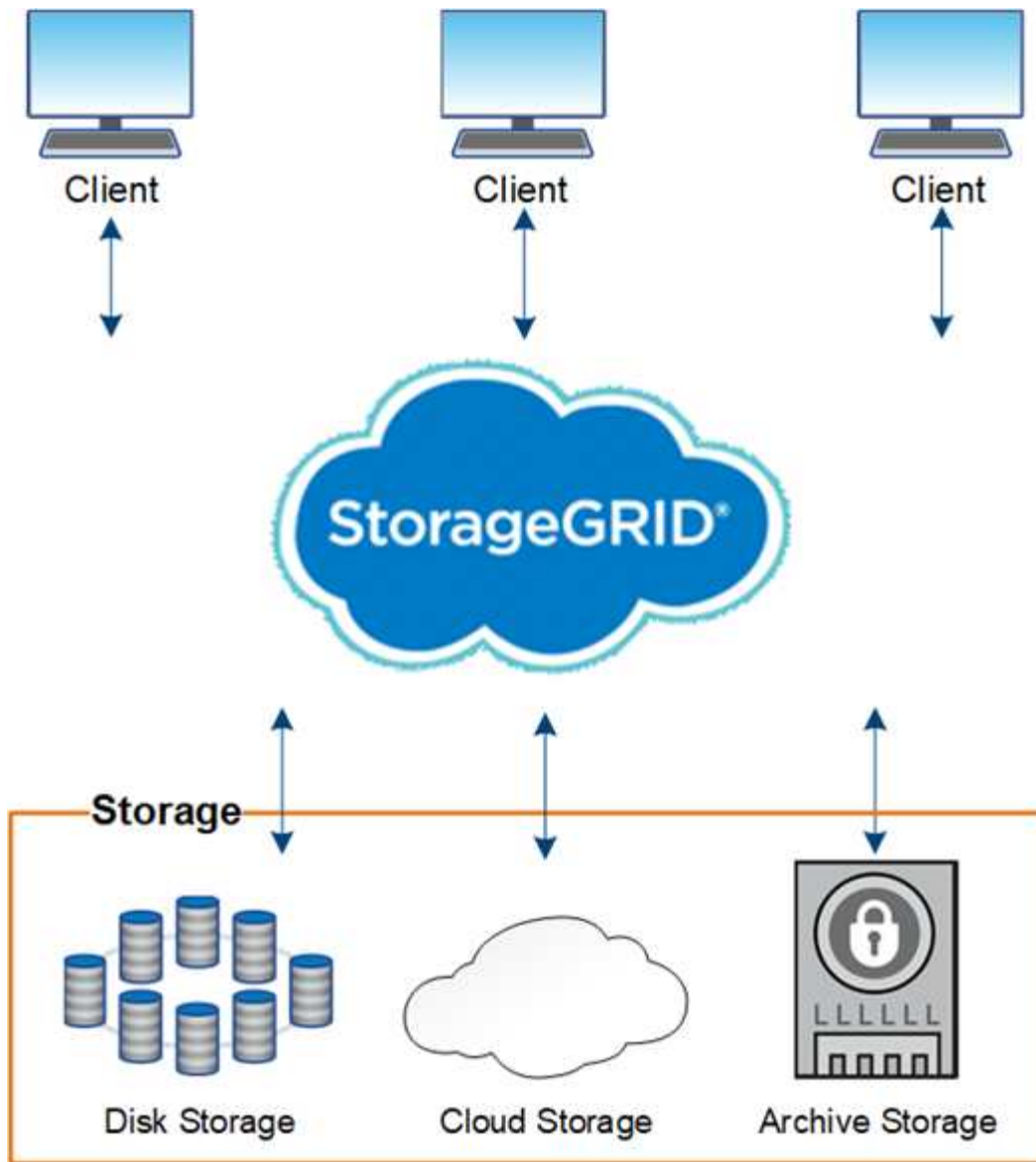
IDC MarketScape recently named NetApp as a leader in the latest report, IDC MarketScape: Worldwide Object-Based Storage 2019 Vendor Assessment. With nearly 20 years of production deployments in the most demanding industries, StorageGRID is a recognized leader in unstructured data.

With StorageGRID, you can achieve the following:

- Deploy multiple StorageGRID instances to access data from any location between data centers and the cloud through a single namespace that easily scales to hundreds of petabytes.
- Provide flexibility to deploy and centrally manage across infrastructures.
- Provide unmatched durability with fifteen-nines of durability leveraging layered Erasure Coding (EC).
- Enable more hybrid multi-cloud capabilities with validated integrations into Amazon S3 Glacier and Azure Blob.
- Meet regulatory obligations and facilitate compliance through tamper-proof data retention, without proprietary APIs or vendor lock-in.

For more information about how StorageGRID can help you solve your most complex unstructured data management problems, see the NetApp StorageGRID homepage.

About Splunk Enterprise

Splunk Enterprise is a platform for turning data into doing. Data generated by various sources such as log files, websites, devices, sensors, and applications are sent to and parsed by the Splunk Indexers, allowing you to derive rich insights from the data. It might identify data breaches, point out customer and product trends, find opportunities to optimize infrastructure, or create actionable insights across a wide variety of use cases.

About Splunk SmartStore

Splunk SmartStore expands on the benefits of the Splunk architecture while simplifying its ability to scale cost-effectively. The decoupling of compute and storage resources results in indexer nodes optimized for I/O with significantly reduced storage needs because they only store a subset of data as cache. You do not have to add extra compute or storage when only one of those resources is necessary, which allows you to realize significant cost savings. You can use cost-effective and easily scalable S3-based object storage, which further simplifies the environment, reduces costs, and allows you to maintain a more massive data set.

Splunk SmartStore delivers significant value to organizations, including the following:

- Lowering storage cost by moving warm data to cost-optimized S3 object storage
- Scaling seamlessly by decoupling storage and compute
- Simplifying business continuity by leveraging resilient cloud-native storage

Benefits of this solution

- **Performance.** The combination of Splunk SmartStore and NetApp StorageGRID provides fast migration of data between hot buckets and warm buckets using object storage. StorageGRID turbocharges the migration process by providing fast performance for large object workloads.
- **Multisite ready.** The StorageGRID distributed architecture allows Splunk SmartStore to extend deployments across single and multiple sites through a single global namespace where data can be accessed from any site regardless of where the data lives.
- **Improved scalability.** Scale storage resources independently from compute resources to meet evolving needs and demands in your Splunk environment, thereby providing improved TCO.
- **Capacity.** Meet rapidly growing volumes in Splunk deployment with StorageGRID by scaling a single namespace to over 560PB.
- **Data availability.** Optimize data availability, performance, geo-distribution, retention, protection, and storage costs with metadata-driven policies that can adjust dynamically as the business value of your data evolves.

Increase performance with the SmartStore cache, which is a component of the indexer that handles the transfer of bucket copies between local (hot) and remote (warm) storage. Splunk sizing for this solution is based on the guidelines provided by Splunk. The solution allows adding compute, hot storage, or S3 resources to meet the growing demand in terms of the number of users or ingest rate across single and multisite deployments.

Intelligent tiering and cost savings

As customers realize the power and ease of using Splunk data analytics, they naturally want to index an ever-growing amount of data. As the amount of data grows, so does the compute and storage infrastructure required to service it. Since older data is referenced less frequently, committing the same amount of compute resources and consuming expensive primary storage becomes increasingly inefficient. To operate at scale, customers benefit from moving warm data to a more cost-effective tier, freeing compute and primary storage for hot data.

Splunk SmartStore with StorageGRID offers organizations a scalable, performant, and cost-effective solution. Because SmartStore is data-aware, it automatically evaluates data-access patterns to determine which data needs to be accessible for real-time analytics (hot data) and which data should reside in lower-cost long-term storage (warm data). SmartStore uses the industry-standard AWS S3 API dynamically and intelligently, placing data in S3 storage provided by StorageGRID. The flexible scale-out architecture of StorageGRID allows the warm data tier to grow cost effectively as needed. The node-based architecture of StorageGRID makes sure that performance and cost requirements are met optimally.

The following image illustrates the Splunk and StorageGRID tiering.
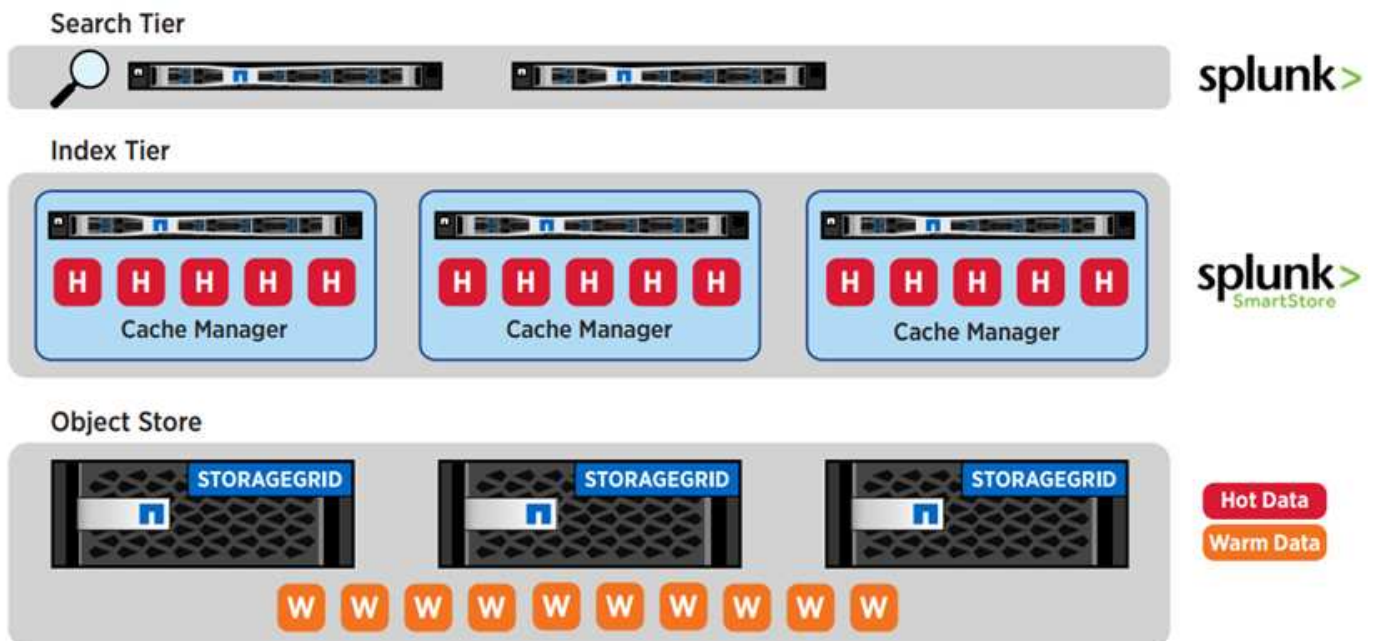
The industry-leading combination of Splunk SmartStore with NetApp StorageGRID delivers the benefits of decoupled architecture through a full-stack solution.

Solution overview

NetApp StorageGRID

NetApp StorageGRID is a high-performance and cost-effective object storage platform. It offers intelligent, policy-driven global data management using a distributed, node-based grid architecture. It simplifies the

management of petabytes of unstructured data and billions of objects through its ubiquitous global object namespace combined with sophisticated data management features. Single-call object access extends across sites and simplifies high availability architectures while ensuring continual object access regardless of site or infrastructure outages.

Multitenancy allows multiple cloud and enterprise unstructured data applications to be securely serviced within the same grid, increasing the ROI and use cases for StorageGRID. Multiple service levels can be created with metadata-driven object lifecycle policies, optimizing durability, protection, performance, and locality across multiple geographies. Users can adjust policies and realign the data landscape non-disruptively as their requirements change.

SmartStore leverages StorageGRID as the remote storage tier and allows customers to deploy multiple geographically distributed sites for robust availability and durability, presented as a single object namespace. This allows Splunk SmartStore to take advantage of the StorageGRID high performance, dense capacity, and ability to scale to hundreds of nodes across multiple physical sites using a single URL to interact with the objects. This single URL also allows storage expansion, upgrades, and repairs to be nondisruptive, even beyond a single site. The StorageGRID unique data management policy engine provides optimized levels of performance and durability and adherence to data locality requirements.

Splunk Enterprise

Splunk, a leader in the collection and analysis of machine-generated data, helps simplify and modernize IT thru its operational analytics capabilities. It also expands into business analytics, security, and IoT use cases. Storage is a critical enabler for a successful Splunk software deployment.

Machine-generated-data is the fastest-growing type of big data. The format is unpredictable and comes from many different sources, often at high rates and in great volumes. These workload characteristics are often referred to as digital exhaust. Splunk SmartStore helps to make sense of this data and provides smart data tiering for optimized placement of hot and warm data on the most cost-effective storage tier.

Splunk SmartStore

Splunk SmartStore is an indexer capability that uses object storage (also referred to as remote storage or remote storage tiers) such as StorageGRID to store warm data using the S3 protocol.

As a deployment's data volume increases, demand for storage typically outpaces demand for computer resources. SmartStore allows you to manage your indexer storage and compute resources cost-effectively by scaling compute and storage separately.

SmartStore introduces a remote storage tier, using the S3 protocol, and a cache manager. These features allow data to reside either locally on indexers or remote storage. The cache manager, which resides on the indexer, manages data movement between the indexer and the remote storage tier. Data is stored in buckets (hot and warm) along with bucket metadata.

With SmartStore, you can reduce the indexer storage footprint to a minimum and choose I/O-optimized compute resources because most data resides on the remote storage tier. The indexer maintains a local cache, representing the minimal amount of data necessary to return the results requested and predicted. The local cache contains hot buckets, copies of warm buckets participating in active or recent searches, and bucket metadata.

Splunk SmartStore with StorageGRID enables customers to incrementally scale the environment with high-performance and cost-effective remote storage while providing a high degree of elasticity to the overall solution. This allows customers to add any components (hot storage and/or warm S3 storage) in any given quantity at any given time, whether they need more indexers, change data retention, or to increase the ingest rate without any disruption.

Flexible StorageGRID features for Splunk SmartStore

StorageGRID has a wide variety of features that users can leverage and customize for their ever-changing environment. From deploying to scaling your Splunk SmartStore, your environment demands rapid adoption to changes and should be nondisruptive to Splunk. The StorageGRID flexible data management policies (ILM) and traffic classifiers (QoS) let you plan and adapt to your environment.

Simple management with Grid Manager

Grid Manager is the browser-based graphical interface that allows you to configure, manage, and monitor your StorageGRID system across globally distributed locations in a single pane of glass, as shown in the following image.
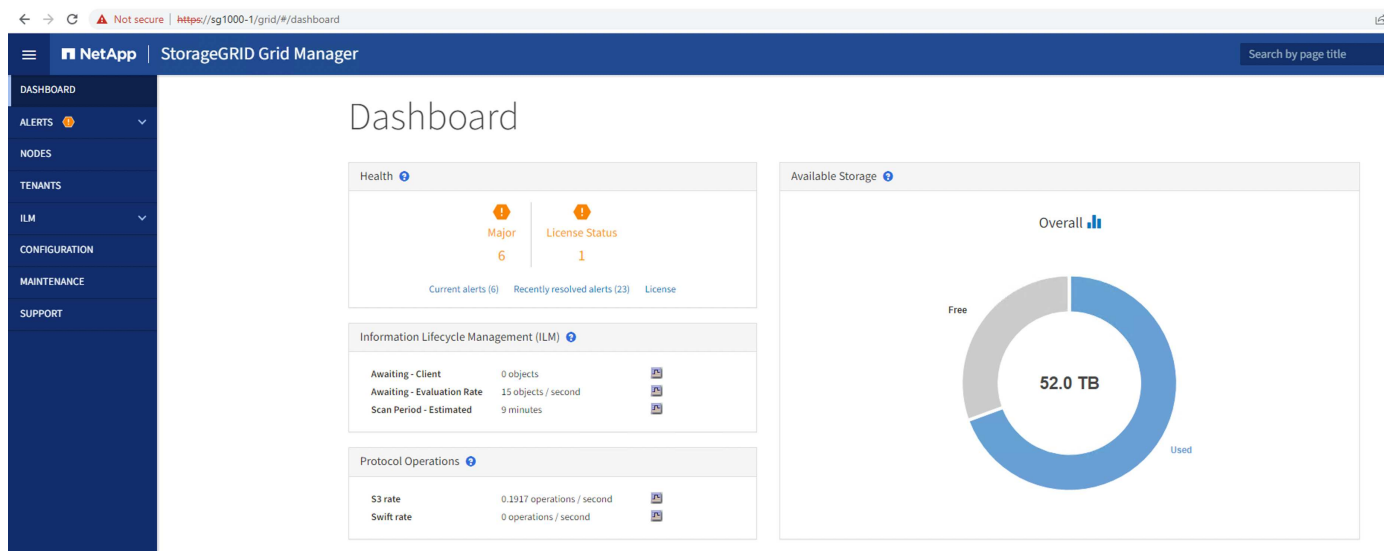
Perform the following tasks with the Grid Manager interface:

- Manage globally distributed, petabyte-scale repositories of objects such as images, video, and records.
- Monitor grid nodes and services to ensure object availability.
- Manage the placement of object data over time using information lifecycle management (ILM) rules. These rules govern what happens to an object's data after it is ingested, how it is protected from loss, where object data is stored, and for how long.
- Monitor transactions, performance, and operations within the system.

NetApp StorageGRID App for Splunk

The NetApp StorageGRID App for Splunk is an application specific for Splunk Enterprise. This app works in conjunction with the NetApp StorageGRID Add-on for Splunk. It provides visibility into StorageGRID health, account usage information, security audit details, resource usage and monitoring, and so on.

The following image shows the StorageGRID App for Splunk.
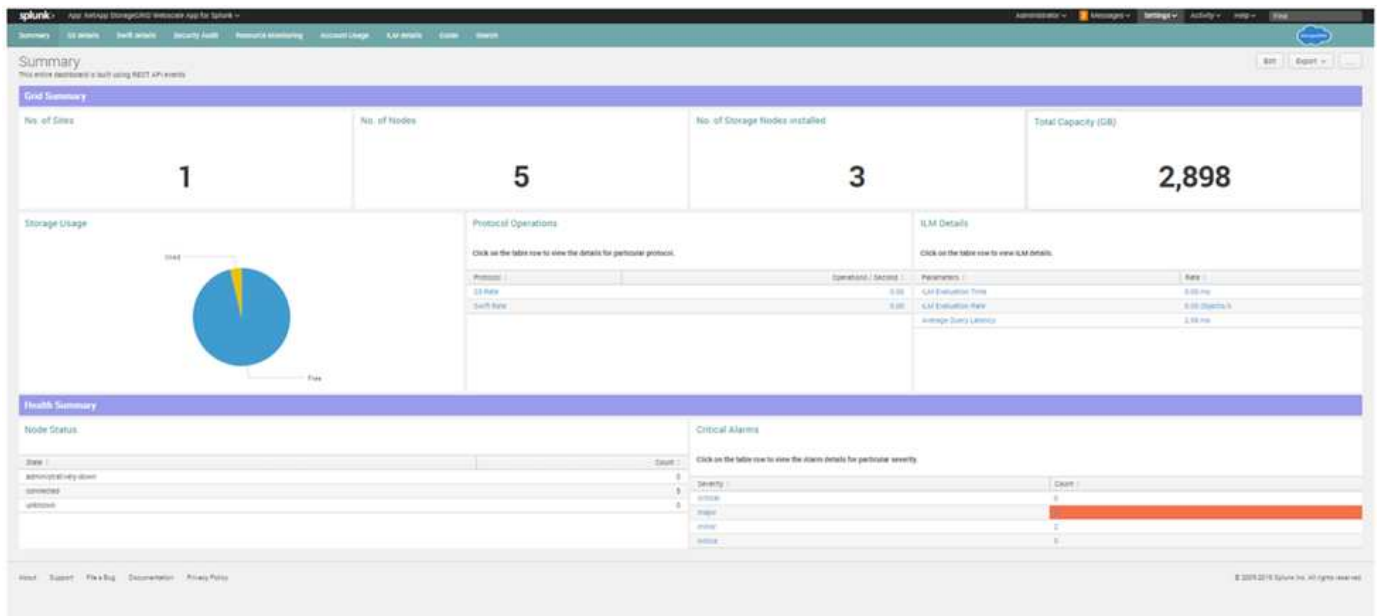
ILM policies

StorageGRID has flexible data management policies that include keeping multiple copies of your objects and using EC (erasure coding) schemes like 2+1 and 4+2 (and many others) to store your objects depending on specific performance and data protection requirements. As workloads and requirements change over time, it's common that ILM policies must change over time as well. Modifying ILM policies is a core feature, allowing StorageGRID customers to adapt to their ever-changing environment quickly and easily.

Performance

StorageGRID scales performance by adding more nodes, which can be VMs or bare metal or purpose-built appliances like the SG5712, SG5760, SG6060, or SGF6024. In our tests, we exceeded the SmartStore key performance requirements with a minimum-sized three-node grid using the SG6060 appliance. As customers scale their Splunk infrastructure with additional indexers, they can add more storage nodes to increase performance and capacity.

Load Balancer and endpoint configuration

Admin nodes in StorageGRID provide the Grid Manager UI (user interface) and REST API endpoint to view, configure, and manage your StorageGRID system, as well as audit logs to track system activity. To provide a highly available S3 endpoint for Splunk SmartStore remote storage, we implemented the StorageGRID load balancer, which runs as a service on admin nodes and gateway nodes. In addition, the load balancer also manages local traffic and talks to the GSLB (Global Server Load Balancing) to help with disaster recovery.

To further enhance endpoint configuration, StorageGRID provides traffic classification policies built into the admin node, lets you monitor your workload traffic, and apply various quality-of-service (QoS) limits to your workloads. Traffic classification policies are applied to endpoints on the StorageGRID Load Balancer service for gateway nodes and admin nodes. These policies can assist with traffic limiting and monitoring.

Splunk architecture

Key definitions

The next two tables list the Splunk and NetApp components used in the distributed Splunk deployment.

This table lists the Splunk hardware components for the distributed Splunk Enterprise configuration.

Splunk component	Task
Indexer	Repository for Splunk Enterprise data
Universal forwarder	Responsible for ingesting data and forwarding data to the indexers
Search head	The user front end used to search data in indexers
Cluster master	Manages the Splunk installation of indexers and search heads
Monitoring Console	Centralized monitoring tool used across the entire deployment
License master	License master handles Splunk Enterprise licensing
Deployment server	Updates configurations and distributes apps to processing component
Storage component	Task
NetApp AFF	All-flash storage used to manage hot tier data. Also known as local storage.
NetApp StorageGRID	S3 object storage used to manage warm tier data. Used by SmartStore to move data between the hot and warm tier. Also known as remote storage.

This table lists the components in the Splunk storage architecture.

Splunk component	Task	Responsible component
SmartStore	Provides indexers with the ability to tier data from local storage to object storage.	Splunk
Hot	The landing spot where universal forwarders place newly written data. Storage is writable, and data is searchable. This data tier is typically composed of SSDs or fast HDDs.	ONTAP
Cache Manager	Manages local cache of indexed data, fetches warm data from remote storage when a search occurs, and evicts least frequently used data from the cache.	SmartStore

Splunk component	Task	Responsible component
Warm	Data is rolled logically to the bucket, renamed to the warm tier first from the hot tier. Data within this tier is protected and, like the hot tier, can be composed of larger capacity SSDs or HDDs. Both incremental and full backups are supported using common data protection solutions.	StorageGRID

Splunk distributed deployments

To support larger environments in which data originates on many machines, you need to process large volumes of data. If many users need to search the data, you can scale the deployment by distributing Splunk Enterprise instances across multiple machines. This is known as a distributed deployment.

In a typical distributed deployment, each Splunk Enterprise instance performs a specialized task and resides on one of three processing tiers corresponding to the main processing functions.

The following table lists the Splunk Enterprise processing tiers.

Tier	Component	Description
Data input	Forwarder	A forwarder consumes data and then forwards the data to a group of indexers.
Indexing	Indexer	An indexer indexes incoming data that it usually receives from a group of forwarders. The indexer transforms the data into events and stores the events in an index. The indexer also searches the indexed data in response to search requests from a search head.
Search management	Search head	A search head serves as a central resource for searching. The search heads in a cluster are interchangeable and have access to the same searches, dashboards, knowledge objects, and so on, from any member of the search head cluster.

The following table lists the important components used in a distributed Splunk Enterprise environment.

Component	Description	Responsibility
Index cluster master	Coordinates activities and updates of an indexer cluster	Index management

Component	Description	Responsibility
Index cluster	Group of Splunk Enterprise indexers that are configured to replicate data with each other	Indexing
Search head deployer	Handles deployment and updates to the cluster master	Search head management
Search head cluster	Group of search heads that serves as a central resource for searching	Search management
Load Balancers	Used by clustered components to handle increasing demand by search heads, indexers and S3 target to distribute load across clustered components.	Load Management for clustered components

See the following benefits of Splunk Enterprise distributed deployments:

- Access diverse or dispersed data sources
- Provide functionality to handle the data needs for enterprises of any size and complexity
- Achieve high availability and ensure disaster recovery with data replication and multisite deployment

Splunk SmartStore

SmartStore is an indexer capability that enables remote object stores such as Amazon S3 to store indexed data. As a deployment's data volume increases, demand for storage typically outpaces demand for compute resources. SmartStore allows you to manage your indexer storage and compute resources cost-effectively by scaling those resources separately.

SmartStore introduces a remote storage tier and a cache manager. These features allow data to reside either locally on indexers or on the remote storage tier. The cache manager manages data movement between the indexer and the remote storage tier, which is configured on the indexer.

With SmartStore, you can reduce the indexer storage footprint to a minimum and choose I/O-optimized compute resources. Most data resides on the remote storage. The indexer maintains a local cache that contains a minimal amount of data: hot buckets, copies of warm buckets participating in active or recent searches, and bucket metadata.

Splunk SmartStore data flow

When data incoming from various sources reaches the indexers, data is indexed and saved locally in a hot bucket. The indexer also replicates the hot bucket data to target indexers. So far, the data flow is identical to the data flow for non-SmartStore indexes.

When the hot bucket rolls to warm, the data flow diverges. The source indexer copies the warm bucket to the remote object store (remote storage tier) while leaving the existing copy in its cache, because searches tend to run across recently indexed data. However, the target indexers delete their copies because the remote store provides high availability without maintaining multiple local copies. The master copy of the bucket now resides in the remote store.

The following image shows the Splunk SmartStore data flow.
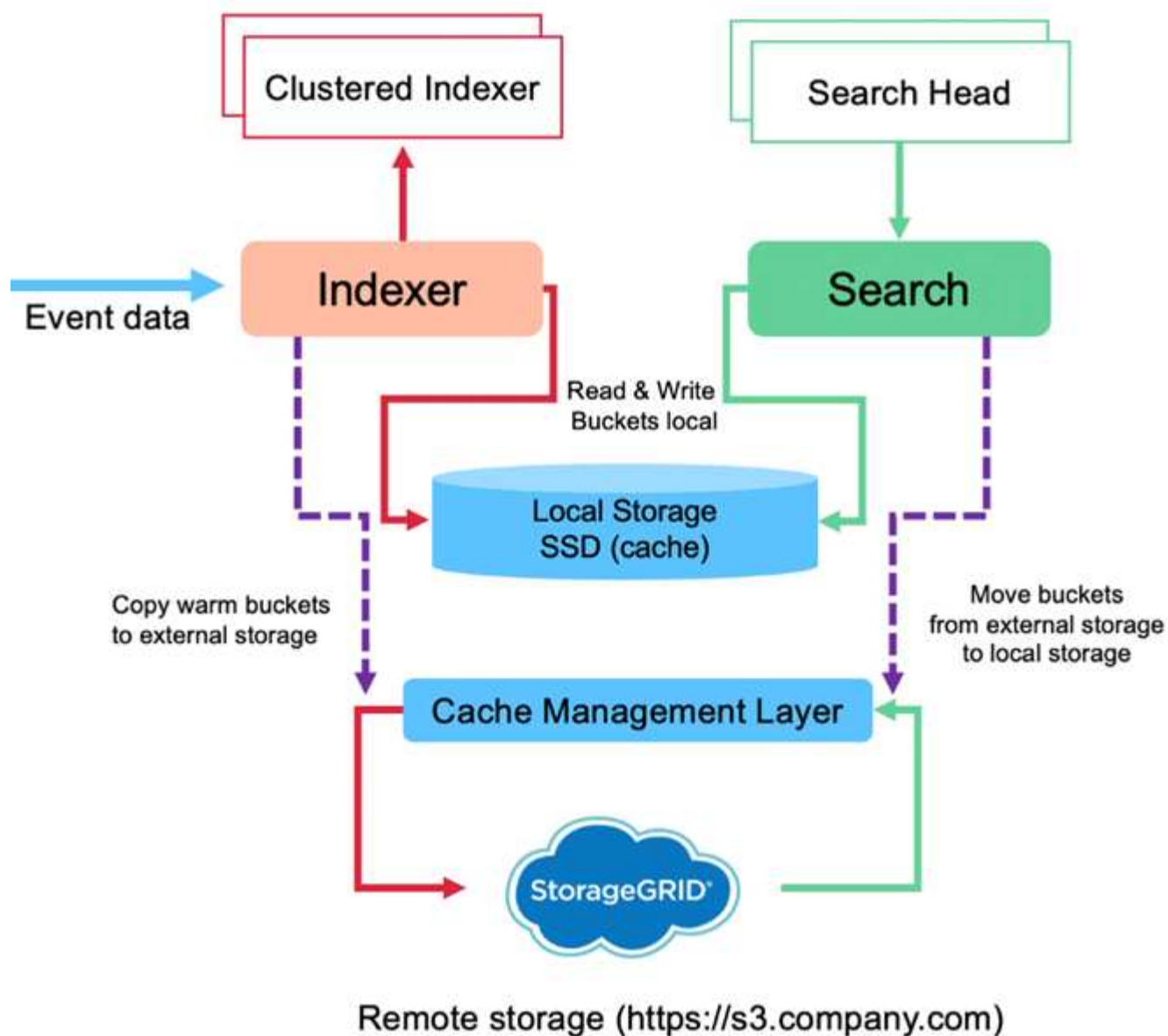
The cache manager on the indexer is central to the SmartStore data flow. It fetches copies of buckets from the remote store as necessary to handle search requests. It also evicts older or less searched copies of buckets from the cache, because the likelihood of their participating in searches decreases over time.

The cache manager's job is to optimize the use of the available cache while ensuring that searches have immediate access to the buckets they need.

Software requirements

The table below lists the software components that are required to implement the solution. The software components that are used in any implementation of the solution might vary based on customer requirements.

Product family	Product name	Product version	Operating system
NetApp StorageGRID	StorageGRID object storage	11.6	n/a

Product family	Product name	Product version	Operating system
CentOS	CentOS	8.1	CentOS 7.x
Splunk Enterprise	Splunk Enterprise with SmartStore	8.0.3	CentOS 7.x

Single and multisite requirements

In an Enterprise Splunk environment (medium and large deployments) where data originates on many machines and where many users need to search the data, you can scale your deployment by distributing Splunk Enterprise instances across single and multiple sites.

See the following benefits of Splunk Enterprise distributed deployments:

- Access diverse or dispersed data sources
- Provide functionality to handle the data needs for enterprises of any size and complexity
- Achieve high availability and ensure disaster recovery with data replication and multisite deployment
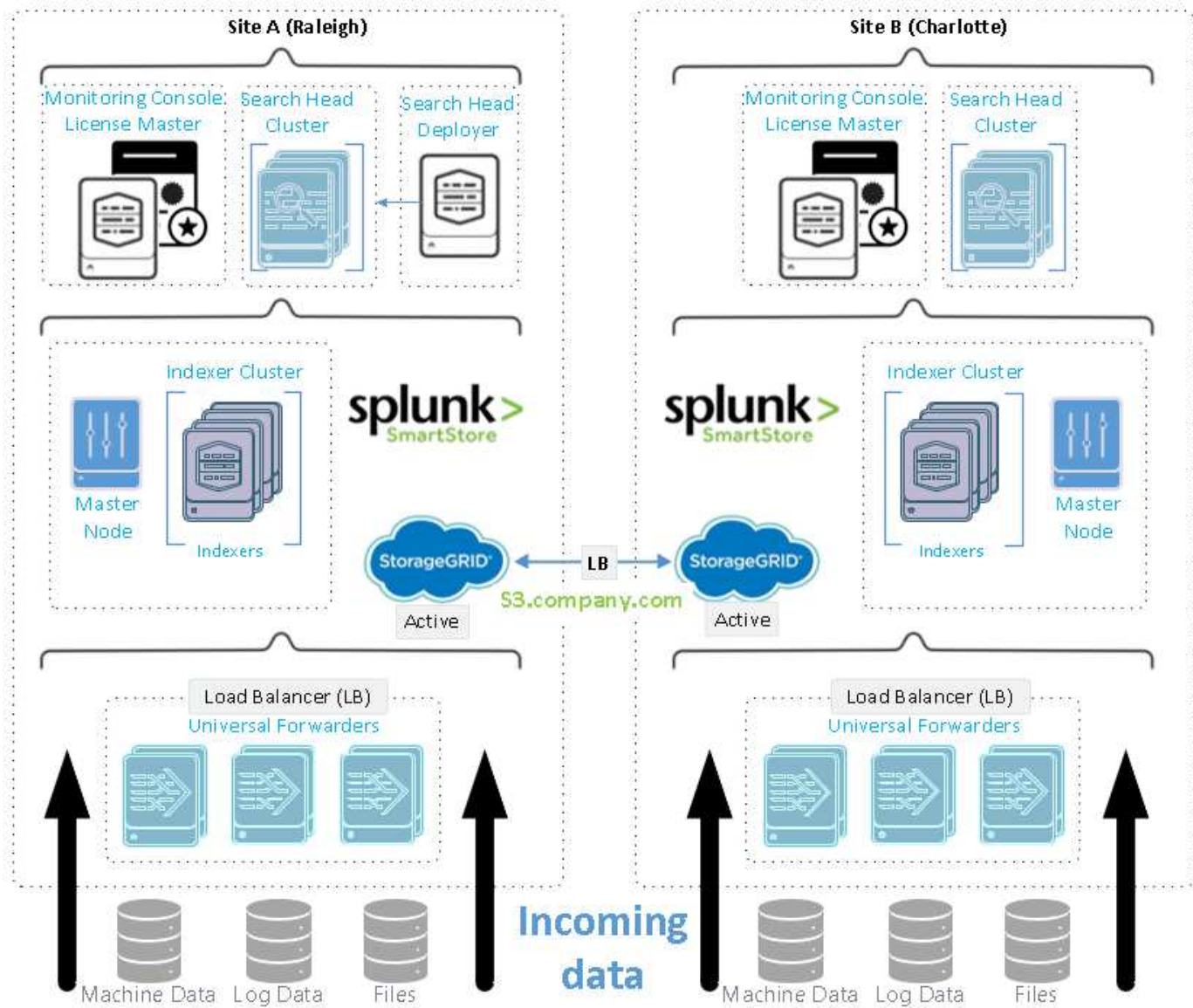
The following table lists the components used in a distributed Splunk Enterprise environment.

Component	Description	Responsibility
Index cluster master	Coordinates activities and updates of an indexer cluster	Index management
Index cluster	Group of Splunk Enterprise indexers that are configured to replicate each other's data	Indexing
Search head deployer	Handles deployment and updates to the cluster master	Search head management
Search head cluster	Group of search heads that serves as a central resource for searching	Search management
Load balancers	Used by clustered components to handle increasing demand by search heads, indexers and S3 target to distribute load across clustered components.	Load management for clustered components

This figure depicts an example of a single-site distributed deployment.

Single Site Deployment
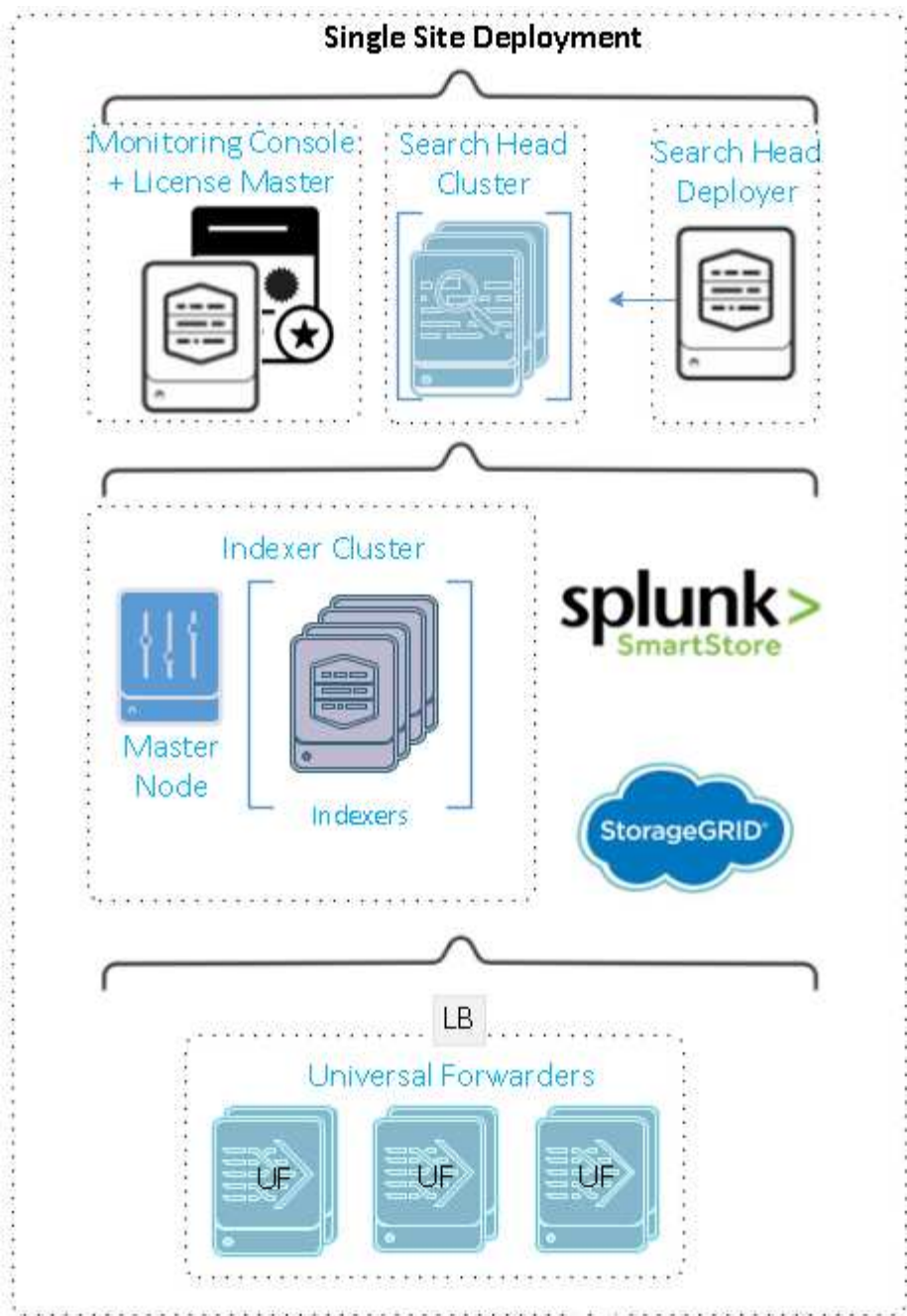
This figure depicts an example of a multisite distributed deployment.

Hardware requirements

The following tables list the minimum number of hardware components that are required to implement the solution. The hardware components that are used in specific implementations of the solution might vary based on customer requirements.

> ⓘ Regardless of whether you have deployed Splunk SmartStore and StorageGRID in a single site or in multiple sites, all systems are managed from the StorageGRID GRID Manager in a single pane of glass. See the section "Simple Management with Grid Manager" for more details.

This table lists the hardware used for a single site.

Hardware	Quantity	Disk	Usable capacity	Note
StorageGRID SG1000	1	n/a	n/a	Admin node and load balancer
StorageGRID SG6060	4	x48, 8TB (NL-SAS HDD)	1PB	Remote storage

This table lists the hardware used for a multisite configuration (per site).

| Hardware	Quantity	Disk	Usable capacity	Note
StorageGRID SG1000	2	n/a	n/a	Admin node and Load balancer
StorageGRID SG6060	4	x48, 8TB (NL-SAS HDD)	1PB	Remote storage

NetApp StorageGRID Load Balancer: SG1000

Object storage requires the use of a load balancer to present the cloud storage namespace. StorageGRID supports third- party load balancers from leading vendors like F5 and Citrix, but many customers choose the enterprise-grade StorageGRID balancer for simplicity, resiliency, and high performance. The StorageGRID load balancer is available as a VM, container, or purpose-built appliance.

The StorageGRID SG1000 facilitates the use of high availability (HA) groups and intelligent load balancing for S3 data-path connections. No other on-prem object storage system provides a customized load balancer.

The SG1000 appliance provides the following features:

- A load balancer and, optionally, admin node functions for a StorageGRID system
- The StorageGRID Appliance Installer to simplify node deployment and configuration
- Simplified configuration of S3 endpoints and SSL
- Dedicated bandwidth (versus sharing a third-party load balancer with other applications)
- Up to 4 x 100Gbps aggregate Ethernet bandwidth

The following image shows the SG1000 Gateway Services appliance.
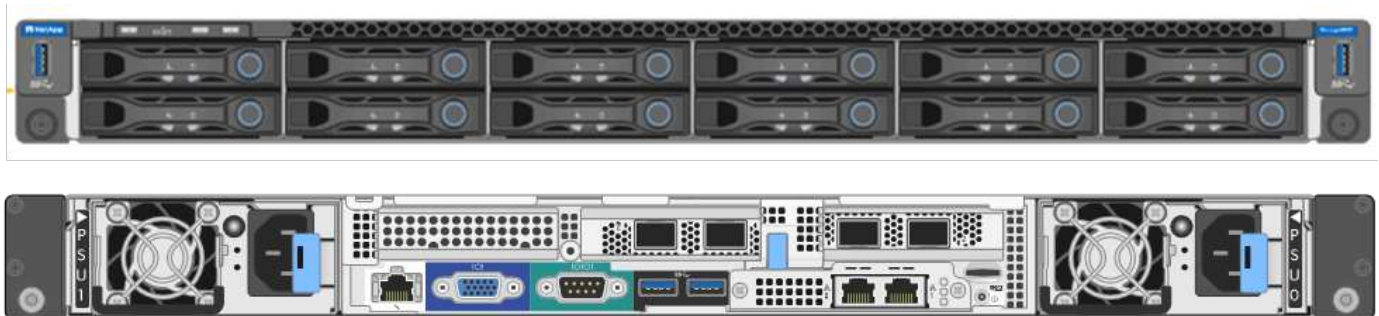
SG6060

The StorageGRID SG6060 appliance includes a compute controller (SG6060) and a storage controller shelf (E-Series E2860) that contains two storage controllers and 60 drives. This appliance provides the following features:

- Scale up to 400PB in a single namespace.
- Up to 4x 25Gbps aggregate Ethernet bandwidth.
- Includes the StorageGRID Appliance Installer to simplify node deployment and configuration.
- Each SG6060 appliance can have one or two additional expansion shelves for a total of 180 drives.

- Two E-Series E2800 controllers (duplex configuration) to provide storage controller failover support.
- Five-drawer drive shelf that holds sixty 3.5-inch drives (two solid-state drives, and 58 NL-SAS drives).

The following image shows the SG6060 appliance.

Splunk design

The following table lists the Splunk configuration for a single site.

Splunk component	Task	Quantity	Cores	Memory	OS
Universal forwarder	Responsible for ingesting data and forwarding data to the indexers	4	16 Cores	32GB RAM	CentOS 8.1

Splunk component	Task	Quantity	Cores	Memory	OS
Indexer	Manages the user data	10	16 Cores	32GB RAM	CentOS 8.1
Search head	User front end searches data in indexers	3	16 Cores	32GB RAM	CentOS 8.1
Search head deployer	Handles updates for search head clusters	1	16 Cores	32GB RAM	CentOS 8.1
Cluster master	Manages the Splunk installation and indexers	1	16 Cores	32GB RAM	CentOS 8.1
Monitoring Console and license master	Performs centralized monitoring of the entire Splunk deployment and manages Splunk licenses	1	16 Cores	32GB RAM	CentOS 8.1

The following tables describe the Splunk configuration for multisite configurations.

This table lists the Splunk configuration for a multisite configuration (site A).

Splunk component	Task	Quantity	Cores	Memory	OS
Universal forwarder	Responsible for ingesting data and forwarding data to the indexers.	4	16 Cores	32GB RAM	CentOS 8.1
Indexer	Manages the user data	10	16 Cores	32GB RAM	CentOS 8.1
Search head	User front end searches data in indexers	3	16 Cores	32GB RAM	CentOS 8.1
Search head deployer	Handles updates for search head clusters	1	16 Cores	32GB RAM	CentOS 8.1
Cluster master	Manages the Splunk installation and indexers	1	16 Cores	32GB RAM	CentOS 8.1

Splunk component	Task	Quantity	Cores	Memory	OS
Monitoring Console and license master	Performs centralized monitoring of the entire Splunk deployment and manages Splunk licenses.	1	16 Cores	32GB RAM	CentOS 8.1

This table lists the Splunk configuration for a multisite configuration (site B).

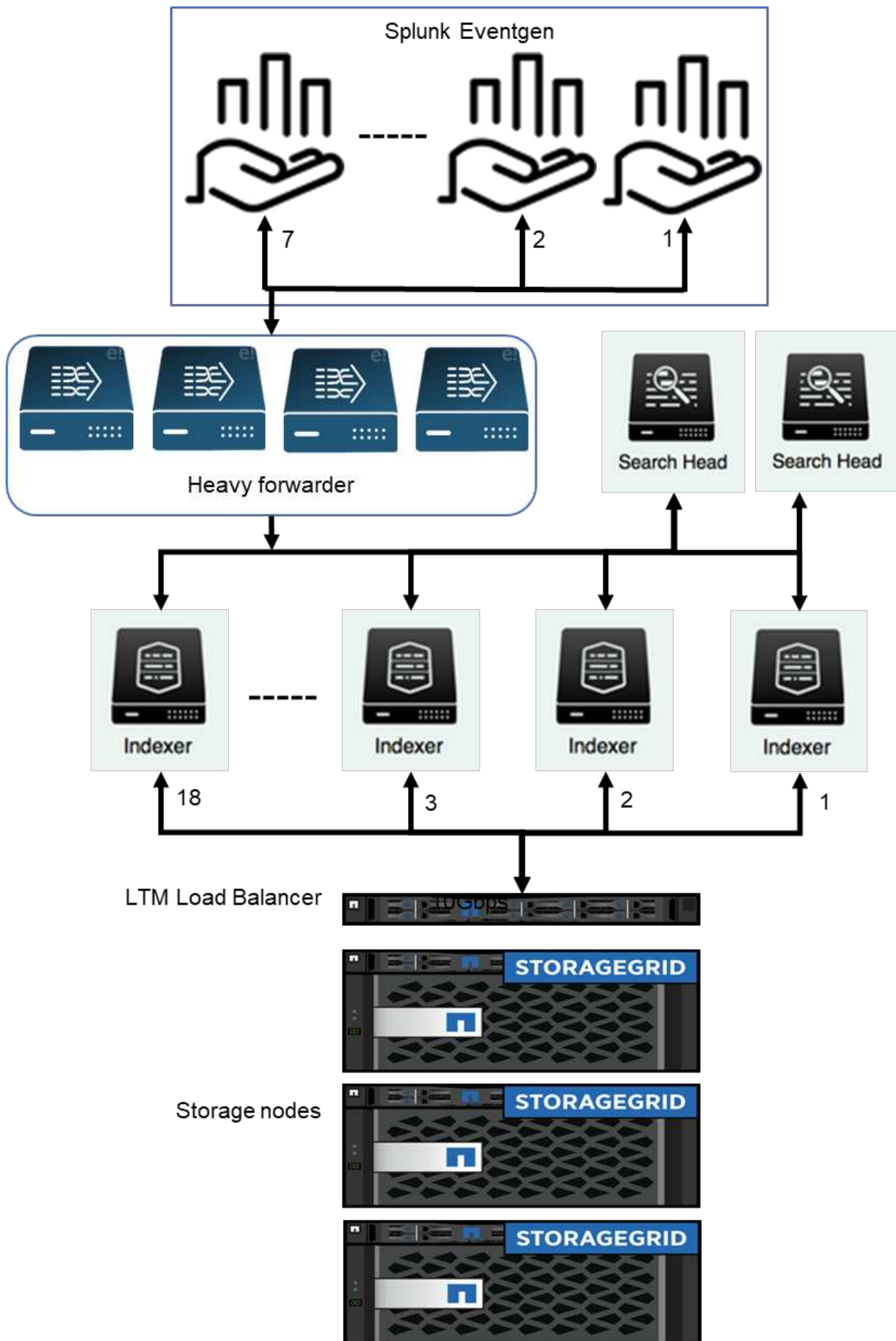| Splunk component	Task	Quantity	Cores	Memory	OS
Universal forwarder	Responsible for ingesting data and forwarding data to the indexers	4	16 Cores	32GB RAM	CentOS 8.1
Indexer	Manages the user data	10	16 Cores	32GB RAM	CentOS 8.1
Search head	User front end searches data in indexers	3	16 Cores	32GB RAM	CentOS 8.1
Cluster master	Manages the Splunk installation and indexers	1	16 Cores	32GB RAM	CentOS 8.1
Monitoring Console and license master	Performs centralized monitoring of the entire Splunk deployment and manages Splunk licenses	1	16 Cores	32GB RAM	CentOS 8.1

Single-site SmartStore performance

This section describes Splunk SmartStore performance on a NetApp StorageGRID controller. Splunk SmartStore moves warm data to remote storage, which in this case is StorageGRID object storage in the performance validation.

Hot/cache storage
EF600

StorageGRID 6060
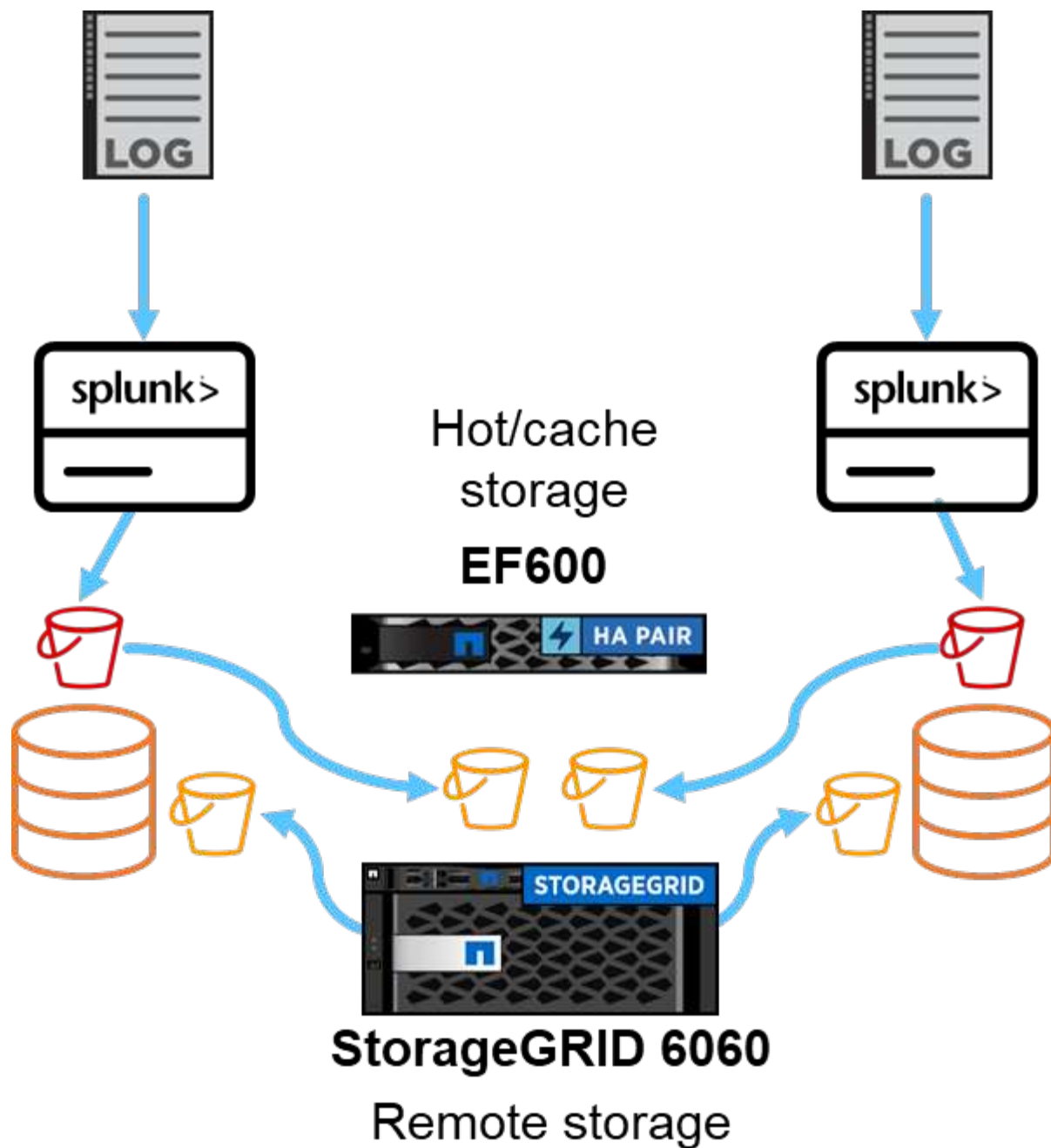
Remote storage

We used EF600 for hot/cache storage and StorageGRID 6060 for remote storage. We used the following architecture for the performance validation. We used two search heads, four heavy forwarders to forward the data to indexers, seven Splunk Event Generators (Eventgens) to generate the real-time data, and 18 indexers to store the data.

Splunk Eventgen

7   2   1

Heavy forwarder

Search Head   Search Head

Indexer   Indexer   Indexer   Indexer

18   3   2   1

LTM Load Balancer   10 Gbps

STORAGEGRID

Storage nodes   STORAGEGRID

STORAGEGRID

Configuration

This table lists the hardware used for the SmartStorage performance validation.

Splunk component	Task	Quantity	Cores	Memory	OS
Heavy forwarder	Responsible for ingesting data and forwarding data to the indexers	4	16 cores	32GB RAM	SLED 15 SP2
Indexer	Manages the user data	18	16 cores	32GB RAM	SLED 15 SP2
Search head	User front-end searches data in indexers	2	16 cores	32GB RAM	SLED 15 SP2
Search head deployer	Handles updates for search head clusters	1	16 cores	32GB RAM	SLED 15 SP2
Cluster master	Manages the Splunk installation and indexers	1	16 cores	32GB RAM	SLED 15 SP2
Monitoring Console and license master	Performs centralized monitoring of the entire Splunk deployment and manages Splunk licenses	1	16 cores	32GB RAM	SLED 15 SP2

SmartStore remote store performance validation

In this performance validation, we configured the SmartStore cache in local storage on all the indexers for 10 days of data. We enabled the `maxDataSize=auto` (750MB bucket size) in Splunk cluster manager and pushed the changes to all the indexers. To measure the upload performance, we ingested 10TB per day for 10 days and rolled over all hot buckets to warm at the same time and captured the peak and average throughput per instance and deployment-wide from the SmartStore Monitoring Console dashboard.

This image shows the data ingested in one day.

Enterprise license group   ⇄ Change license group

This server is configured to use licenses from the **Enterprise license group**

[Add license]   [Usage report]

Alerts

Licensing alerts notify you of excessive indexing warnings and licensing misconfigurations. ⎆ Learn more

Current

●	1 pool warning reported by 1 indexer	Correct by midnight to avoid warning ⎆ Learn more
●	1 pool quota overage warning reported by 1 indexer	Correct by midnight to avoid warning ⎆ Learn more

Permanent

●	48 pool quota overage warnings reported by 12 indexers	1 day ago

Splunk Internal License DO NOT DISTRIBUTE stack   ⎆ Learn more

Licenses		Volume	Expiration	Status	
Splunk Internal License DO NOT DISTRIBUTE	Notes	2,097,152 MB	Oct 15, 2021, 2:59:59 AM	expired	Delete
Splunk Internal License DO NOT DISTRIBUTE	Notes	10,485,760 MB	Jul 2, 2022, 2:59:59 AM	valid	Delete
Effective daily volume		**10,485,760 MB**			

Pools	Indexers	Volume used today	
auto_generated_pool_enterprise		▆▆▆▆▆▆ 10,878,328 MB / 10,485,760 MB	Edit / Delete
	rtp-idx0005	902,186 MB (8.604%)	
	rtp-idx0006	766,053 MB (7.306%)	
	rtp-idx0010	943,927 MB (9.002%)	
	rtp-idx0008	931,854 MB (8.887%)	
	rtp-idx0001	855,659 MB (8.16%)	
	rtp-idx0012	949,412 MB (9.054%)	
	rtp-idx0011	910,235 MB (8.681%)	
	rtp-idx0002	906,379 MB (8.644%)	
	rtp-idx0007	963,664 MB (9.19%)	
	rtp-idx0009	949,847 MB (9.058%)	
	rtp-idx0003	883,446 MB (8.425%)	
	rtp-idx0004	915,666 MB (8.732%)	

[⊕ Add pool]

Local server information

Indexer name	rtp-mc-lm
Volume used today	0 MB
Warning count	0
Debug information	All license details
	All indexer details

We ran the following command from cluster master (the index name is `eventgen-test`). Then we captured the peak and average upload throughput per instance and deployment-wide through the SmartStore Monitoring Console dashboards.

```
for i in rtp-idx0001 rtp-idx0002 rtp-idx0003 rtp-idx0004 rtp-idx0005 rtp-
idx0006 rtp-idx0007 rtp-idx0008 rtp-idx0009 rtp-idx0010 rtp-idx0011 rtp-
idx0012 rtp-idx0013011 rtdx0014 rtp-idx0015 rtp-idx0016 rtp-idx0017 rtp-
idx0018 ; do  ssh $i "hostname;  date; /opt/splunk/bin/splunk _internal
call /data/indexes/eventgen-test/roll-hot-buckets -auth admin:12345678;
sleep 1  "; done
```

(i)   The cluster master has password-less authentication to all indexers (rtp-idx0001…rtp-idx0018).

To measure the download performance, we evicted all data from the cache by running the evict CLI twice by using the following command.

(i)   We ran the following command from cluster master and ran the search from the search head on top of 10 days of data from the remote store from StorageGRID. We then captured the peak and average upload throughput per instance and deployment-wide through the SmartStore Monitoring Console dashboards.

```
for i in rtp-idx0001 rtp-idx0002 rtp-idx0003 rtp-idx0004 rtp-idx0005 rtp-
idx0006 rtp-idx0007 rtp-idx0008 rtp-idx0009 rtp-idx0010 rtp-idx0011 rtp-
idx0012 rtp-idx0013 rtp-idx0014 rtp-idx0015 rtp-idx0016 rtp-idx0017 rtp-
idx0018 ; do  ssh $i " hostname;  date; /opt/splunk/bin/splunk _internal
call /services/admin/cacheman/_evict -post:mb 1000000000 -post:path
/mnt/EF600 -method POST  -auth admin:12345678;   "; done
```

The indexer configurations were pushed from SmartStore cluster master. The cluster master had the following configuration for the indexer.

```
Rtp-cm01:~ # cat /opt/splunk/etc/master-apps/_cluster/local/indexes.conf
[default]
maxDataSize = auto
#defaultDatabase = eventgen-basic
defaultDatabase = eventgen-test
hotlist_recency_secs = 864000
repFactor = auto
[volume:remote_store]
storageType = remote
path = s3://smartstore2
remote.s3.access_key = U64TUHONBNC98GQGL60R
remote.s3.secret_key = UBoXNE0jmECie05Z7iCYVzbSB6WJFckiYLcdm2yg
remote.s3.endpoint = 3.sddc.netapp.com:10443
remote.s3.signature_version = v2
remote.s3.clientCert =
[eventgen-basic]
homePath = $SPLUNK_DB/eventgen-basic/db
```

```
coldPath = $SPLUNK_DB/eventgen-basic/colddb
thawedPath = $SPLUNK_DB/eventgen-basic/thawed
[eventgen-migration]
homePath = $SPLUNK_DB/eventgen-scale/db
coldPath = $SPLUNK_DB/eventgen-scale/colddb
thawedPath = $SPLUNK_DB/eventgen-scale/thaweddb
[main]
homePath = $SPLUNK_DB/$_index_name/db
coldPath = $SPLUNK_DB/$_index_name/colddb
thawedPath = $SPLUNK_DB/$_index_name/thaweddb
[history]
homePath = $SPLUNK_DB/$_index_name/db
coldPath = $SPLUNK_DB/$_index_name/colddb
thawedPath = $SPLUNK_DB/$_index_name/thaweddb
[summary]
homePath = $SPLUNK_DB/$_index_name/db
coldPath = $SPLUNK_DB/$_index_name/colddb
thawedPath = $SPLUNK_DB/$_index_name/thaweddb
[remote-test]
homePath = $SPLUNK_DB/$_index_name/db
coldPath = $SPLUNK_DB/$_index_name/colddb
#for storagegrid config
remotePath = volume:remote_store/$_index_name
thawedPath = $SPLUNK_DB/$_index_name/thaweddb
[eventgen-test]
homePath = $SPLUNK_DB/$_index_name/db
maxDataSize=auto
maxHotBuckets=1
maxWarmDBCount=2
coldPath = $SPLUNK_DB/$_index_name/colddb
#for storagegrid config
remotePath = volume:remote_store/$_index_name
thawedPath = $SPLUNK_DB/$_index_name/thaweddb
[eventgen-evict-test]
homePath = $SPLUNK_DB/$_index_name/db
coldPath = $SPLUNK_DB/$_index_name/colddb
#for storagegrid config
remotePath = volume:remote_store/$_index_name
thawedPath = $SPLUNK_DB/$_index_name/thaweddb
maxDataSize = auto_high_volume
maxWarmDBCount = 5000
rtp-cm01:~ #
```
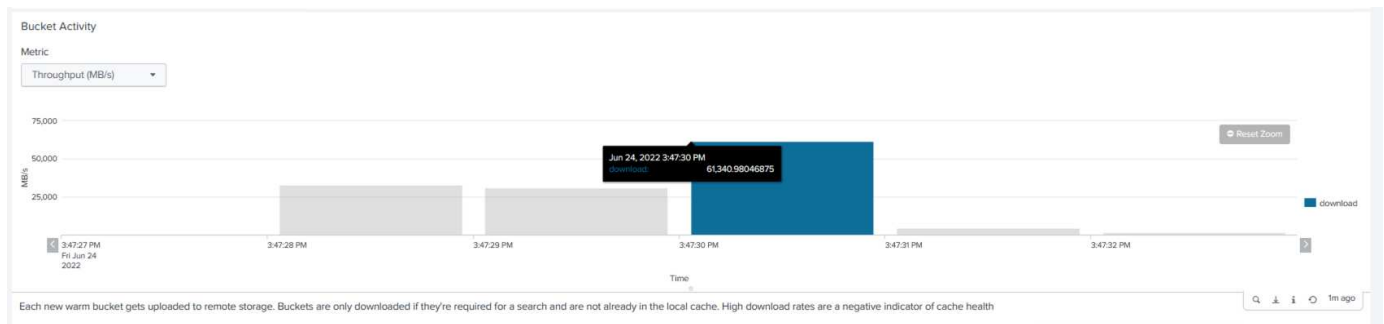
We ran the following search query on the search head to collect the performance matrix.

We collected the performance information from the cluster master. The peak performance was 61.34GBps.

The average performance was approximately 29GBps.

Bucket Activity

Metric

Throughput (MB/s) ▾
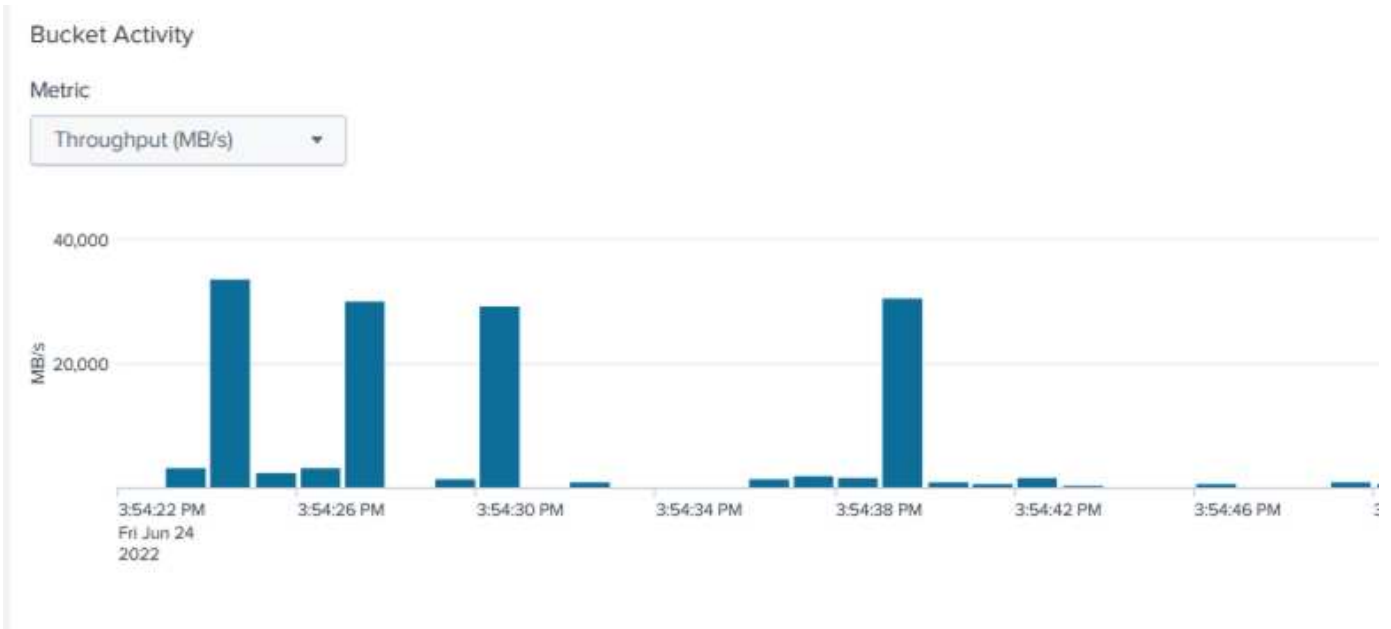
StorageGRID performance

SmartStore performance is based on searching for specific patterns and strings from large amounts of data. In this validation, the events are generated using Eventgen on a specific Splunk index (eventgen-test) through the search head, and the request goes to StorageGRID for most of the queries. The following image shows the hits and misses of the query data. The hits data is from the local disk and the misses data is from the StorageGRID controller.

ⓘ | The green color shows the hits data and the orange color shows the misses data.
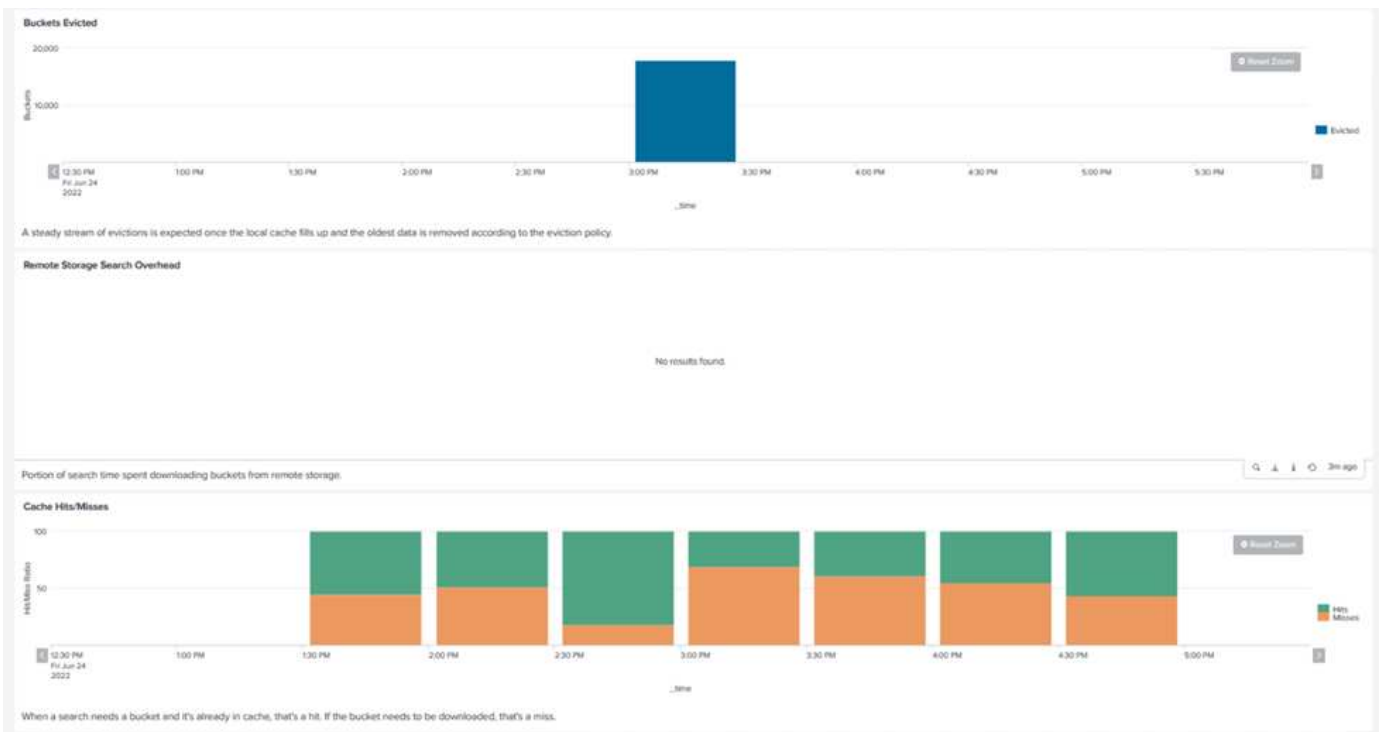
When the query runs for the search on StorageGRID, the time for the S3 retrieve rate from StorageGRID is shown in the following image.

SmartStore-Site-1 (Site) ☑

| Network | Storage | **Objects** | ILM | Platform services | Load b |

1 hour   **1 day**   1 week

S3 ingest and retrieve ❓
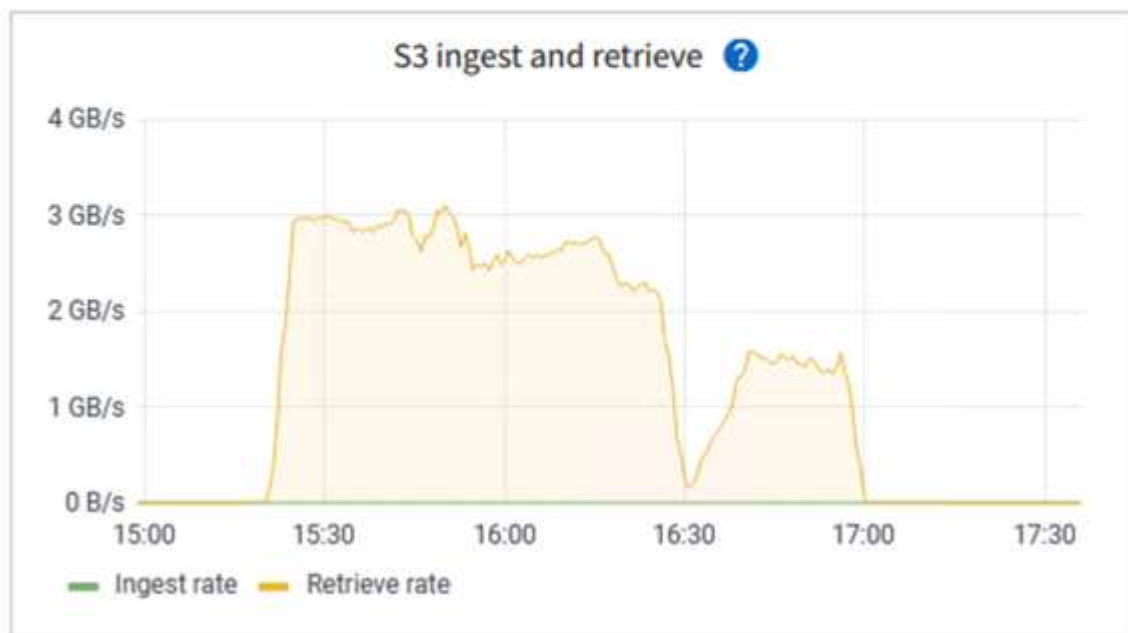
StorageGRID hardware usage

The StorageGRID instance has one load balancer and three StorageGRID controllers. CPU utilization for all three controllers is from 75% to 100%.

NetApp-CIG-SGA01 (Storage Node)

Overview | Hardware | Network | Storage | Objects | ILM

1 hour   1 day   1 week

CPU utilization

NetApp-CIG-SGA02 (Storage Node)

Overview | Hardware | Network | Storage | Objects | ILM | T

1 hour   1 day   1 week

CPU utilization

NetApp-CIG-SGA03 (Storage Node)

Overview | Hardware | Network | Storage | Objects | ILM | Ta

1 hour   1 day   1 week

CPU utilization
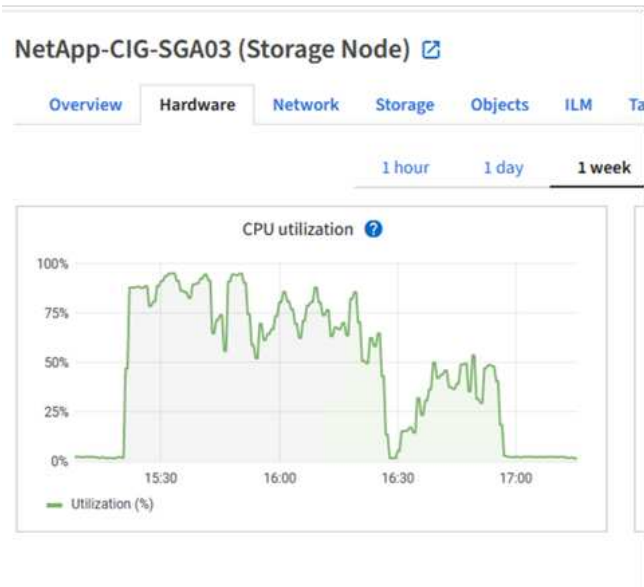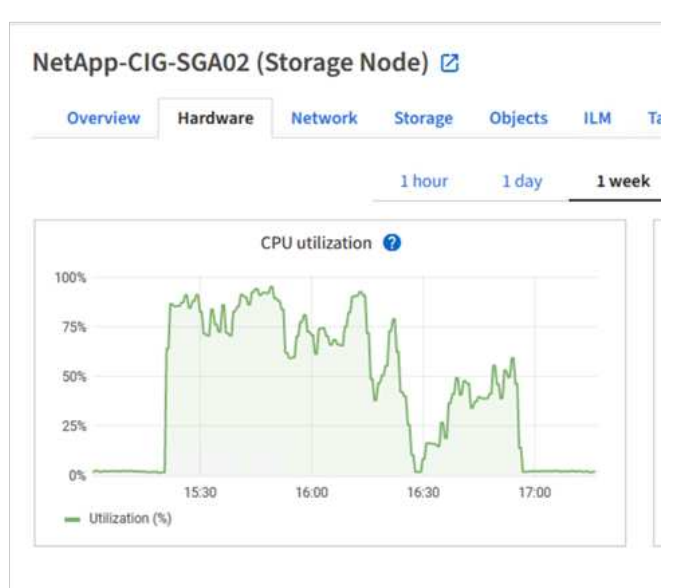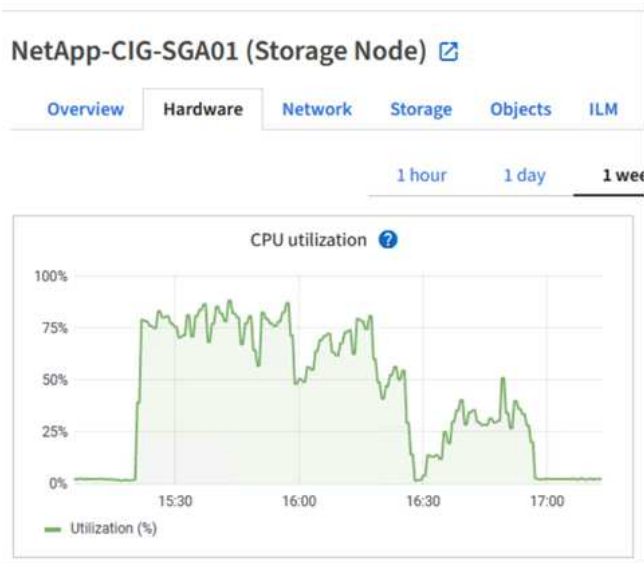
SmartStore with NetApp storage controller - benefits for the customer

- **Decoupling compute and storage.** The Splunk SmartStore decouples compute and storage, which helps you to scale them independently.

- **Data on-demand.** SmartStore brings data close to compute on-demand and provides compute and storage elasticity and cost efficiency to achieve longer data retention at scale.

- **AWS S3 API compliant.** SmartStore uses the AWS S3 API to communicate with restore storage, which is an AWS S3 and S3 API-compliant object store such as StorageGRID.

- **Reduces storage requirement and cost.** SmartStore reduces the storage requirements for aged data (warm/cold). It only needs a single copy of data because NetApp storage provides data protection and takes care of failure and high availability.

- **Hardware failure.** Node failure in a SmartStore deployment does not make the data inaccessible and has a much faster indexer recovery from hardware failure or data imbalance.

- Application and data-aware cache.

- Add-remove indexers and setup-teardown cluster on-demand.

- Storage tier is no longer tied to hardware.

Conclusion

Splunk Enterprise is the market-leading SIEM solution driving outcomes across Security, IT, and DevOps teams. The use of Splunk has increased considerably across our customer's organizations. Therefore, there is a need to add more data sources while also retaining the data for a longer period, thus stressing the Splunk infrastructure.

The combination of Splunk SmartStore and NetApp StorageGRID is designed to provide a scalable architecture for organizations to achieve improved ingest performance with SmartStore and StorageGRID object storage and increased scalability for a Splunk environment across multiple geographical regions.

Where to find additional information

To learn more about the information that is described in this document, review the following documents and/or websites:

- NetApp StorageGRID Documentation Resources

  https://www.netapp.com/data-storage/storagegrid/documentation/

- NetApp Product Documentation

  https://docs.netapp.com

- Splunk Enterprise Documentation

  https://docs.splunk.com/Documentation/Splunk

- Splunk Enterprise About SmartStore

  https://docs.splunk.com/Documentation/Splunk/8.0.6/Indexer/AboutSmartStore

- Splunk Enterprise Distributed Deployment Manual

  https://docs.splunk.com/Documentation/Splunk/8.0.6/Deploy/Distributedoverview

- Splunk Enterprise Managing Indexers and Clusters of Indexers

  https://docs.splunk.com/Documentation/Splunk/8.0.6/Indexer/Aboutindexesandindexers

Version history

Version	Date	Document version history
1.0	July 2022	Initial release.

Big Data Analytics Data to Artificial Intelligence

TR-4732: Big data analytics data to artificial intelligence

Karthikeyan Nagalingam, NetApp

This document describes how to move big-data analytics data and HPC data to AI. AI processes NFS data through NFS exports, whereas customers often have their AI data in a big-data analytics platform, such as HDFS, Blob, or S3 storage as well as HPC platforms such as GPFS. This paper provides guidelines for moving big-data-analytics data and HPC data to AI by using NetApp XCP and NIPAM. We also discuss the business benefits of moving data from big data and HPC to AI.

Concepts and components

Big data analytics storage

Big data analytics is the major storage provider for HDFS. A customer often uses a Hadoop-compatible file system (HCFS) such as Windows Azure Blob Storage, MapR File System (MapR-FS), and S3 object storage.

General parallel file system

IBM's GPFS is an enterprise file system that provides an alternative to HDFS. GPFS provides flexibility for applications to decide the block size and replication layout, which provide good performance and efficiency.

NetApp In-Place Analytics Module

The NetApp In-Place Analytics Module (NIPAM) serves as a driver for Hadoop clusters to access NFS data. It has four components: a connection pool, an NFS InputStream, a file handle cache, and an NFS OutputStream. For more information, see TR-4382: NetApp In-Place Analytics Module.

Hadoop Distributed Copy

Hadoop Distributed Copy (DistCp) is a distributed copy tool used for large inter-cluster and intra-cluster coping tasks. This tool uses MapReduce for data distribution, error handling, and reporting. It expands the list of files and directories and inputs them to map tasks to copy the data from the source list. The image below shows the DistCp operation in HDFS and nonHDFS.

Hadoop distcp basic process

Hadoop distcp and NetApp In-Place Analytics Module
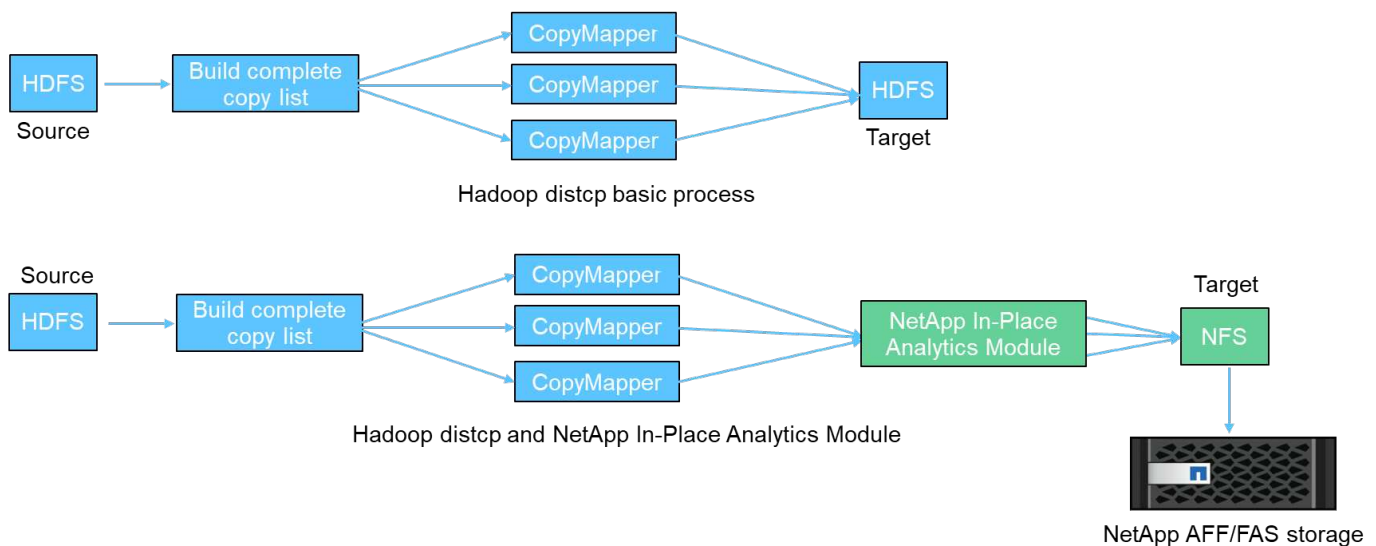
Hadoop DistCp moves data between the two HDFS systems without using an additional driver. NetApp provides the driver for non-HDFS systems. For an NFS destination, NIPAM provides the driver to copy data that Hadoop DistCp uses to communicate with NFS destinations when copying data.

30

NetApp Cloud Volumes Service

The NetApp Cloud Volumes Service is a cloud-native file service with extreme performance. This service helps customers accelerate their time-to-market by rapidly spinning resources up and down and using NetApp features to improve productivity and reduce staff downtime. The Cloud Volumes Service is the right alternative for disaster recovery and back up to cloud because it reduces the overall data-center footprint and consumes less native public cloud storage.

NetApp XCP

NetApp XCP is client software that enables fast and reliable any-to-NetApp and NetApp-to-NetApp data migration. This tool is designed to copy a large amount of unstructured NAS data from any NAS system to a NetApp storage controller. The XCP Migration Tool uses a multicore, multichannel I/O streaming engine that can process many requests in parallel, such as data migration, file or directory listings, and space reporting. This is the default NetApp data Migration Tool. You can use XCP to copy data from a Hadoop cluster and HPC to NetApp NFS storage. The diagram below shows data transfer from a Hadoop and HPC cluster to a NetApp NFS volume using XCP.
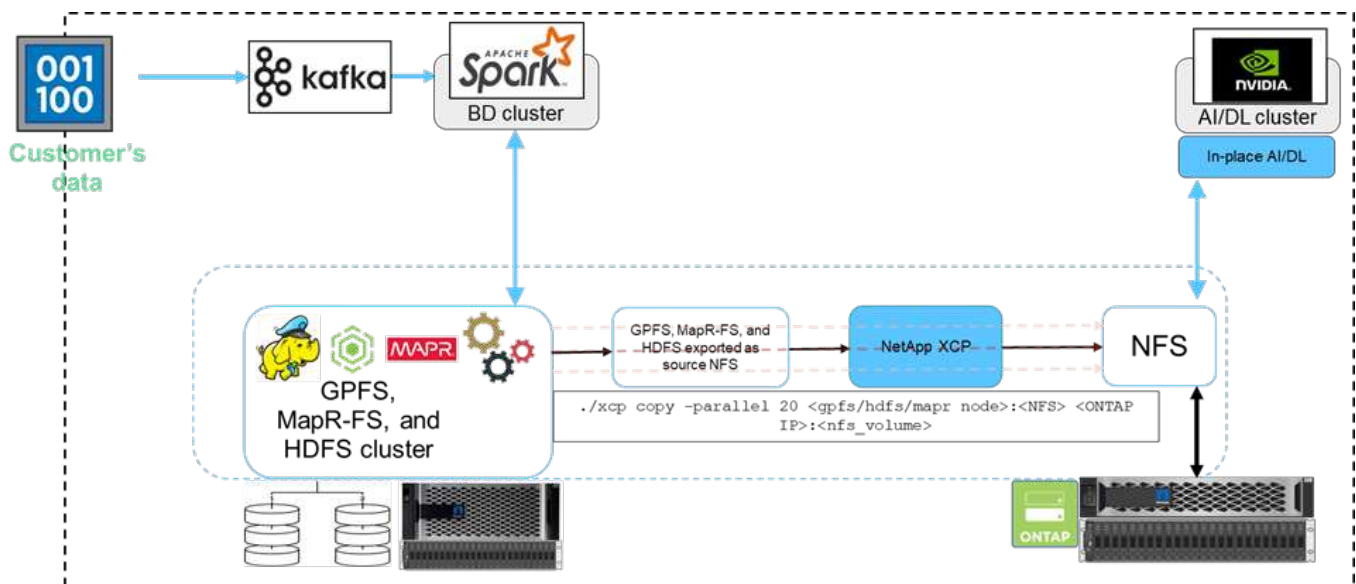
NetApp Cloud Sync

NetApp Cloud Sync is a hybrid data replication software-as-a-service that transfers and synchronizes NFS, S3, and CIFS data seamlessly and securely between on-premises storage and cloud storage. This software is used for data migration, archiving, collaboration, analytics, and more. After data is transferred, Cloud Sync continuously syncs the data between the source and destination. Going forward, it then transfers the delta. It also secures the data within your own network, in the cloud, or on premises. This software is based on a pay-as-you-go model, which provides a cost-effective solution and provides monitoring and reporting capabilities for your data transfer.

Customer challenges

Customers might face the following challenges when trying to access data from big-data analytics for AI operations:

- Customer data is in a data lake repository. The data lake can contain different types of data such as structured, unstructured, semi-structured, logs, and machine-to-machine data. All these data types must be processed in AI systems.

- AI is not compatible with Hadoop file systems. A typical AI architecture is not able to directly access HDFS and HCFS data, which must be moved to an AI-understandable file system (NFS).

- Moving data lake data to AI typically requires specialized processes. The amount of data in the data lake can be very large. A customer must have an efficient, high-throughput, and cost-effective way to move data into AI systems.

- Syncing data. If a customer wants to sync data between the big-data platform and AI, sometimes the data processed through AI can be used with big data for analytical processing.

Data mover solution

In a big-data cluster, data is stored in HDFS or HCFS, such as MapR-FS, the Windows Azure Storage Blob, S3, or the Google file system. We performed testing with HDFS, MapR-FS, and S3 as the source to copy data to NetApp ONTAP NFS export with the help of NIPAM by using the `hadoop distcp` command from the source.

The following diagram illustrates the typical data movement from a Spark cluster running with HDFS storage to a NetApp ONTAP NFS volume so that NVIDIA can process AI operations.
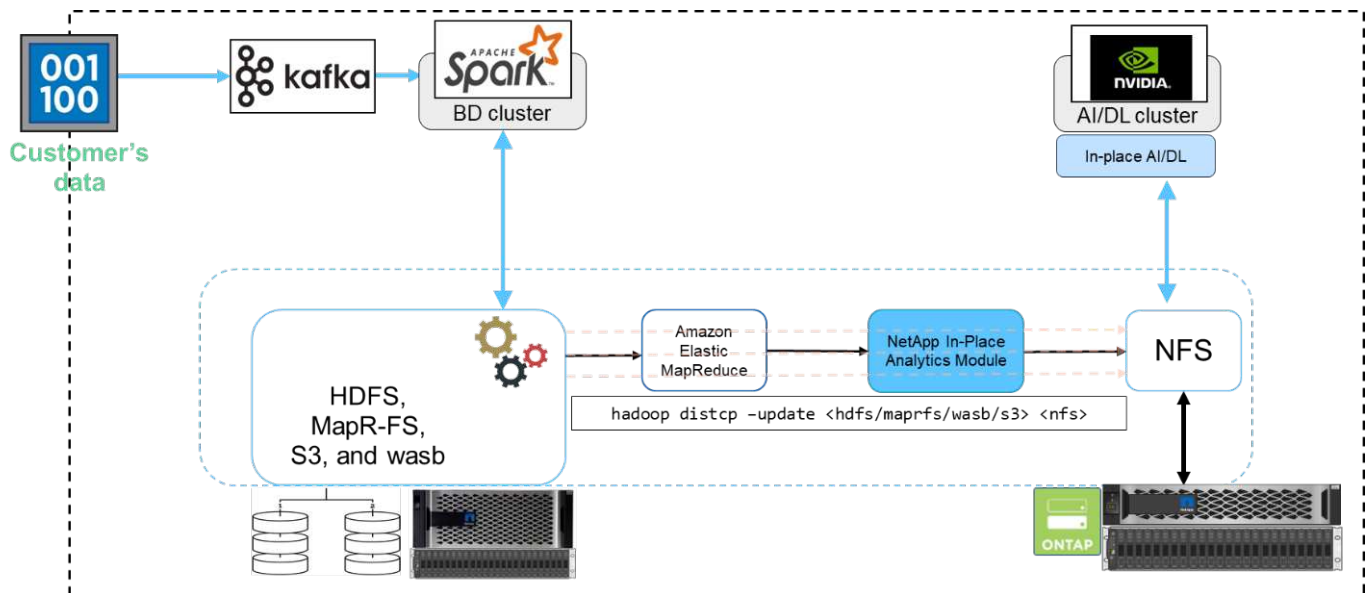
The `hadoop distcp` command uses the MapReduce program to copy the data. NIPAM works with MapReduce to act as a driver for the Hadoop cluster when copying data. NIPAM can distribute a load across multiple network interfaces for a single export. This process maximizes the network throughput by distributing the data across multiple network interfaces when you copy the data from HDFS or HCFS to NFS.

> ⓘ NIPAM is not supported or certified with MapR.

Data mover solution for AI

The data mover solution for AI is based on customers' needs to process Hadoop data from AI operations. NetApp moves data from HDFS to NFS by using the NIPAM. In one use case, the customer needed to move data to NFS on the premises and another customer needed to move data from the Windows Azure Storage Blob to Cloud Volumes Service in order to process the data from the GPU cloud instances in the cloud.

The following diagram illustrates the data mover solution details.
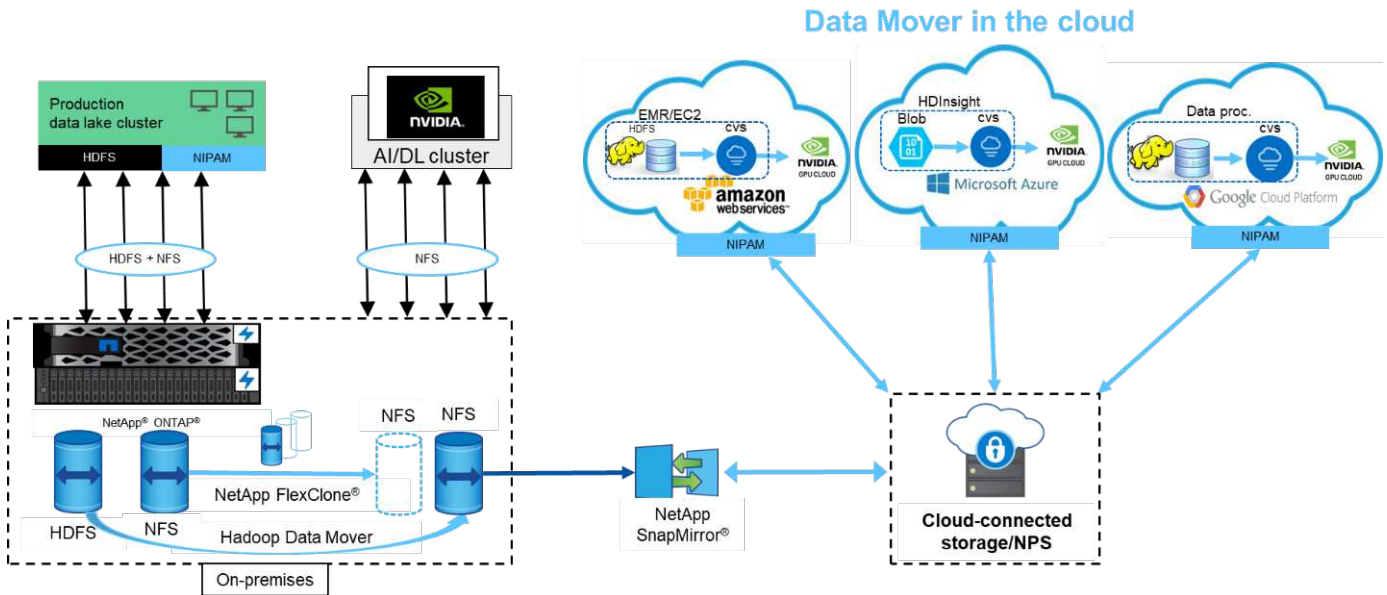
The following steps are required to build the data mover solution:

1. ONTAP SAN provides HDFS, and NAS provides the NFS volume through NIPAM to the production data lake cluster.

2. The customer's data is in HDFS and NFS. The NFS data can be production data from other applications that is used for big data analytics and AI operations.

3. NetApp FlexClone technology creates a clone of the production NFS volume and provisions it to the AI cluster on premises.

4. Data from an HDFS SAN LUN is copied into an NFS volume with NIPAM and the `hadoop distcp` command. NIPAM uses the bandwidth of multiple network interfaces to transfer data. This process reduces the data copy time so that more data can be transferred.

5. Both NFS volumes are provisioned to the AI cluster for AI operations.

6. To process on-the-premises NFS data with GPUs in the cloud, the NFS volumes are mirrored to NetApp Private Storage (NPS) with NetApp SnapMirror technology and mounted to cloud service providers for GPUs.

7. The customer wants to process data in EC2/EMR, HDInsight, or DataProc services in GPUs from cloud service providers. The Hadoop data mover moves the data from Hadoop services to the Cloud Volumes Services with NIPAM and the `hadoop distcp` command.

8. The Cloud Volumes Service data is provisioned to AI through the NFS protocol.Data that is processed through AI can be sent on an on-premises location for big data analytics in addition to the NVIDIA cluster through NIPAM, SnapMirror, and NPS.

In this scenario, the customer has large file-count data in the NAS system at a remote location that is required for AI processing on the NetApp storage controller on premises. In this scenario, it's better to use the XCP Migration Tool to migrate the data at a faster speed.

The hybrid-use-case customer can use Cloud Sync to migrate on-premises data from NFS, CIFS, and S3 data to the cloud and vice versa for AI processing by using GPUs such as those in an NVIDIA cluster. Both Cloud Sync and the XCP Migration Tool are used for the NFS data migration to NetApp ONTAP NFS.

GPFS to NetApp ONTAP NFS

In this validation, we used four servers as Network Shared Disk (NSD) servers to provide physical disks for GPFS. GPFS is created on top of the NSD disks to export them as NFS exports so that NFS clients can access them, as shown in the figure below. We used XCP to copy the data from GPFS- exported NFS to a NetApp NFS volume.

NetApp® XCP

NFS v3 clients

NFS
portmap
lockd
mountd
nfsd
statd
ssh

General parallel file system
(cluster file system)

**NSD servers
(RHEL/SUSE Linux)**
(interfaces interacting with storage systems)

JBOD

LUN group

NetApp
ONTAP®
(FCP/NFS)

NFS volume
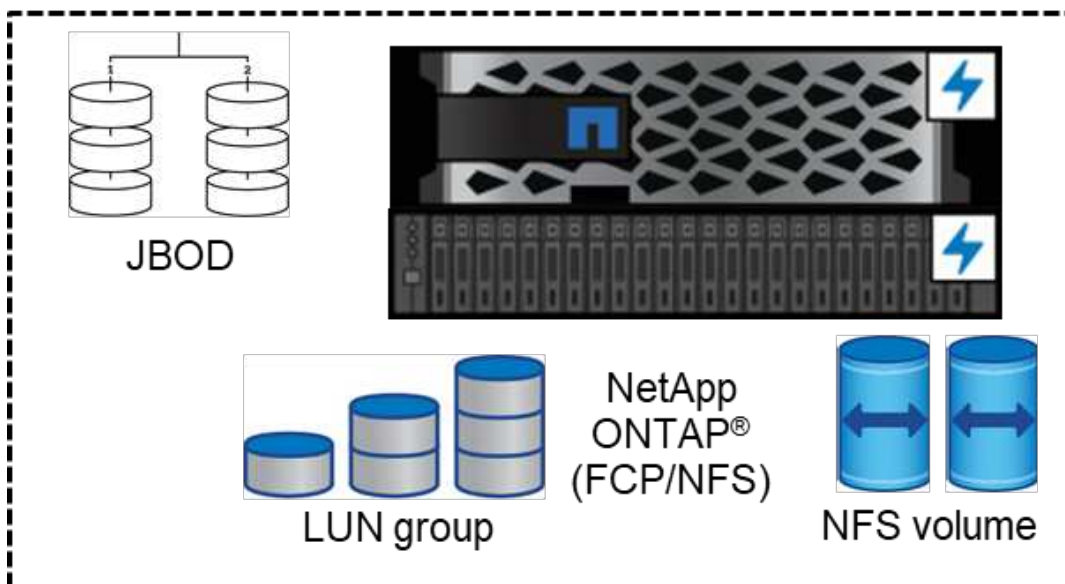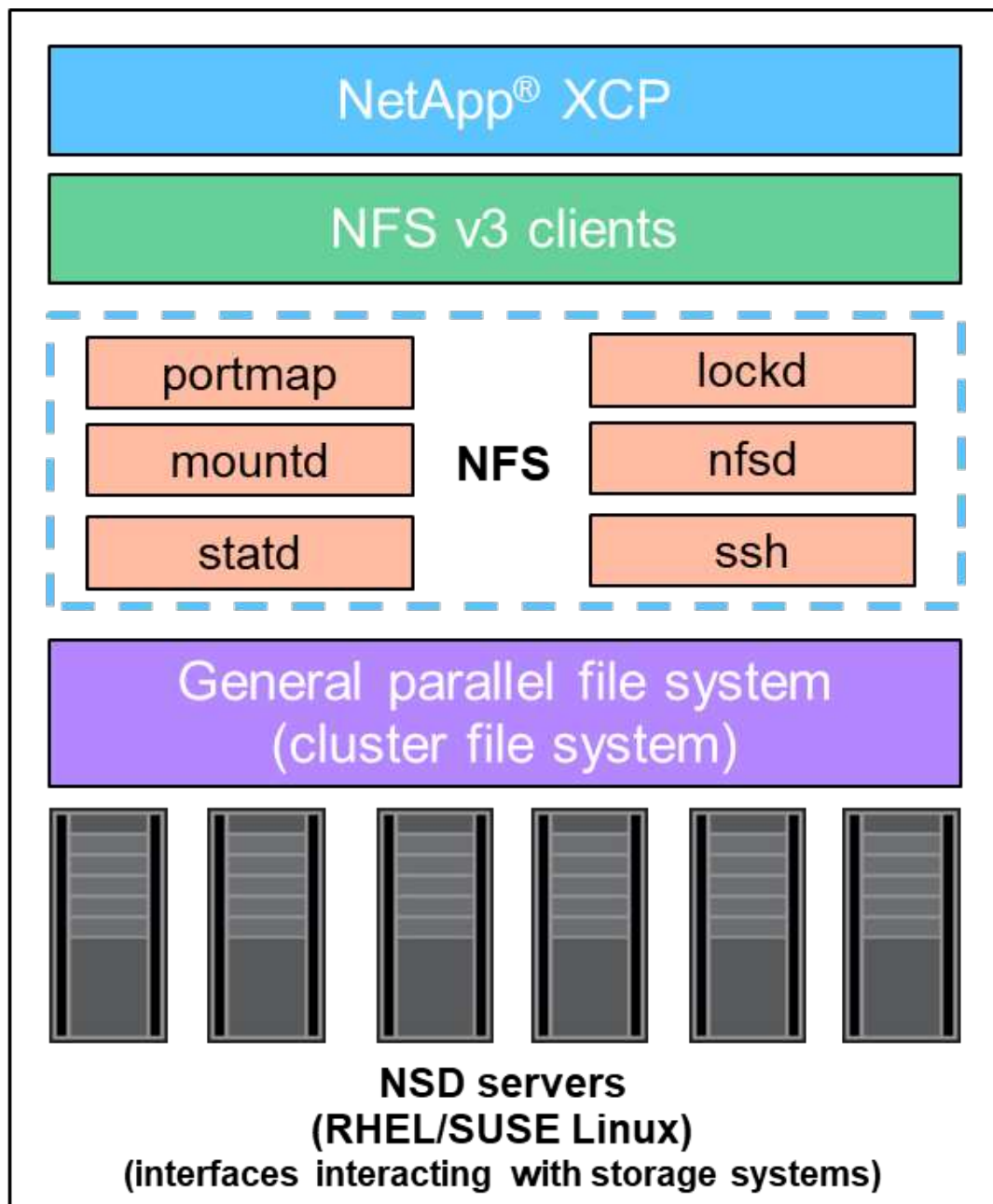
GPFS essentials

The following node types are used in GPFS:

- **Admin node.** Specifies an optional field containing a node name used by the administration commands to communicate between nodes. For example, the admin node `mastr-51.netapp.com` could pass a network check to all other nodes in the cluster.

- **Quorum node.** Determines whether a node is included in the pool of nodes from which quorum is derived. You need at least one node as a quorum node.

- **Manager Node.** Indicates whether a node is part of the node pool from which file system managers and token managers can be selected. It is a good idea to define more than one node as a manager node. How many nodes you designate as manager depends on the workload and the number of GPFS server licenses you have. If you are running large parallel jobs, you might need more manager nodes than in a four-node cluster supporting a web application.

- **NSD Server.** The server that prepares each physical disk for use with GPFS.

- **Protocol node.** The node that shares GPFS data directly through any Secure Shell (SSH) protocol with the NFS. This node requires a GPFS server license.

List of operations for GPFS, NFS, and XCP

This section provides the list of operations that create GPFS, export GPFS as an NFS export, and transfer the data by using XCP.

Create GPFS

To create GPFS, complete the following steps:

1. Download and install spectrum-scale data access for the Linux version on one of the servers.
2. Install the prerequisite package (chef for example) in all nodes and disable Security-Enhanced Linux (SELinux) in all nodes.
3. Set up the install node and add the admin node and the GPFS node to the cluster definition file.
4. Add the manager node, the quorum node, the NSD servers, and the GPFS node.
5. Add the GUI, admin, and GPFS nodes, and add an additional GUI server if required.
6. Add another GPFS node and check the list of all nodes.
7. Specify a cluster name, profile, remote shell binary, remote file copy binary, and port range to be set on all the GPFS nodes in the cluster definition file.
8. View the GPFS configuration settings and add an additional admin node.
9. Disable the data collection and upload the data package to the IBM Support Center.
10. Enable NTP and precheck the configurations before install.
11. Configure, create, and check the NSD disks.
12. Create the GPFS.
13. Mount the GPFS.
14. Verify and provide the required permissions to the GPFS.
15. Verify the GPFS read and write by running the `dd` command.

Export GPFS into NFS

To export the GPFS into NFS, complete the following steps:

1. Export GPFS as NFS through the `/etc/exports` file.

2. Install the required NFS server packages.

3. Start the NFS service.

4. List the files in the GPFS to validate the NFS client.

Configure NFS client

To configure the NFS client, complete the following steps:

1. Export the GPFS as NFS through the `/etc/exports` file.

2. Start the NFS client services.

3. Mount the GPFS through the NFS protocol on the NFS client.

4. Validate the list of GPFS files in the NFS mounted folder.

5. Move the data from GPFS exported NFS to NetApp NFS by using XCP.

6. Validate the GPFS files on the NFS client.

HDFS and MapR-FS to ONTAP NFS

For this solution, NetApp validated the migration of data from data lake (HDFS) and MapR cluster data to ONTAP NFS. The data resided in MapR-FS and HDFS. NetApp XCP introduced a new feature that directly migrates the data from a distributed file system such as HDFS and MapR-FS to ONTAP NFS. XCP uses async threads and HDFS C API calls to communicate and transfer data from MapR- FS as well as HDFS. The below figure shows the data migration from data lake (HDFS) and MapR-FS to ONTAP NFS. With this new feature, you don't have to export the source as an NFS share.
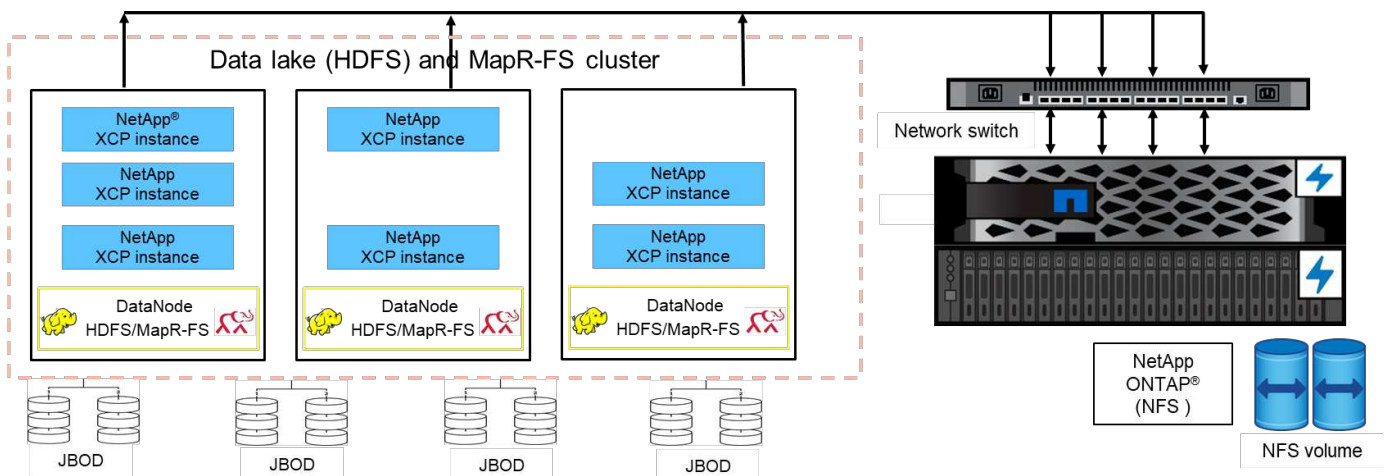
Why are customers moving from HDFS and MapR-FS to NFS?

Most of the Hadoop distributions such as Cloudera and Hortonworks use HDFS and MapR distributions uses their own filesystem called Mapr-FS to store data. HDFS and MapR-FS data provides the valuable insights to data scientists that can be leveraged in machine learning (ML) and deep learning (DL). The data in HDFS and MapR-FS is not shared, which means it cannot be used by other applications. Customers are looking for shared data, specifically in the banking sector where customers' sensitive data is used by multiple applications. The latest version of Hadoop (3.x or later) supports NFS data source, which can be accessed without additional third-party software. With the new NetApp XCP feature, data can be moved directly from HDFS and MapR-FS to NetApp NFS in order to provide access to multiple applications

Testing was done in Amazon Web Services (AWS) to transfer the data from MapR-FS to NFS for the initial performance test with 12 MAPR nodes and 4 NFS servers.

	Quantity	Size	vCPU	Memory	Storage	Network
NFS server	4	i3en.24xlarge	96	488GiB	8x 7500 NVMe SSD	100
MapR nodes	12	I3en.12xlarge	48	384GiB	4x 7500 NVMe SSD	50

Based on initial testing, we obtained 20GBps throughput and were able to transfer 2PB per day of data.

For more information about HDFS data migration without exporting HDFS to NFS, see the "Deployment steps - NAS" section in TR-4863: TR-4863: Best-Practice Guidelines for NetApp XCP - Data Mover, File Migration, and Analytics.

Next: Business benefits.

Business benefits

Previous: HDFS and MapR-FS to ONTAP NFS.

Moving data from big data analytics to AI provides the following benefits:

- The ability to extract data from different Hadoop file systems and GPFS into a unified NFS storage system
- A Hadoop-integrated and automated way to transfer data
- A reduction in the cost of library development for moving data from Hadoop file systems
- Maximum performance by aggregated throughput of multiple network interfaces from a single source of data by using NIPAM
- Scheduled and on-demand methods to transfer data
- Storage efficiency and enterprise management capability for unified NFS data by using ONTAP data management software
- Zero cost for data movement with the Hadoop method for data transfer

Next: GPFS to NFS-Detailed steps.

GPFS to NFS-Detailed steps

Previous: Business benefits.

This section provides the detailed steps needed to configure GPFS and move data into NFS by using NetApp XCP.

Configure GPFS

1. Download and Install Spectrum Scale Data Access for Linux on one of the servers.

```
[root@mastr-51 Spectrum_Scale_Data_Access-5.0.3.1-x86_64-Linux-
install_folder]# ls
Spectrum_Scale_Data_Access-5.0.3.1-x86_64-Linux-install
[root@mastr-51 Spectrum_Scale_Data_Access-5.0.3.1-x86_64-Linux-
install_folder]# chmod +x Spectrum_Scale_Data_Access-5.0.3.1-x86_64-
Linux-install
[root@mastr-51 Spectrum_Scale_Data_Access-5.0.3.1-x86_64-Linux-
install_folder]# ./Spectrum_Scale_Data_Access-5.0.3.1-x86_64-Linux-
install --manifest
manifest
…
<contents removes to save page space>
…
```

2. Install the prerequisite package (including chef and the kernel headers) on all nodes.

```
[root@mastr-51 5.0.3.1]# for i in 51 53 136 138 140  ; do ssh
10.63.150.$i "hostname; rpm -ivh  /gpfs_install/chef* "; done
mastr-51.netapp.com
warning: /gpfs_install/chef-13.6.4-1.el7.x86_64.rpm: Header V4 DSA/SHA1
Signature, key ID 83ef826a: NOKEY
Preparing...
#####################################
package chef-13.6.4-1.el7.x86_64 is already installed
mastr-53.netapp.com
warning: /gpfs_install/chef-13.6.4-1.el7.x86_64.rpm: Header V4 DSA/SHA1
Signature, key ID 83ef826a: NOKEY
Preparing...
#####################################
Updating / installing...
chef-13.6.4-1.el7
#####################################
Thank you for installing Chef!
workr-136.netapp.com
warning: /gpfs_install/chef-13.6.4-1.el7.x86_64.rpm: Header V4 DSA/SHA1
Signature, key ID 83ef826a: NOKEY
Preparing...
#####################################
Updating / installing...
```

```
chef-13.6.4-1.el7
#######################################
Thank you for installing Chef!
workr-138.netapp.com
warning: /gpfs_install/chef-13.6.4-1.el7.x86_64.rpm: Header V4 DSA/SHA1
Signature, key ID 83ef826a: NOKEY
Preparing...
#######################################
Updating / installing...
chef-13.6.4-1.el7
#######################################
Thank you for installing Chef!
workr-140.netapp.com
warning: /gpfs_install/chef-13.6.4-1.el7.x86_64.rpm: Header V4 DSA/SHA1
Signature, key ID 83ef826a: NOKEY
Preparing...
#######################################
Updating / installing...
chef-13.6.4-1.el7
#######################################
Thank you for installing Chef!
[root@mastr-51 5.0.3.1]#
[root@mastr-51 installer]# for i in 51 53 136 138 140  ; do ssh
10.63.150.$i "hostname; yumdownloader kernel-headers-3.10.0-
862.3.2.el7.x86_64 ; rpm -Uvh --oldpackage kernel-headers-3.10.0-
862.3.2.el7.x86_64.rpm"; done
mastr-51.netapp.com
Loaded plugins: priorities, product-id, subscription-manager
Preparing...
#######################################
Updating / installing...
kernel-headers-3.10.0-862.3.2.el7
#######################################
Cleaning up / removing...
kernel-headers-3.10.0-957.21.2.el7
#######################################
mastr-53.netapp.com
Loaded plugins: product-id, subscription-manager
Preparing...
#######################################
Updating / installing...
kernel-headers-3.10.0-862.3.2.el7
#######################################
Cleaning up / removing...
kernel-headers-3.10.0-862.11.6.el7
#######################################
```

```
workr-136.netapp.com
Loaded plugins: product-id, subscription-manager
Repository ambari-2.7.3.0 is listed more than once in the configuration
Preparing...
#####################################
Updating / installing...
kernel-headers-3.10.0-862.3.2.el7
#####################################
Cleaning up / removing...
kernel-headers-3.10.0-862.11.6.el7
#####################################
workr-138.netapp.com
Loaded plugins: product-id, subscription-manager
Preparing...
#####################################
package kernel-headers-3.10.0-862.3.2.el7.x86_64 is already installed
workr-140.netapp.com
Loaded plugins: product-id, subscription-manager
Preparing...
#####################################
Updating / installing...
kernel-headers-3.10.0-862.3.2.el7
#####################################
Cleaning up / removing...
kernel-headers-3.10.0-862.11.6.el7
#####################################
[root@mastr-51 installer]#
```

3. Disable SELinux in all nodes.

```
[root@mastr-51 5.0.3.1]# for i in 51 53 136 138 140  ; do ssh
10.63.150.$i "hostname; sudo setenforce 0"; done
mastr-51.netapp.com
setenforce: SELinux is disabled
mastr-53.netapp.com
setenforce: SELinux is disabled
workr-136.netapp.com
setenforce: SELinux is disabled
workr-138.netapp.com
setenforce: SELinux is disabled
workr-140.netapp.com
setenforce: SELinux is disabled
[root@mastr-51 5.0.3.1]#
```

4. Set up the install node.

```
[root@mastr-51 installer]# ./spectrumscale setup -s 10.63.150.51
[ INFO  ] Installing prerequisites for install node
[ INFO  ] Existing Chef installation detected. Ensure the PATH is
configured so that chef-client and knife commands can be run.
[ INFO  ] Your control node has been configured to use the IP
10.63.150.51 to communicate with other nodes.
[ INFO  ] Port 8889 will be used for chef communication.
[ INFO  ] Port 10080 will be used for package distribution.
[ INFO  ] Install Toolkit setup type is set to Spectrum Scale (default).
If an ESS is in the cluster, run this command to set ESS mode:
./spectrumscale setup -s server_ip -st ess
[ INFO  ] SUCCESS
[ INFO  ] Tip : Designate protocol, nsd and admin nodes in your
environment to use during install:./spectrumscale -v node add <node> -p
-a -n
[root@mastr-51 installer]#
```

5. Add the admin node and the GPFS node to the cluster definition file.

```
[root@mastr-51 installer]# ./spectrumscale node add mastr-51 -a
[ INFO  ] Adding node mastr-51.netapp.com as a GPFS node.
[ INFO  ] Setting mastr-51.netapp.com as an admin node.
[ INFO  ] Configuration updated.
[ INFO  ] Tip : Designate protocol or nsd nodes in your environment to
use during install:./spectrumscale node add <node> -p -n
[root@mastr-51 installer]#
```

6. Add the manager node and the GPFS node.

```
[root@mastr-51 installer]# ./spectrumscale node add mastr-53 -m
[ INFO  ] Adding node mastr-53.netapp.com as a GPFS node.
[ INFO  ] Adding node mastr-53.netapp.com as a manager node.
[root@mastr-51 installer]#
```

7. Add the quorum node and the GPFS node.

```
[root@mastr-51 installer]# ./spectrumscale node add workr-136 -q
[ INFO  ] Adding node workr-136.netapp.com as a GPFS node.
[ INFO  ] Adding node workr-136.netapp.com as a quorum node.
[root@mastr-51 installer]#
```

8. Add the NSD servers and the GPFS node.

```
[root@mastr-51 installer]# ./spectrumscale node add workr-138 -n
[ INFO  ] Adding node workr-138.netapp.com as a GPFS node.
[ INFO  ] Adding node workr-138.netapp.com as an NSD server.
[ INFO  ] Configuration updated.
[ INFO  ] Tip :If all node designations are complete, add NSDs to your
cluster definition and define required filessytems:./spectrumscale nsd
add <device> -p <primary node> -s <secondary node> -fs <file system>
[root@mastr-51 installer]#
```

9. Add the GUI, admin, and GPFS nodes.

```
[root@mastr-51 installer]# ./spectrumscale node add workr-136 -g
[ INFO  ] Setting workr-136.netapp.com as a GUI server.
[root@mastr-51 installer]# ./spectrumscale node add workr-136 -a
[ INFO  ] Setting workr-136.netapp.com as an admin node.
[ INFO  ] Configuration updated.
[ INFO  ] Tip : Designate protocol or nsd nodes in your environment to
use during install:./spectrumscale node add <node> -p -n
[root@mastr-51 installer]#
```

10. Add another GUI server.

```
[root@mastr-51 installer]# ./spectrumscale node add mastr-53 -g
[ INFO  ] Setting mastr-53.netapp.com as a GUI server.
[root@mastr-51 installer]#
```

11. Add another GPFS node.

```
[root@mastr-51 installer]# ./spectrumscale node add workr-140
[ INFO  ] Adding node workr-140.netapp.com as a GPFS node.
[root@mastr-51 installer]#
```

12. Verify and list all nodes.

```
[root@mastr-51 installer]# ./spectrumscale node list
[ INFO  ] List of nodes in current configuration:
[ INFO  ] [Installer Node]
[ INFO  ] 10.63.150.51
[ INFO  ]
[ INFO  ] [Cluster Details]
[ INFO  ] No cluster name configured
[ INFO  ] Setup Type: Spectrum Scale
[ INFO  ]
[ INFO  ] [Extended Features]
[ INFO  ] File Audit logging     : Disabled
[ INFO  ] Watch folder          : Disabled
[ INFO  ] Management GUI         : Enabled
[ INFO  ] Performance Monitoring : Disabled
[ INFO  ] Callhome              : Enabled
[ INFO  ]
[ INFO  ] GPFS                Admin   Quorum  Manager   NSD    Protocol
GUI   Callhome   OS   Arch
[ INFO  ] Node                         Node    Node      Node    Server   Node
Server   Server
[ INFO  ] mastr-51.netapp.com    X
rhel7  x86_64
[ INFO  ] mastr-53.netapp.com                       X
X            rhel7  x86_64
[ INFO  ] workr-136.netapp.com   X       X
X            rhel7  x86_64
[ INFO  ] workr-138.netapp.com                                     X
rhel7  x86_64
[ INFO  ] workr-140.netapp.com
rhel7  x86_64
[ INFO  ]
[ INFO  ] [Export IP address]
[ INFO  ] No export IP addresses configured
[root@mastr-51 installer]#
```

13. Specify a cluster name in the cluster definition file.

```
[root@mastr-51 installer]# ./spectrumscale config gpfs -c mastr-
51.netapp.com
[ INFO  ] Setting GPFS cluster name to mastr-51.netapp.com
[root@mastr-51 installer]#
```

14. Specify the profile.

```
[root@mastr-51 installer]# ./spectrumscale config gpfs -p default
[ INFO  ] Setting GPFS profile to default
[root@mastr-51 installer]#
 Profiles options: default [gpfsProtocolDefaults], random I/O
 [gpfsProtocolsRandomIO], sequential I/O [gpfsProtocolDefaults], random
 I/O [gpfsProtocolRandomIO]
```

15. Specify the remote shell binary to be used by GPFS; use `-r` argument.

```
[root@mastr-51 installer]# ./spectrumscale config gpfs -r /usr/bin/ssh
[ INFO  ] Setting Remote shell command to /usr/bin/ssh
[root@mastr-51 installer]#
```

16. Specify the remote file copy binary to be used by GPFS; use `-rc` argument.

```
[root@mastr-51 installer]# ./spectrumscale config gpfs -rc /usr/bin/scp
[ INFO  ] Setting Remote file copy command to /usr/bin/scp
[root@mastr-51 installer]#
```

17. Specify the port range to be set on all GPFS nodes; use `-e` argument.

```
[root@mastr-51 installer]# ./spectrumscale config gpfs -e 60000-65000
[ INFO  ] Setting GPFS Daemon communication port range to 60000-65000
[root@mastr-51 installer]#
```

18. View the GPFS config settings.

```
[root@mastr-51 installer]# ./spectrumscale config gpfs --list
[ INFO  ] Current settings are as follows:
[ INFO  ] GPFS cluster name is mastr-51.netapp.com.
[ INFO  ] GPFS profile is default.
[ INFO  ] Remote shell command is /usr/bin/ssh.
[ INFO  ] Remote file copy command is /usr/bin/scp.
[ INFO  ] GPFS Daemon communication port range is 60000-65000.
[root@mastr-51 installer]#
```

19. Add an admin node.

```
[root@mastr-51 installer]# ./spectrumscale node add 10.63.150.53 -a
[ INFO  ] Setting mastr-53.netapp.com as an admin node.
[ INFO  ] Configuration updated.
[ INFO  ] Tip : Designate protocol or nsd nodes in your environment to
use during install:./spectrumscale node add <node> -p -n
[root@mastr-51 installer]#
```

20. Disable the data collection and upload the data package to the IBM Support Center.

```
[root@mastr-51 installer]# ./spectrumscale callhome disable
[ INFO  ] Disabling the callhome.
[ INFO  ] Configuration updated.
[root@mastr-51 installer]#
```

21. Enable NTP.

```
[root@mastr-51 installer]# ./spectrumscale config ntp -e on
[root@mastr-51 installer]# ./spectrumscale config ntp -l
[ INFO  ] Current settings are as follows:
[ WARN  ] No value for Upstream NTP Servers(comma separated IP's with NO
space between multiple IPs) in clusterdefinition file.
[root@mastr-51 installer]# ./spectrumscale config ntp -s 10.63.150.51
[ WARN  ] The NTP package must already be installed and full
bidirectional access to the UDP port 123 must be allowed.
[ WARN  ] If NTP is already running on any of your nodes, NTP setup will
be skipped. To stop NTP run 'service ntpd stop'.
[ WARN  ] NTP is already on
[ INFO  ] Setting Upstream NTP Servers(comma separated IP's with NO
space between multiple IPs) to 10.63.150.51
[root@mastr-51 installer]# ./spectrumscale config ntp -e on
[ WARN  ] NTP is already on
[root@mastr-51 installer]# ./spectrumscale config ntp -l
[ INFO  ] Current settings are as follows:
[ INFO  ] Upstream NTP Servers(comma separated IP's with NO space
between multiple IPs) is 10.63.150.51.
[root@mastr-51 installer]#

[root@mastr-51 installer]# service ntpd start
Redirecting to /bin/systemctl start ntpd.service
[root@mastr-51 installer]# service ntpd status
Redirecting to /bin/systemctl status ntpd.service
● ntpd.service - Network Time Service
   Loaded: loaded (/usr/lib/systemd/system/ntpd.service; enabled; vendor
preset: disabled)
```

```
   Active: active (running) since Tue 2019-09-10 14:20:34 UTC; 1s ago
  Process: 2964 ExecStart=/usr/sbin/ntpd -u ntp:ntp $OPTIONS
(code=exited, status=0/SUCCESS)
 Main PID: 2965 (ntpd)
   CGroup: /system.slice/ntpd.service
           └─2965 /usr/sbin/ntpd -u ntp:ntp -g


Sep 10 14:20:34 mastr-51.netapp.com ntpd[2965]: ntp_io: estimated max
descriptors: 1024, initial socket boundary: 16
Sep 10 14:20:34 mastr-51.netapp.com ntpd[2965]: Listen and drop on 0
v4wildcard 0.0.0.0 UDP 123
Sep 10 14:20:34 mastr-51.netapp.com ntpd[2965]: Listen and drop on 1
v6wildcard :: UDP 123
Sep 10 14:20:34 mastr-51.netapp.com ntpd[2965]: Listen normally on 2 lo
127.0.0.1 UDP 123
Sep 10 14:20:34 mastr-51.netapp.com ntpd[2965]: Listen normally on 3
enp4s0f0 10.63.150.51 UDP 123
Sep 10 14:20:34 mastr-51.netapp.com ntpd[2965]: Listen normally on 4 lo
::1 UDP 123
Sep 10 14:20:34 mastr-51.netapp.com ntpd[2965]: Listen normally on 5
enp4s0f0 fe80::219:99ff:feef:99fa UDP 123
Sep 10 14:20:34 mastr-51.netapp.com ntpd[2965]: Listening on routing
socket on fd #22 for interface updates
Sep 10 14:20:34 mastr-51.netapp.com ntpd[2965]: 0.0.0.0 c016 06 restart
Sep 10 14:20:34 mastr-51.netapp.com ntpd[2965]: 0.0.0.0 c012 02 freq_set
kernel 11.890 PPM
[root@mastr-51 installer]#
```

22. Precheck the configurations before Install.

```
[root@mastr-51 installer]# ./spectrumscale install -pr
[ INFO  ] Logging to file: /usr/lpp/mmfs/5.0.3.1/installer/logs/INSTALL-
PRECHECK-10-09-2019_14:51:43.log
[ INFO  ] Validating configuration
[ INFO  ] Performing Chef (deploy tool) checks.
[ WARN  ] NTP is already running on: mastr-51.netapp.com. The install
toolkit will no longer setup NTP.
[ INFO  ] Node(s): ['workr-138.netapp.com'] were defined as NSD node(s)
but the toolkit has not been told about any NSDs served by these node(s)
nor has the toolkit been told to create new NSDs on these node(s). The
install will continue and these nodes will be assigned server licenses.
If NSDs are desired, either add them to the toolkit with
<./spectrumscale nsd add> followed by a <./spectrumscale install> or add
them manually afterwards using mmcrnsd.
[ INFO  ] Install toolkit will not configure file audit logging as it
has been disabled.
[ INFO  ] Install toolkit will not configure watch folder as it has been
disabled.
[ INFO  ] Checking for knife bootstrap configuration...
[ INFO  ] Performing GPFS checks.
[ INFO  ] Running environment checks
[ INFO  ] Skipping license validation as no existing GPFS cluster
detected.
[ INFO  ] Checking pre-requisites for portability layer.
[ INFO  ] GPFS precheck OK
[ INFO  ] Performing Performance Monitoring checks.
[ INFO  ] Running environment checks for Performance Monitoring
[ INFO  ] Performing GUI checks.
[ INFO  ] Performing FILE AUDIT LOGGING checks.
[ INFO  ] Running environment checks for file  Audit logging
[ INFO  ] Network check from admin node workr-136.netapp.com to all
other nodes in the cluster passed
[ INFO  ] Network check from admin node mastr-51.netapp.com to all other
nodes in the cluster passed
[ INFO  ] Network check from admin node mastr-53.netapp.com to all other
nodes in the cluster passed
[ INFO  ] The install toolkit will not configure call home as it is
disabled. To enable call home, use the following CLI command:
./spectrumscale callhome enable
[ INFO  ] Pre-check successful for install.
[ INFO  ] Tip : ./spectrumscale install
[root@mastr-51 installer]#
```

23. Configure the NSD disks.

```
[root@mastr-51 cluster-test]# cat disk.1st
%nsd: device=/dev/sdf
nsd=nsd1
servers=workr-136
usage=dataAndMetadata
failureGroup=1

%nsd: device=/dev/sdf
nsd=nsd2
servers=workr-138
usage=dataAndMetadata
failureGroup=1
```

24. Create the NSD disks.

```
[root@mastr-51 cluster-test]# mmcrnsd -F disk.1st -v no
mmcrnsd: Processing disk sdf
mmcrnsd: Processing disk sdf
mmcrnsd: Propagating the cluster configuration data to all
   affected nodes.  This is an asynchronous process.
[root@mastr-51 cluster-test]#
```

25. Check the NSD disk status.

```
[root@mastr-51 cluster-test]# mmlsnsd

 File system    Disk name    NSD servers
---------------------------------------------------------------------
---
  (free disk)   nsd1         workr-136.netapp.com
  (free disk)   nsd2         workr-138.netapp.com

[root@mastr-51 cluster-test]#
```

26. Create the GPFS.

```
[root@mastr-51 cluster-test]# mmcrfs gpfs1 -F disk.1st -B 1M -T /gpfs1

The following disks of gpfs1 will be formatted on node workr-
136.netapp.com:
    nsd1: size 3814912 MB
    nsd2: size 3814912 MB
Formatting file system ...
Disks up to size 33.12 TB can be added to storage pool system.
Creating Inode File
Creating Allocation Maps
Creating Log Files
Clearing Inode Allocation Map
Clearing Block Allocation Map
Formatting Allocation Map for storage pool system
Completed creation of file system /dev/gpfs1.
mmcrfs: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
[root@mastr-51 cluster-test]#
```

27. Mount the GPFS.

```
[root@mastr-51 cluster-test]# mmmount all -a
Tue Oct  8 18:05:34 UTC 2019: mmmount: Mounting file systems ...
[root@mastr-51 cluster-test]#
```

28. Check and provide the required permissions to the GPFS.

```
[root@mastr-51 cluster-test]# mmlsdisk gpfs1
disk         driver    sector     failure holds     holds
storage
name         type      size       group metadata data   status
availability pool
------------ -------- ------ ----------- -------- ----- -------------
------------ ------------
nsd1         nsd          512           1 Yes       Yes   ready           up
system
nsd2         nsd          512           1 Yes       Yes   ready           up
system
[root@mastr-51 cluster-test]#

[root@mastr-51 cluster-test]# for i in 51 53 136 138  ; do ssh
10.63.150.$i "hostname; chmod 777 /gpfs1" ; done;
mastr-51.netapp.com
mastr-53.netapp.com
workr-136.netapp.com
workr-138.netapp.com
[root@mastr-51 cluster-test]#
```

29. Check the GPFS read and write by running the `dd` command.

```
[root@mastr-51 cluster-test]# dd if=/dev/zero of=/gpfs1/testfile
bs=1024M count=5
5+0 records in
5+0 records out
5368709120 bytes (5.4 GB) copied, 8.3981 s, 639 MB/s
[root@mastr-51 cluster-test]# for i in 51 53 136 138  ; do ssh
10.63.150.$i "hostname; ls -ltrh /gpfs1" ; done;
mastr-51.netapp.com
total 5.0G
-rw-r--r-- 1 root root 5.0G Oct  8 18:10 testfile
mastr-53.netapp.com
total 5.0G
-rw-r--r-- 1 root root 5.0G Oct  8 18:10 testfile
workr-136.netapp.com
total 5.0G
-rw-r--r-- 1 root root 5.0G Oct  8 18:10 testfile
workr-138.netapp.com
total 5.0G
-rw-r--r-- 1 root root 5.0G Oct  8 18:10 testfile
[root@mastr-51 cluster-test]#
```

Export GPFS into NFS

To export GPFS into NFS, complete the following steps:

1. Export the GPFS as NFS through the `/etc/exports` file.

```
[root@mastr-51 gpfs1]# cat /etc/exports
/gpfs1          *(rw,fsid=745)
[root@mastr-51 gpfs1]
```

2. Install the required NFS server packages.

```
[root@mastr-51 ~]# yum install rpcbind
Loaded plugins: priorities, product-id, search-disabled-repos,
subscription-manager
Resolving Dependencies
--> Running transaction check
---> Package rpcbind.x86_64 0:0.2.0-47.el7 will be updated
---> Package rpcbind.x86_64 0:0.2.0-48.el7 will be an update
--> Finished Dependency Resolution

Dependencies Resolved


========================================================================
========================================================================
========================================================================
=====================
 Package                                             Arch
Version                                           Repository
Size
========================================================================
========================================================================
========================================================================
=====================
Updating:
 rpcbind                                             x86_64
0.2.0-48.el7                                          rhel-7-
server-rpms                                           60 k

Transaction Summary
========================================================================
========================================================================
========================================================================
=====================
Upgrade  1 Package
```

```
Total download size: 60 k
Is this ok [y/d/N]: y
Downloading packages:
No Presto metadata available for rhel-7-server-rpms
rpcbind-0.2.0-48.el7.x86_64.rpm
|  60 kB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Updating   : rpcbind-0.2.0-48.el7.x86_64
1/2
  Cleanup    : rpcbind-0.2.0-47.el7.x86_64
2/2
  Verifying  : rpcbind-0.2.0-48.el7.x86_64
1/2
  Verifying  : rpcbind-0.2.0-47.el7.x86_64
2/2

Updated:
  rpcbind.x86_64 0:0.2.0-48.el7

Complete!
[root@mastr-51 ~]#
```

3. Start the NFS service.

```
[root@mastr-51 ~]# service nfs status
Redirecting to /bin/systemctl status nfs.service
● nfs-server.service - NFS server and services
   Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; disabled;
vendor preset: disabled)
  Drop-In: /run/systemd/generator/nfs-server.service.d
           └─order-with-mounts.conf
   Active: inactive (dead)
[root@mastr-51 ~]# service rpcbind start
Redirecting to /bin/systemctl start rpcbind.service
[root@mastr-51 ~]# service nfs start
Redirecting to /bin/systemctl start nfs.service
[root@mastr-51 ~]# service nfs status
Redirecting to /bin/systemctl status nfs.service
● nfs-server.service - NFS server and services
   Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; disabled;
vendor preset: disabled)
  Drop-In: /run/systemd/generator/nfs-server.service.d
           └─order-with-mounts.conf
   Active: active (exited) since Wed 2019-11-06 16:34:50 UTC; 2s ago
  Process: 24402 ExecStartPost=/bin/sh -c if systemctl -q is-active
gssproxy; then systemctl reload gssproxy ; fi (code=exited,
status=0/SUCCESS)
  Process: 24383 ExecStart=/usr/sbin/rpc.nfsd $RPCNFSDARGS (code=exited,
status=0/SUCCESS)
  Process: 24379 ExecStartPre=/usr/sbin/exportfs -r (code=exited,
status=0/SUCCESS)
 Main PID: 24383 (code=exited, status=0/SUCCESS)
   CGroup: /system.slice/nfs-server.service

Nov 06 16:34:50 mastr-51.netapp.com systemd[1]: Starting NFS server and
services...
Nov 06 16:34:50 mastr-51.netapp.com systemd[1]: Started NFS server and
services.
[root@mastr-51 ~]#
```

4. List the files in GPFS to validate the NFS client.

```
[root@mastr-51 gpfs1]# df -Th
Filesystem                             Type       Size  Used Avail
Use% Mounted on
/dev/mapper/rhel_stlrx300s6--22--irmc-root xfs      94G   55G   39G
59% /
devtmpfs                               devtmpfs    32G     0   32G
0% /dev
tmpfs                                  tmpfs       32G     0   32G
0% /dev/shm
tmpfs                                  tmpfs       32G  3.3G   29G
11% /run
tmpfs                                  tmpfs       32G     0   32G
0% /sys/fs/cgroup
/dev/sda7                              xfs        9.4G  210M  9.1G
3% /boot
tmpfs                                  tmpfs      6.3G     0  6.3G
0% /run/user/10065
tmpfs                                  tmpfs      6.3G     0  6.3G
0% /run/user/10068
tmpfs                                  tmpfs      6.3G     0  6.3G
0% /run/user/10069
10.63.150.213:/nc_volume3              nfs4       380G  8.0M  380G
1% /mnt
tmpfs                                  tmpfs      6.3G     0  6.3G
0% /run/user/0
gpfs1                                  gpfs       7.3T  9.1G  7.3T
1% /gpfs1
[root@mastr-51 gpfs1]#
[root@mastr-51 ~]# cd /gpfs1
[root@mastr-51 gpfs1]# ls
catalog  ces  gpfs-ces  ha  testfile
[root@mastr-51 gpfs1]#
[root@mastr-51 ~]# cd /gpfs1
[root@mastr-51 gpfs1]# ls
ces  gpfs-ces  ha  testfile
[root@mastr-51 gpfs1]# ls -ltrha
total 5.1G
dr-xr-xr-x   2 root root 8.0K Jan  1  1970 .snapshots
-rw-r--r--   1 root root 5.0G Oct  8 18:10 testfile
dr-xr-xr-x. 30 root root 4.0K Oct  8 18:19 ..
drwxr-xr-x   2 root root 4.0K Nov  5 20:02 gpfs-ces
drwxr-xr-x   2 root root 4.0K Nov  5 20:04 ha
drwxrwxrwx   5 root root 256K Nov  5 20:04 .
drwxr-xr-x   4 root root 4.0K Nov  5 20:35 ces
[root@mastr-51 gpfs1]#
```

Configure the NFS client

To configure the NFS client, complete the following steps:

1. Install packages in the NFS client.

```
[root@hdp2 ~]# yum install nfs-utils rpcbind
Loaded plugins: product-id, search-disabled-repos, subscription-manager
HDP-2.6-GPL-repo-4
| 2.9 kB  00:00:00
HDP-2.6-repo-4
| 2.9 kB  00:00:00
HDP-3.0-GPL-repo-2
| 2.9 kB  00:00:00
HDP-3.0-repo-2
| 2.9 kB  00:00:00
HDP-3.0-repo-3
| 2.9 kB  00:00:00
HDP-3.1-repo-1
| 2.9 kB  00:00:00
HDP-3.1-repo-51
| 2.9 kB  00:00:00
HDP-UTILS-1.1.0.22-repo-1
| 2.9 kB  00:00:00
HDP-UTILS-1.1.0.22-repo-2
| 2.9 kB  00:00:00
HDP-UTILS-1.1.0.22-repo-3
| 2.9 kB  00:00:00
HDP-UTILS-1.1.0.22-repo-4
| 2.9 kB  00:00:00
HDP-UTILS-1.1.0.22-repo-51
| 2.9 kB  00:00:00
ambari-2.7.3.0
| 2.9 kB  00:00:00
epel/x86_64/metalink
|   13 kB  00:00:00
epel
| 5.3 kB  00:00:00
mysql-connectors-community
| 2.5 kB  00:00:00
mysql-tools-community
| 2.5 kB  00:00:00
mysql56-community
| 2.5 kB  00:00:00
rhel-7-server-optional-rpms
| 3.2 kB  00:00:00
rhel-7-server-rpms
```

```
|  3.5 kB  00:00:00
(1/10): mysql-connectors-community/x86_64/primary_db
|  49 kB  00:00:00
(2/10): mysql-tools-community/x86_64/primary_db
|  66 kB  00:00:00
(3/10): epel/x86_64/group_gz
|  90 kB  00:00:00
(4/10): mysql56-community/x86_64/primary_db
| 241 kB  00:00:00
(5/10): rhel-7-server-optional-rpms/7Server/x86_64/updateinfo
| 2.5 MB  00:00:00
(6/10): rhel-7-server-rpms/7Server/x86_64/updateinfo
| 3.4 MB  00:00:00
(7/10): rhel-7-server-optional-rpms/7Server/x86_64/primary_db
| 8.3 MB  00:00:00
(8/10): rhel-7-server-rpms/7Server/x86_64/primary_db
|  62 MB  00:00:01
(9/10): epel/x86_64/primary_db
| 6.9 MB  00:00:08
(10/10): epel/x86_64/updateinfo
| 1.0 MB  00:00:13
Resolving Dependencies
--> Running transaction check
---> Package nfs-utils.x86_64 1:1.3.0-0.61.el7 will be updated
---> Package nfs-utils.x86_64 1:1.3.0-0.65.el7 will be an update
---> Package rpcbind.x86_64 0:0.2.0-47.el7 will be updated
---> Package rpcbind.x86_64 0:0.2.0-48.el7 will be an update
--> Finished Dependency Resolution


Dependencies Resolved


================================================================================
================================================
 Package                 Arch                  Version
Repository                      Size
================================================================================
================================================
Updating:
 nfs-utils               x86_64                1:1.3.0-0.65.el7
rhel-7-server-rpms               412 k
 rpcbind                 x86_64                0.2.0-48.el7
rhel-7-server-rpms                60 k

Transaction Summary
================================================================================
================================================
```

```
Upgrade  2 Packages


Total download size: 472 k
Is this ok [y/d/N]: y
Downloading packages:
No Presto metadata available for rhel-7-server-rpms
(1/2): rpcbind-0.2.0-48.el7.x86_64.rpm
|  60 kB  00:00:00
(2/2): nfs-utils-1.3.0-0.65.el7.x86_64.rpm
| 412 kB  00:00:00
--------------------------------------------------------------------------
-------------------------------------------------
Total
1.2 MB/s | 472 kB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Updating   : rpcbind-0.2.0-48.el7.x86_64
1/4
service rpcbind start

  Updating   : 1:nfs-utils-1.3.0-0.65.el7.x86_64
2/4
  Cleanup    : 1:nfs-utils-1.3.0-0.61.el7.x86_64
3/4
  Cleanup    : rpcbind-0.2.0-47.el7.x86_64
4/4
  Verifying  : 1:nfs-utils-1.3.0-0.65.el7.x86_64
1/4
  Verifying  : rpcbind-0.2.0-48.el7.x86_64
2/4
  Verifying  : rpcbind-0.2.0-47.el7.x86_64
3/4
  Verifying  : 1:nfs-utils-1.3.0-0.61.el7.x86_64
4/4

Updated:
  nfs-utils.x86_64 1:1.3.0-0.65.el7
rpcbind.x86_64 0:0.2.0-48.el7

Complete!
[root@hdp2 ~]#
```

2. Start the NFS client services.

```
[root@hdp2 ~]# service rpcbind start
Redirecting to /bin/systemctl start rpcbind.service
 [root@hdp2 ~]#
```

3. Mount the GPFS through the NFS protocol on the NFS client.

```
[root@hdp2 ~]# mkdir /gpfstest
[root@hdp2 ~]# mount 10.63.150.51:/gpfs1 /gpfstest
[root@hdp2 ~]# df -h
Filesystem                             Size  Used Avail Use% Mounted on
/dev/mapper/rhel_stlrx300s6--22-root   1.1T  113G  981G  11% /
devtmpfs                               126G     0  126G   0% /dev
tmpfs                                  126G   16K  126G   1% /dev/shm
tmpfs                                  126G  510M  126G   1% /run
tmpfs                                  126G     0  126G   0%
/sys/fs/cgroup
/dev/sdd2                              197M  191M  6.6M  97% /boot
tmpfs                                   26G     0   26G   0% /run/user/0
10.63.150.213:/nc_volume2              95G  5.4G   90G   6% /mnt
10.63.150.51:/gpfs1                    7.3T  9.1G  7.3T   1% /gpfstest
[root@hdp2 ~]#
```

4. Validate the list of GPFS files in the NFS-mounted folder.

```
[root@hdp2 ~]# cd /gpfstest/
[root@hdp2 gpfstest]# ls
ces   gpfs-ces   ha   testfile
[root@hdp2 gpfstest]# ls -l
total 5242882
drwxr-xr-x 4 root root       4096 Nov  5 15:35 ces
drwxr-xr-x 2 root root       4096 Nov  5 15:02 gpfs-ces
drwxr-xr-x 2 root root       4096 Nov  5 15:04 ha
-rw-r--r-- 1 root root 5368709120 Oct  8 14:10 testfile
[root@hdp2 gpfstest]#
```

5. Move the data from the GPFS- exported NFS to the NetApp NFS by using XCP.

```
[root@hdp2 linux]# ./xcp copy -parallel 20 10.63.150.51:/gpfs1
10.63.150.213:/nc_volume2/
XCP 1.4-17914d6; (c) 2019 NetApp, Inc.; Licensed to Karthikeyan
Nagalingam [NetApp Inc] until Tue Nov  5 12:39:36 2019

xcp: WARNING: your license will expire in less than one week! You can
renew your license at https://xcp.netapp.com
xcp: open or create catalog 'xcp': Creating new catalog in
'10.63.150.51:/gpfs1/catalog'
xcp: WARNING: No index name has been specified, creating one with name:
autoname_copy_2019-11-11_12.14.07.805223
xcp: mount '10.63.150.51:/gpfs1': WARNING: This NFS server only supports
1-second timestamp granularity. This may cause sync to fail because
changes will often be undetectable.
 34 scanned, 32 copied, 32 indexed, 1 giant, 301 MiB in (59.5 MiB/s),
784 KiB out (155 KiB/s), 6s
 34 scanned, 32 copied, 32 indexed, 1 giant, 725 MiB in (84.6 MiB/s),
1.77 MiB out (206 KiB/s), 11s
 34 scanned, 32 copied, 32 indexed, 1 giant, 1.17 GiB in (94.2 MiB/s),
2.90 MiB out (229 KiB/s), 16s
 34 scanned, 32 copied, 32 indexed, 1 giant, 1.56 GiB in (79.8 MiB/s),
3.85 MiB out (194 KiB/s), 21s
 34 scanned, 32 copied, 32 indexed, 1 giant, 1.95 GiB in (78.4 MiB/s),
4.80 MiB out (191 KiB/s), 26s
 34 scanned, 32 copied, 32 indexed, 1 giant, 2.35 GiB in (80.4 MiB/s),
5.77 MiB out (196 KiB/s), 31s
 34 scanned, 32 copied, 32 indexed, 1 giant, 2.79 GiB in (89.6 MiB/s),
6.84 MiB out (218 KiB/s), 36s
 34 scanned, 32 copied, 32 indexed, 1 giant, 3.16 GiB in (75.3 MiB/s),
7.73 MiB out (183 KiB/s), 41s
 34 scanned, 32 copied, 32 indexed, 1 giant, 3.53 GiB in (75.4 MiB/s),
8.64 MiB out (183 KiB/s), 46s
 34 scanned, 32 copied, 32 indexed, 1 giant, 4.00 GiB in (94.4 MiB/s),
9.77 MiB out (230 KiB/s), 51s
 34 scanned, 32 copied, 32 indexed, 1 giant, 4.46 GiB in (94.3 MiB/s),
10.9 MiB out (229 KiB/s), 56s
 34 scanned, 32 copied, 32 indexed, 1 giant, 4.86 GiB in (80.2 MiB/s),
11.9 MiB out (195 KiB/s), 1m1s
Sending statistics...
34 scanned, 33 copied, 34 indexed, 1 giant, 5.01 GiB in (81.8 MiB/s),
12.3 MiB out (201 KiB/s), 1m2s.
[root@hdp2 linux]#
```

6. Validate the GPFS files on the NFS client.

```
[root@hdp2 mnt]# df -Th
Filesystem                           Type      Size  Used Avail Use%
Mounted on
/dev/mapper/rhel_stlrx300s6--22-root xfs       1.1T  113G  981G  11% /
devtmpfs                             devtmpfs  126G     0  126G   0%
/dev
tmpfs                                tmpfs     126G   16K  126G   1%
/dev/shm
tmpfs                                tmpfs     126G  518M  126G   1%
/run
tmpfs                                tmpfs     126G     0  126G   0%
/sys/fs/cgroup
/dev/sdd2                            xfs       197M  191M  6.6M  97%
/boot
tmpfs                                tmpfs      26G     0   26G   0%
/run/user/0
10.63.150.213:/nc_volume2            nfs4       95G  5.4G   90G   6%
/mnt
10.63.150.51:/gpfs1                  nfs4      7.3T  9.1G  7.3T   1%
/gpfstest
[root@hdp2 mnt]#
[root@hdp2 mnt]# ls -ltrha
total 128K
dr-xr-xr-x   2 root       root          4.0K Dec 31  1969
.snapshots
drwxrwxrwx   2 root       root          4.0K Feb 14  2018 data
drwxrwxrwx   3 root       root          4.0K Feb 14  2018
wcresult
drwxrwxrwx   3 root       root          4.0K Feb 14  2018
wcresult1
drwxrwxrwx   2 root       root          4.0K Feb 14  2018
wcresult2
drwxrwxrwx   2 root       root          4.0K Feb 16  2018
wcresult3
-rw-r--r--   1 root       root          2.8K Feb 20  2018
READMEdemo
drwxrwxrwx   3 root       root          4.0K Jun 28 13:38 scantg
drwxrwxrwx   3 root       root          4.0K Jun 28 13:39
scancopyFromLocal
-rw-r--r--   1 hdfs       hadoop        1.2K Jul  3 19:28 f3
-rw-r--r--   1 hdfs       hadoop        1.2K Jul  3 19:28 README
-rw-r--r--   1 hdfs       hadoop        1.2K Jul  3 19:28 f9
-rw-r--r--   1 hdfs       hadoop        1.2K Jul  3 19:28 f6
-rw-r--r--   1 hdfs       hadoop        1.2K Jul  3 19:28 f5
-rw-r--r--   1 hdfs       hadoop        1.2K Jul  3 19:30 f4
-rw-r--r--   1 hdfs       hadoop        1.2K Jul  3 19:30 f8
```

```
-rw-r--r--    1 hdfs          hadoop                  1.2K Jul  3 19:30 f2
-rw-r--r--    1 hdfs          hadoop                  1.2K Jul  3 19:30 f7
drwxrwxrwx    2 root          root                    4.0K Jul  9 11:14 test
drwxrwxrwx    3 root          root                    4.0K Jul 10 16:35
warehouse
drwxr-xr-x    3        10061 tester1                  4.0K Jul 15 14:40 sdd1
drwxrwxrwx    3 testeruser1 hadoopkerberosgroup 4.0K Aug 20 17:00
kermkdir
-rw-r--r--    1 testeruser1 hadoopkerberosgroup    0 Aug 21 14:20 newfile
drwxrwxrwx    2 testeruser1 hadoopkerberosgroup 4.0K Aug 22 10:13
teragen1copy_3
drwxrwxrwx    2 testeruser1 hadoopkerberosgroup 4.0K Aug 22 10:33
teragen2copy_1
-rw-rwxr--    1 root          hdfs                    1.2K Sep 19 16:38 R1
drwx------    3 root          root                    4.0K Sep 20 17:28 user
-rw-r--r--    1 root          root                    5.0G Oct  8 14:10
testfile
drwxr-xr-x    2 root          root                    4.0K Nov  5 15:02 gpfs-
ces
drwxr-xr-x    2 root          root                    4.0K Nov  5 15:04 ha
drwxr-xr-x    4 root          root                    4.0K Nov  5 15:35 ces
dr-xr-xr-x. 26 root          root                    4.0K Nov  6 11:40 ..
drwxrwxrwx   21 root          root                    4.0K Nov 11 12:14 .
drwxrwxrwx    7 nobody        nobody                  4.0K Nov 11 12:14 catalog
[root@hdp2 mnt]#
```

Next: MapR-FS to ONTAP NFS.

MapR-FS to ONTAP NFS

Previous: GPFS to NFS - Detailed steps.

This section provides the detailed steps needed to move MapR-FS data into ONTAP NFS by using NetApp XCP.

1. Provision three LUNs for each MapR node and give the LUNs ownership of all MapR nodes.

2. During installation, choose newly added LUNs for MapR cluster disks that are used for MapR-FS.

3. Install a MapR cluster according to the MapR 6.1 documentation.

4. Check the basic Hadoop operations using MapReduce commands such as `hadoop jar xxx`.

5. Keep customer data in MapR-FS. For example, we generated approximately a terabyte of sample data in MapR-FS by using Teragen.

6. Configure MapR-FS as NFS export.

   a. Disable the nlockmgr service on all MapR nodes.

```
root@workr-138: ~$ rpcinfo -p
   program vers proto    port  service
    100000    4   tcp    111  portmapper
    100000    3   tcp    111  portmapper
    100000    2   tcp    111  portmapper
    100000    4   udp    111  portmapper
    100000    3   udp    111  portmapper
    100000    2   udp    111  portmapper
    100003    4   tcp   2049  nfs
    100227    3   tcp   2049  nfs_acl
    100003    4   udp   2049  nfs
    100227    3   udp   2049  nfs_acl
    100021    3   udp  55270  nlockmgr
    100021    4   udp  55270  nlockmgr
    100021    3   tcp  35025  nlockmgr
    100021    4   tcp  35025  nlockmgr
    100003    3   tcp   2049  nfs
    100005    3   tcp   2049  mountd
    100005    1   tcp   2049  mountd
    100005    3   udp   2049  mountd
    100005    1   udp   2049  mountd
root@workr-138: ~$

root@workr-138: ~$ rpcinfo -d 100021 3
root@workr-138: ~$ rpcinfo -d 100021 4
```

b. Export specific folders from MapR-FS on all MapR nodes in the `/opt/mapr/conf/exports` file. Do not export the parent folder with different permissions when you export sub folders.

```
[mapr@workr-138 ~]$ cat /opt/mapr/conf/exports
# Sample Exports file
# for /mapr exports
# <Path> <exports_control>
#access_control -> order is specific to default
# list the hosts before specifying a default for all
#  a.b.c.d,1.2.3.4(ro) d.e.f.g(ro) (rw)
#  enforces ro for a.b.c.d & 1.2.3.4 and everybody else is rw
# special path to export clusters in mapr-clusters.conf. To disable
exporting,
# comment it out. to restrict access use the exports_control
#
#/mapr (rw)
#karthik
/mapr/my.cluster.com/tmp/testnfs /maprnfs3 (rw)
#to export only certain clusters, comment out the /mapr & uncomment.
#/mapr/clustername (rw)
#to export /mapr only to certain hosts (using exports_control)
#/mapr a.b.c.d(rw),e.f.g.h(ro)
# export /mapr/cluster1 rw to a.b.c.d & ro to e.f.g.h (denied for
others)
#/mapr/cluster1 a.b.c.d(rw),e.f.g.h(ro)
# export /mapr/cluster2 only to e.f.g.h (denied for others)
#/mapr/cluster2 e.f.g.h(rw)
# export /mapr/cluster3 rw to e.f.g.h & ro to others
#/mapr/cluster2 e.f.g.h(rw) (ro)
#to export a certain cluster, volume or a subdirectory as an alias,
#comment out  /mapr & uncomment
#/mapr/clustername         /alias1 (rw)
#/mapr/clustername/vol     /alias2 (rw)
#/mapr/clustername/vol/dir /alias3 (rw)
#only the alias will be visible/exposed to the nfs client not the
mapr path, host options as before
[mapr@workr-138 ~]$
```

7. Refresh the MapR-FS NFS service.

```
root@workr-138: tmp$ maprcli nfsmgmt refreshexports
ERROR (22) -  You do not have a ticket to communicate with
127.0.0.1:9998. Retry after obtaining a new ticket using maprlogin
root@workr-138: tmp$ su - mapr
[mapr@workr-138 ~]$ maprlogin password -cluster my.cluster.com
[Password for user 'mapr' at cluster 'my.cluster.com': ]
MapR credentials of user 'mapr' for cluster 'my.cluster.com' are written
to '/tmp/maprticket_5000'
[mapr@workr-138 ~]$ maprcli nfsmgmt refreshexports
```

8. Assign a virtual IP range to a specific server or a set of servers in the MapR cluster. Then the MapR cluster assigns an IP to a specific server for NFS data access. The IPs enable high availability, which means that, if a server or network with a particular IP experiences failure, the next IP from the range of IPs can be used for NFS access.

> (i) If you would like to provide NFS access from all MapR nodes, then you can assign a set of virtual IPs to each server, and you can use the resources from each MapR node for NFS data access.

9. Check the virtual IPs assigned on each MapR node and use them for NFS data access.

```
root@workr-138: ~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
```

```
       valid_lft forever preferred_lft forever
2: ens3f0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc mq state UP
group default qlen 1000
    link/ether 90:1b:0e:d1:5d:f9 brd ff:ff:ff:ff:ff:ff
    inet 10.63.150.138/24 brd 10.63.150.255 scope global noprefixroute
ens3f0
       valid_lft forever preferred_lft forever
    inet 10.63.150.96/24 scope global secondary ens3f0:~m0
       valid_lft forever preferred_lft forever
    inet 10.63.150.97/24 scope global secondary ens3f0:~m1
       valid_lft forever preferred_lft forever
    inet6 fe80::921b:eff:fed1:5df9/64 scope link
       valid_lft forever preferred_lft forever
3: eno1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
group default qlen 1000
    link/ether 90:1b:0e:d1:af:b4 brd ff:ff:ff:ff:ff:ff
4: ens3f1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
group default qlen 1000
    link/ether 90:1b:0e:d1:5d:fa brd ff:ff:ff:ff:ff:ff
5: eno2: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state
DOWN group default qlen 1000
    link/ether 90:1b:0e:d1:af:b5 brd ff:ff:ff:ff:ff:ff
[root@workr-138: ~$
[root@workr-140 ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens3f0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc mq state UP
group default qlen 1000
    link/ether 90:1b:0e:d1:5e:03 brd ff:ff:ff:ff:ff:ff
    inet 10.63.150.140/24 brd 10.63.150.255 scope global noprefixroute
ens3f0
       valid_lft forever preferred_lft forever
    inet 10.63.150.92/24 scope global secondary ens3f0:~m0
       valid_lft forever preferred_lft forever
    inet6 fe80::921b:eff:fed1:5e03/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: eno1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
group default qlen 1000
    link/ether 90:1b:0e:d1:af:9a brd ff:ff:ff:ff:ff:ff
4: ens3f1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
group default qlen 1000
```

```
     link/ether 90:1b:0e:d1:5e:04 brd ff:ff:ff:ff:ff:ff
5: eno2: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state
DOWN group default qlen 1000
     link/ether 90:1b:0e:d1:af:9b brd ff:ff:ff:ff:ff:ff
[root@workr-140 ~]#
```

10. Mount the NFS- exported MapR-FS using the assigned virtual IP for checking the NFS operation. However, this step is not required for data transfer using NetApp XCP.

```
root@workr-138: tmp$ mount -v -t nfs 10.63.150.92:/maprnfs3
/tmp/testmount/
mount.nfs: timeout set for Thu Dec  5 15:31:32 2019
mount.nfs: trying text-based options
 'vers=4.1,addr=10.63.150.92,clientaddr=10.63.150.138'
mount.nfs: mount(2): Protocol not supported
mount.nfs: trying text-based options
 'vers=4.0,addr=10.63.150.92,clientaddr=10.63.150.138'
mount.nfs: mount(2): Protocol not supported
mount.nfs: trying text-based options 'addr=10.63.150.92'
mount.nfs: prog 100003, trying vers=3, prot=6
mount.nfs: trying 10.63.150.92 prog 100003 vers 3 prot TCP port 2049
mount.nfs: prog 100005, trying vers=3, prot=17
mount.nfs: trying 10.63.150.92 prog 100005 vers 3 prot UDP port 2049
mount.nfs: portmap query retrying: RPC: Timed out
mount.nfs: prog 100005, trying vers=3, prot=6
mount.nfs: trying 10.63.150.92 prog 100005 vers 3 prot TCP port 2049
root@workr-138: tmp$ df -h
Filesystem             Size  Used Avail Use% Mounted on
/dev/sda7               84G   48G   37G  57% /
devtmpfs               126G     0  126G   0% /dev
tmpfs                  126G     0  126G   0% /dev/shm
tmpfs                  126G   19M  126G   1% /run
tmpfs                  126G     0  126G   0% /sys/fs/cgroup
/dev/sdd1              3.7T  201G  3.5T   6% /mnt/sdd1
/dev/sda6              946M  220M  726M  24% /boot
tmpfs                   26G     0   26G   0% /run/user/5000
gpfs1                  7.3T  9.1G  7.3T   1% /gpfs1
tmpfs                   26G     0   26G   0% /run/user/0
localhost:/mapr        100G     0  100G   0% /mapr
10.63.150.92:/maprnfs3  53T  8.4G   53T   1% /tmp/testmount
root@workr-138: tmp$
```

11. Configure NetApp XCP to transfer data from the MapR-FS NFS gateway to ONTAP NFS.

   a. Configure the catalog location for XCP.

```
[root@hdp2 linux]# cat /opt/NetApp/xFiles/xcp/xcp.ini
# Sample xcp config
[xcp]
#catalog =  10.63.150.51:/gpfs1
catalog =  10.63.150.213:/nc_volume1
```

b. Copy the license file to `/opt/NetApp/xFiles/xcp/`.

```
root@workr-138: src$ cd /opt/NetApp/xFiles/xcp/
root@workr-138: xcp$ ls -ltrha
total 252K
drwxr-xr-x 3 root    root      16 Apr  4  2019 ..
-rw-r--r-- 1 root    root     105 Dec  5 19:04 xcp.ini
drwxr-xr-x 2 root    root      59 Dec  5 19:04 .
-rw-r--r-- 1 faiz89 faiz89   336 Dec  6 21:12 license
-rw-r--r-- 1 root    root     192 Dec  6 21:13 host
-rw-r--r-- 1 root    root    236K Dec 17 14:12 xcp.log
root@workr-138: xcp$
```

c. Activate XCP using the `xcp activate` command.

d. Check the source for NFS export.

```
[root@hdp2 linux]# ./xcp show 10.63.150.92
XCP 1.4-17914d6; (c) 2019 NetApp, Inc.; Licensed to Karthikeyan
Nagalingam [NetApp Inc] until Wed Feb  5 11:07:27 2020
getting pmap dump from 10.63.150.92 port 111...
getting export list from 10.63.150.92...
sending 1 mount and 4 nfs requests to 10.63.150.92...
== RPC Services ==
'10.63.150.92': TCP rpc services: MNT v1/3, NFS v3/4, NFSACL v3, NLM
v1/3/4, PMAP v2/3/4, STATUS v1
'10.63.150.92': UDP rpc services: MNT v1/3, NFS v4, NFSACL v3, NLM
v1/3/4, PMAP v2/3/4, STATUS v1
== NFS Exports ==
 Mounts  Errors  Server
      1       0  10.63.150.92
     Space     Files      Space      Files
      Free      Free       Used       Used Export
  52.3 TiB    53.7B    8.36 GiB     53.7B 10.63.150.92:/maprnfs3
== Attributes of NFS Exports ==
drwxr-xr-x --- root root 2 2 10m51s 10.63.150.92:/maprnfs3
1.77 KiB in (8.68 KiB/s), 3.16 KiB out (15.5 KiB/s), 0s.
[root@hdp2 linux]#
```

e. Transfer the data using XCP from multiple MapR nodes from multiple source IPs and multiple destination IPs (ONTAP LIFs).

```
root@workr-138: linux$ ./xcp_yatin copy --parallel 20
10.63.150.96,10.63.150.97:/maprnfs3/tg4
10.63.150.85,10.63.150.86:/datapipeline_dataset/tg4_dest
XCP 1.6-dev; (c) 2019 NetApp, Inc.; Licensed to Karthikeyan
Nagalingam [NetApp Inc] until Wed Feb  5 11:07:27 2020
xcp: WARNING: No index name has been specified, creating one with
name: autoname_copy_2019-12-06_21.14.38.652652
xcp: mount '10.63.150.96,10.63.150.97:/maprnfs3/tg4': WARNING: This
NFS server only supports 1-second timestamp granularity. This may
cause sync to fail because changes will often be undetectable.
 130 scanned, 128 giants, 3.59 GiB in (723 MiB/s), 3.60 GiB out (724
MiB/s), 5s
 130 scanned, 128 giants, 8.01 GiB in (889 MiB/s), 8.02 GiB out (890
MiB/s), 11s
 130 scanned, 128 giants, 12.6 GiB in (933 MiB/s), 12.6 GiB out (934
MiB/s), 16s
 130 scanned, 128 giants, 16.7 GiB in (830 MiB/s), 16.7 GiB out (831
MiB/s), 21s
 130 scanned, 128 giants, 21.1 GiB in (907 MiB/s), 21.1 GiB out (908
MiB/s), 26s
```

```
 130 scanned, 128 giants, 25.5 GiB in (893 MiB/s), 25.5 GiB out (894
MiB/s), 31s
 130 scanned, 128 giants, 29.6 GiB in (842 MiB/s), 29.6 GiB out (843
MiB/s), 36s
….
[root@workr-140 linux]# ./xcp_yatin copy  --parallel 20
10.63.150.92:/maprnfs3/tg4_2
10.63.150.85,10.63.150.86:/datapipeline_dataset/tg4_2_dest
XCP 1.6-dev; (c) 2019 NetApp, Inc.; Licensed to Karthikeyan
Nagalingam [NetApp Inc] until Wed Feb  5 11:07:27 2020
xcp: WARNING: No index name has been specified, creating one with
name: autoname_copy_2019-12-06_21.14.24.637773
xcp: mount '10.63.150.92:/maprnfs3/tg4_2': WARNING: This NFS server
only supports 1-second timestamp granularity. This may cause sync to
fail because changes will often be undetectable.
 130 scanned, 128 giants, 4.39 GiB in (896 MiB/s), 4.39 GiB out (897
MiB/s), 5s
 130 scanned, 128 giants, 9.94 GiB in (1.10 GiB/s), 9.96 GiB out
(1.10 GiB/s), 10s
 130 scanned, 128 giants, 15.4 GiB in (1.09 GiB/s), 15.4 GiB out
(1.09 GiB/s), 15s
 130 scanned, 128 giants, 20.1 GiB in (953 MiB/s), 20.1 GiB out (954
MiB/s), 20s
 130 scanned, 128 giants, 24.6 GiB in (928 MiB/s), 24.7 GiB out (929
MiB/s), 25s
 130 scanned, 128 giants, 29.0 GiB in (877 MiB/s), 29.0 GiB out (878
MiB/s), 31s
 130 scanned, 128 giants, 33.2 GiB in (852 MiB/s), 33.2 GiB out (853
MiB/s), 36s
 130 scanned, 128 giants, 37.8 GiB in (941 MiB/s), 37.8 GiB out (942
MiB/s), 41s
 130 scanned, 128 giants, 42.0 GiB in (860 MiB/s), 42.0 GiB out (861
MiB/s), 46s
 130 scanned, 128 giants, 46.1 GiB in (852 MiB/s), 46.2 GiB out (853
MiB/s), 51s
 130 scanned, 128 giants, 50.1 GiB in (816 MiB/s), 50.2 GiB out (817
MiB/s), 56s
 130 scanned, 128 giants, 54.1 GiB in (819 MiB/s), 54.2 GiB out (820
MiB/s), 1m1s
 130 scanned, 128 giants, 58.5 GiB in (897 MiB/s), 58.6 GiB out (898
MiB/s), 1m6s
 130 scanned, 128 giants, 62.9 GiB in (900 MiB/s), 63.0 GiB out (901
MiB/s), 1m11s
 130 scanned, 128 giants, 67.2 GiB in (876 MiB/s), 67.2 GiB out (877
MiB/s), 1m16s
```

f. Check the load distribution on the storage controller.

```
Hadoop-AFF8080::*> statistics show-periodic -interval 2 -iterations 0
-summary true -object nic_common -counter rx_bytes|tx_bytes -node
Hadoop-AFF8080-01 -instance e3b
Hadoop-AFF8080: nic_common.e3b: 12/6/2019 15:55:04
 rx_bytes tx_bytes
 -------- --------
     879MB    4.67MB
     856MB    4.46MB
     973MB    5.66MB
     986MB    5.88MB
     945MB    5.30MB
     920MB    4.92MB
     894MB    4.76MB
     902MB    4.79MB
     886MB    4.68MB
     892MB    4.78MB
     908MB    4.96MB
     905MB    4.85MB
     899MB    4.83MB
Hadoop-AFF8080::*> statistics show-periodic -interval 2 -iterations 0
-summary true -object nic_common -counter rx_bytes|tx_bytes -node
Hadoop-AFF8080-01 -instance e9b
Hadoop-AFF8080: nic_common.e9b: 12/6/2019 15:55:07
 rx_bytes tx_bytes
 -------- --------
     950MB    4.93MB
     991MB    5.84MB
     959MB    5.63MB
     914MB    5.06MB
     903MB    4.81MB
     899MB    4.73MB
     892MB    4.71MB
     890MB    4.72MB
     905MB    4.86MB
     902MB    4.90MB
```

Where to find additional information

To learn more about the information that is described in this document, review the following documents and/or websites:

- NetApp In-Place Analytics Module Best Practices

  https://www.netapp.com/us/media/tr-4382.pdf

- NetApp FlexGroup Volume Best Practices and Implementation Guide

  https://www.netapp.com/us/media/tr-4571.pdf

- NetApp Product Documentation

  https://www.netapp.com/us/documentation/index.aspx

Version history

Version	Date	Document version history
Version 3.0	January 2022	Directly move data from HDFS and MapR-FS to NFS by using NetApp XCP.
Version 2.0	January 2020	XCP included as the default data mover. Added MapR-FS to NFS and GPFS to NFS data transfer.
Version 1.0	November 2018	Initial release.

Best practices for Confluent Kafka

TR-4912: Best practice guidelines for Confluent Kafka tiered storage with NetApp

Karthikeyan Nagalingam, Joseph Kandatilparambil, NetApp
Rankesh Kumar, Confluent

Apache Kafka is a community-distributed event-streaming platform capable of handling trillions of events a day. Initially conceived as a messaging queue, Kafka is based on an abstraction of a distributed commit log. Since it was created and open-sourced by LinkedIn in 2011, Kafka has evolved from a messages queue to a full-fledged event-streaming platform. Confluent delivers the distribution of Apache Kafka with the Confluent Platform. The Confluent Platform supplements Kafka with additional community and commercial features designed to enhance the streaming experience of both operators and developers in production at a massive scale.

This document describes the best-practice guidelines for using Confluent Tiered Storage on a NetApp's Object storage offering by providing the following content:

- Confluent verification with NetApp Object storage – NetApp StorageGRID
- Tiered storage performance tests
- Best-practice guidelines for Confluent on NetApp storage systems

Why Confluent Tiered Storage?

Confluent has become the default real-time streaming platform for many applications, especially for big data, analytics, and streaming workloads. Tiered Storage enables users to separate compute from storage in the

Confluent platform. It makes storing data more cost effective, enables you to store virtually infinite amounts of data and scale workloads up (or down) on-demand, and makes administrative tasks like data and tenant rebalancing easier. S3 compatible storage systems can take advantage of all these capabilities to democratize data with all events in one place, eliminating the need for complex data engineering. For more info on why you should use tiered storage for Kafka, check this article by Confluent.

Why NetApp StorageGRID for tiered storage?

StorageGRID is an industry-leading object storage platform by NetApp. StorageGRID is a software-defined, object-based storage solution that supports industry-standard object APIs, including the Amazon Simple Storage Service (S3) API. StorageGRID stores and manages unstructured data at scale to provide secure, durable object storage. Content is placed in the right location, at the right time, and on the right storage tier, optimizing workflows and reducing costs for globally distributed rich media.

The greatest differentiator for StorageGRID is its Information Lifecycle Management (ILM) policy engine that enables policy-driven data lifecycle management. The policy engine can use metadata to manage how data is stored across its lifetime to initially optimize for performance and automatically optimize for cost and durability as data ages.

Enabling Confluent Tiered Storage

The basic idea of tiered storage is to separate the tasks of data storage from data processing. With this separation, it becomes much easier for the data storage tier and the data processing tier to scale independently.

A tiered storage solution for Confluent must contend with two factors. First, it must work around or avoid common object store consistency and availability properties, such as inconsistencies in LIST operations and occasional object unavailability. Secondly, it must correctly handle the interaction between tiered storage and Kafka's replication and fault tolerance model, including the possibility of zombie leaders continuing to tier offset ranges. NetApp Object storage provides both the consistent object availability and HA model make the tired storage available to tier offset ranges. NetApp object storage provides consistent object availability and an HA model to make the tired storage available to tier offset ranges.

With tiered storage, you can use high-performance platforms for low-latency reads and writes near the tail of your streaming data, and you can also use cheaper, scalable object stores like NetApp StorageGRID for high-throughput historical reads. We also have technical solution for Spark with netapp storage controller and details are here. The following figure shows how Kafka fits into a real-time analytics pipeline.

Data Source   Ingest   Real-time processing   Storage   Visualization
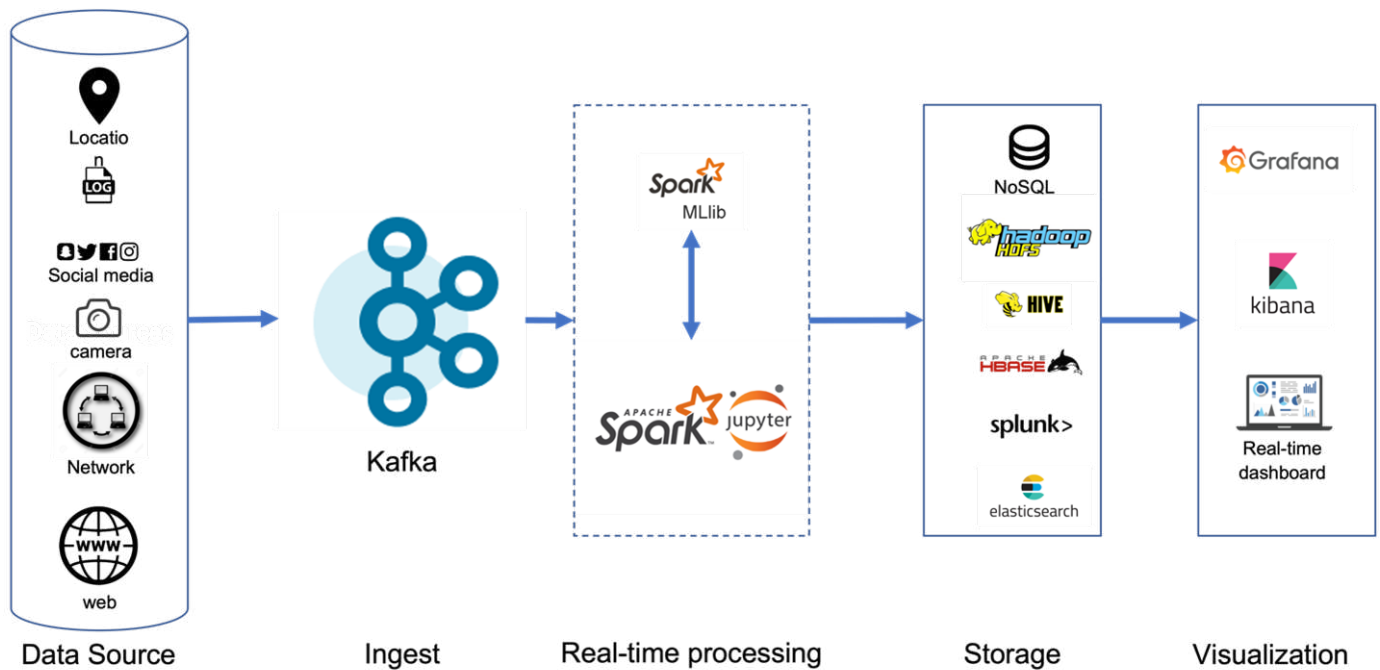
The following figure depicts how NetApp StorageGRID fits in as Confluent Kafka's object storage tier.
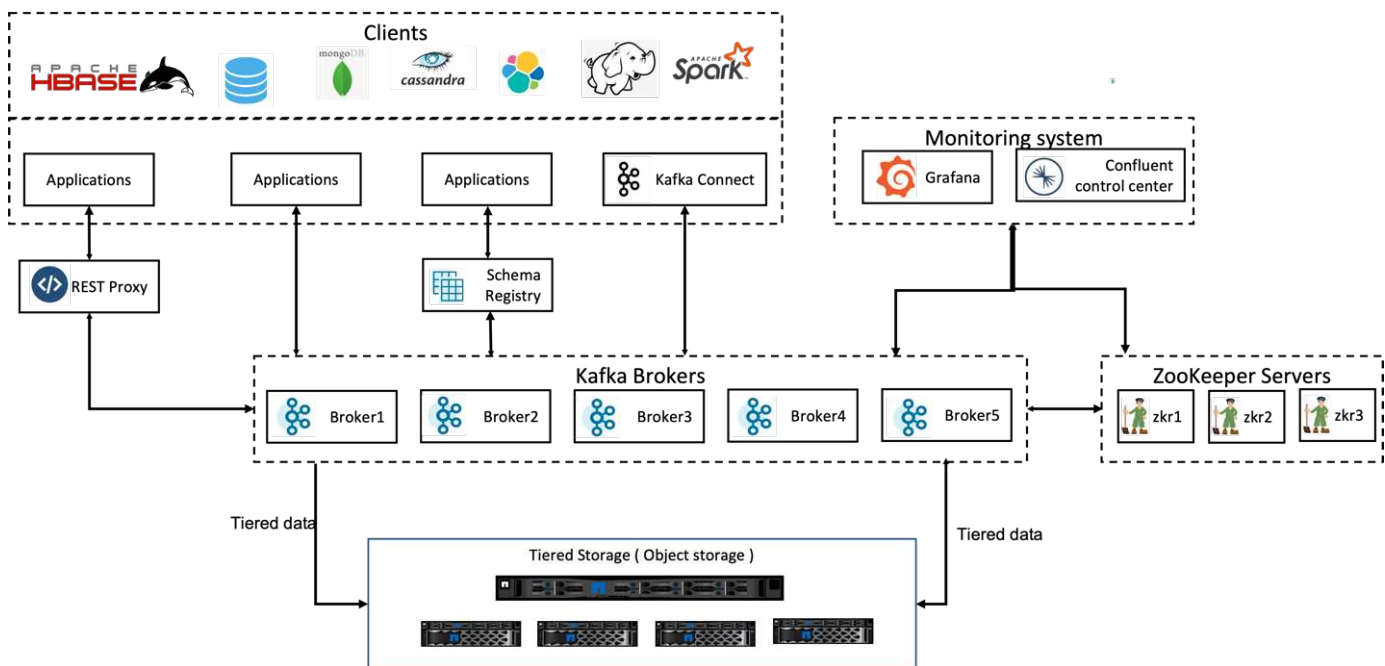
Solution architecture details

This section covers the hardware and software used for Confluent verification. This information is applicable to Confluent Platform deployment with NetApp storage. The following table covers the tested solution architecture and base components.

Solution components	Details
Confluent Kafka version 6.2	• Three zookeepers • Five broker servers • Five tools servers • One Grafana • One control center
Linux (ubuntu 18.04)	All servers
NetApp StorageGRID for tiered storage	• StorageGRID software • 1 x SG1000 (load balancer) • 4 x SGF6024 • 4 x 24 x 800 SSDs • S3 protocol • 4 x 100GbE (network connectivity between broker and StorageGRID instances)
15 Fujitsu PRIMERGY RX2540 servers	Each equipped with: * 2 CPUs, 16 physical cores total * Intel Xeon * 256GB physical memory * 100GbE dual port

Technology overview

This section describes the technology used in this solution.

NetApp StorageGRID

NetApp StorageGRID is a high-performance, cost-effective object storage platform. By using tiered storage, most of the data on Confluent Kafka, which is stored in local storage or the SAN storage of the broker, is offloaded to the remote object store. This configuration results in significant operational improvements by reducing the time and cost to rebalance, expand, or shrink clusters or replace a failed broker. Object storage plays an important role in managing data that resides on the object store tier, which is why picking the right object storage is important.

StorageGRID offers intelligent, policy-driven global data management using a distributed, node-based grid architecture. It simplifies the management of petabytes of unstructured data and billions of objects through its ubiquitous global object namespace combined with sophisticated data management features. Single-call object access extends across sites and simplifies high availability architectures while ensuring continual object access, regardless of site or infrastructure outages.

Multitenancy allows multiple unstructured cloud and enterprise data applications to be securely serviced within the same grid, increasing the ROI and use cases for NetApp StorageGRID. You can create multiple service levels with metadata-driven object lifecycle policies, optimizing durability, protection, performance, and locality

across multiple geographies. Users can adjust data management policies and monitor and apply traffic limits to realign with the data landscape nondisruptively as their requirements change in ever-changing IT environments.

Simple management with Grid Manager

The StorageGRID Grid Manager is a browser-based graphical interface that allows you to configure, manage, and monitor your StorageGRID system across globally distributed locations in a single pane of glass.
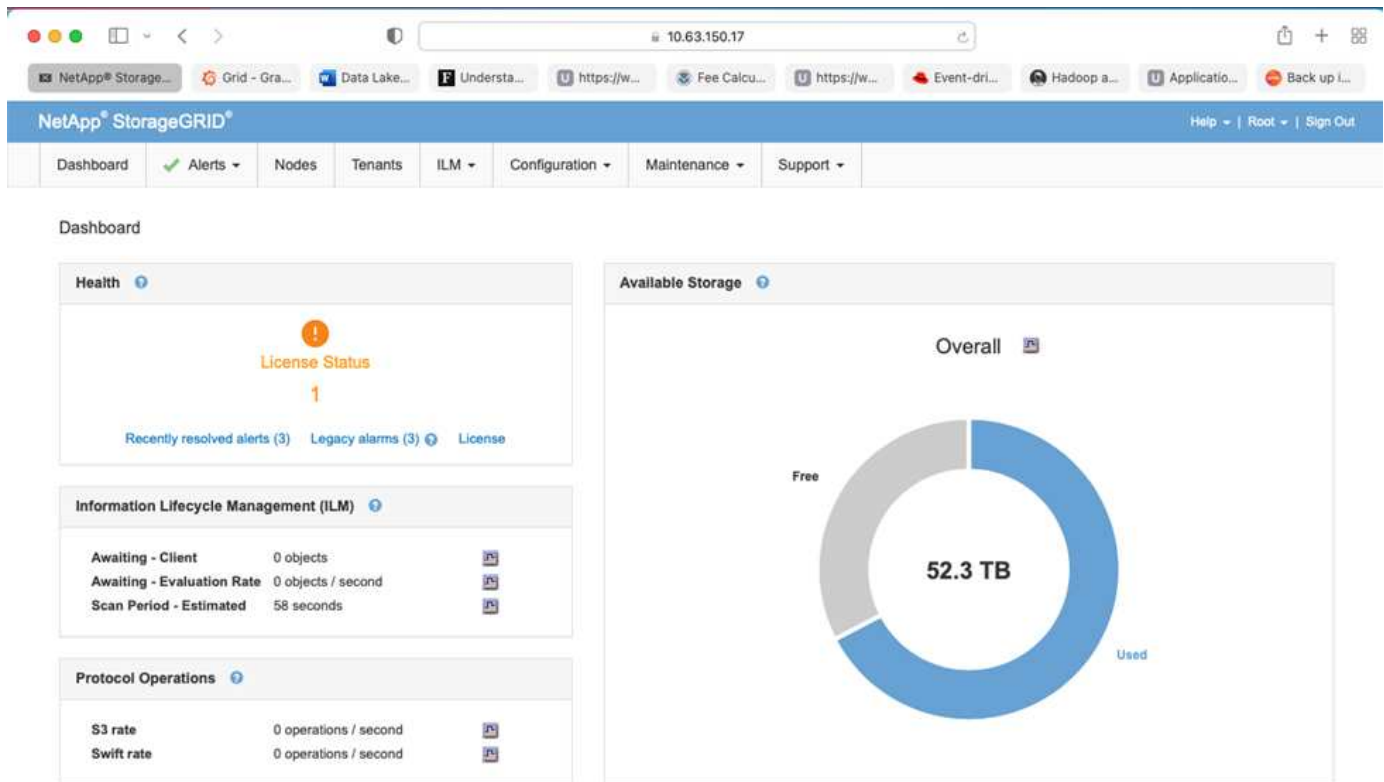
You can perform the following tasks with the StorageGRID Grid Manager interface:

- Manage globally distributed, petabyte-scale repositories of objects such as images, video, and records.

- Monitor grid nodes and services to ensure object availability.

- Manage the placement of object data over time using information lifecycle management (ILM) rules. These rules govern what happens to an object's data after it is ingested, how it is protected from loss, where object data is stored, and for how long.

- Monitor transactions, performance, and operations within the system.

Information Lifecycle Management policies

StorageGRID has flexible data management policies that include keeping replica copies of your objects and using EC (erasure coding) schemes like 2+1 and 4+2 (among others) to store your objects, depending on specific performance and data protection requirements. As workloads and requirements change over time, it's common that ILM policies must change over time as well. Modifying ILM policies is a core feature, allowing StorageGRID customers to adapt to their ever-changing environment quickly and easily. Please check the ILM policy and ILM rules setup in StorageGRID.

Performance

StorageGRID scales performance by adding more storage nodes, which can be VMs, bare metal, or purpose-built appliances like the SG5712, SG5760, SG6060, or SGF6024. In our tests, we exceeded the Apache Kafka key performance requirements with a minimum-sized, three-node grid using the SGF6024 appliance. As customers scale their Kafka cluster with additional brokers, they can add more storage nodes to increase performance and capacity.

Load balancer and endpoint configuration

Admin nodes in StorageGRID provide the Grid Manager UI (user interface) and REST API endpoint to view, configure, and manage your StorageGRID system, as well as audit logs to track system activity. To provide a highly available S3 endpoint for Confluent Kafka tiered storage, we implemented the StorageGRID load balancer, which runs as a service on admin nodes and gateway nodes. In addition, the load balancer also manages local traffic and talks to the GSLB (Global Server Load Balancing) to help with disaster recovery.

To further enhance endpoint configuration, StorageGRID provides traffic classification policies built into the admin node, lets you monitor your workload traffic, and applies various quality-of-service (QoS) limits to your workloads. Traffic classification policies are applied to endpoints on the StorageGRID Load Balancer service for gateway nodes and admin nodes. These policies can assist with traffic shaping and monitoring.

Traffic classification in StorageGRID

StorageGRID has built-in QoS functionality. Traffic classification policies can help monitor different types of S3 traffic coming from a client application. You can then create and apply policies to put limits on this traffic based on in/out bandwidth, the number of read/write concurrent requests, or the read/write request rate.

Apache Kafka

Apache Kafka is a framework implementation of a software bus using stream processing written in Java and Scala. It's aimed to provide a unified, high-throughput, low-latency platform for handling real-time data feeds. Kafka can connect to an external system for data export and import through Kafka Connect and provides Kafka streams, a Java stream processing library. Kafka uses a binary, TCP-based protocol that is optimized for efficiency and relies on a "message set" abstraction that naturally groups messages together to reduce the overhead of the network roundtrip. This enables larger sequential disk operations, larger network packets, and contiguous memory blocks, thereby enabling Kafka to turn a bursty stream of random message writes into linear writes. The following figure depicts the basic data flow of Apache Kafka.
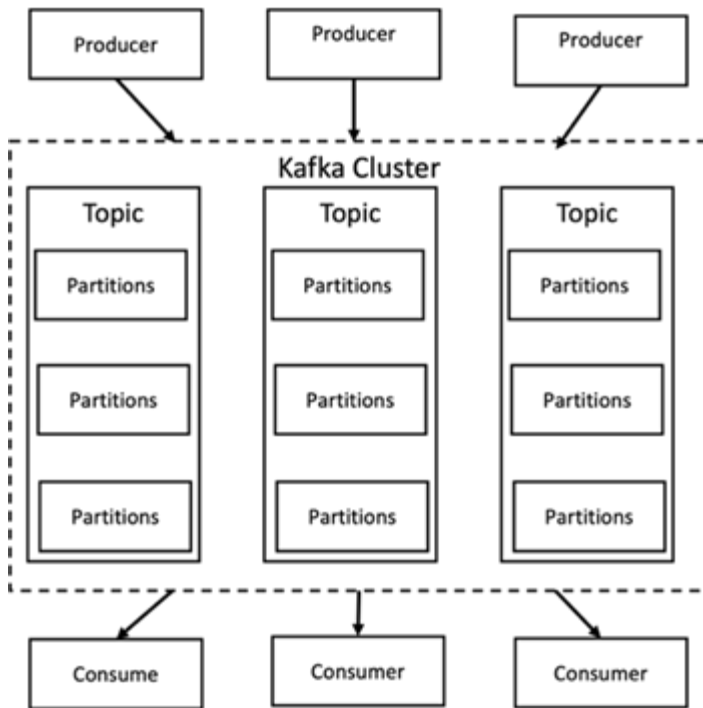
Kafka stores key-value messages that come from an arbitrary number of processes called producers. The data can be partitioned into different partitions within different topics. Within a partition, messages are strictly ordered by their offsets (the position of a message within a partition) and indexed and stored together with a timestamp. Other processes called consumers can read messages from partitions. For stream processing, Kafka offers the Streams API that allows writing Java applications that consume data from Kafka and write results back to Kafka. Apache Kafka also works with external stream processing systems such as Apache Apex, Apache Flink, Apache Spark, Apache Storm, and Apache NiFi.

Kafka runs on a cluster of one or more servers (called brokers), and the partitions of all topics are distributed across the cluster nodes. Additionally, partitions are replicated to multiple brokers. This architecture allows Kafka to deliver massive streams of messages in a fault-tolerant fashion and has allowed it to replace some of the conventional messaging systems like Java Message Service (JMS), Advanced Message Queuing Protocol (AMQP), and so on. Since the 0.11.0.0 release, Kafka offers transactional writes, which provide exactly once stream processing using the Streams API.

Kafka supports two types of topics: regular and compacted. Regular topics can be configured with a retention time or a space bound. If there are records that are older than the specified retention time or if the space bound is exceeded for a partition, Kafka is allowed to delete old data to free storage space. By default, topics are configured with a retention time of 7 days, but it's also possible to store data indefinitely. For compacted topics, records don't expire based on time or space bounds. Instead, Kafka treats later messages as updates to older message with the same key and guarantees never to delete the latest message per key. Users can delete messages entirely by writing a so-called tombstone message with the null value for a specific key.

There are five major APIs in Kafka:

- **Producer API.** Permits an application to publish streams of records.
- **Consumer API.** Permits an application to subscribe to topics and processes streams of records.
- **Connector API.** Executes the reusable producer and consumer APIs that can link the topics to the existing applications.
- **Streams API.** This API converts the input streams to output and produces the result.
- **Admin API.** Used to manage Kafka topics, brokers and other Kafka objects.

The consumer and producer APIs build on top of the Kafka messaging protocol and offer a reference implementation for Kafka consumer and producer clients in Java. The underlying messaging protocol is a binary protocol that developers can use to write their own consumer or producer clients in any programming language. This unlocks Kafka from the Java Virtual Machine (JVM) ecosystem. A list of available non-Java clients is maintained in the Apache Kafka wiki.

Apache Kafka use cases

Apache Kafka is most popular for messaging, website activity tracking, metrics, log aggregation, stream processing, event sourcing, and commit logging.

- Kafka has improved throughput, built-in partitioning, replication, and fault-tolerance, which makes it a good solution for large-scale message-processing applications.

- Kafka can rebuild a user's activities (page views, searches) in a tracking pipeline as a set of real-time publish-subscribe feeds.

- Kafka is often used for operational monitoring data. This involves aggregating statistics from distributed applications to produce centralized feeds of operational data.

- Many people use Kafka as a replacement for a log aggregation solution. Log aggregation typically collects physical log files off of servers and puts them in a central place (for example, a file server or HDFS) for processing. Kafka abstracts files details and provides a cleaner abstraction of log or event data as a stream of messages. This allows for lower-latency processing and easier support for multiple data sources and distributed data consumption.

- Many users of Kafka process data in processing pipelines consisting of multiple stages, in which raw input data is consumed from Kafka topics and then aggregated, enriched, or otherwise transformed into new topics for further consumption or follow-up processing. For example, a processing pipeline for recommending news articles might crawl article content from RSS feeds and publish it to an "articles" topic. Further processing might normalize or deduplicate this content and publish the cleansed article content to a new topic, and a final processing stage might attempt to recommend this content to users. Such processing pipelines create graphs of real-time data flows based on the individual topics.

- Event souring is a style of application design for which state changes are logged as a time-ordered sequence of records. Kafka's support for very large stored log data makes it an excellent backend for an application built in this style.

- Kafka can serve as a kind of external commit-log for a distributed system. The log helps replicate data between nodes and acts as a re-syncing mechanism for failed nodes to restore their data. The log compaction feature in Kafka helps support this use case.

Confluent

Confluent Platform is an enterprise-ready platform that completes Kafka with advanced capabilities designed to help accelerate application development and connectivity, enable transformations through stream processing, simplify enterprise operations at scale, and meet stringent architectural requirements. Built by the original creators of Apache Kafka, Confluent expands the benefits of Kafka with enterprise-grade features while removing the burden of Kafka management or monitoring. Today, over 80% of the Fortune 100 are powered by data streaming technology – and most of those use Confluent.
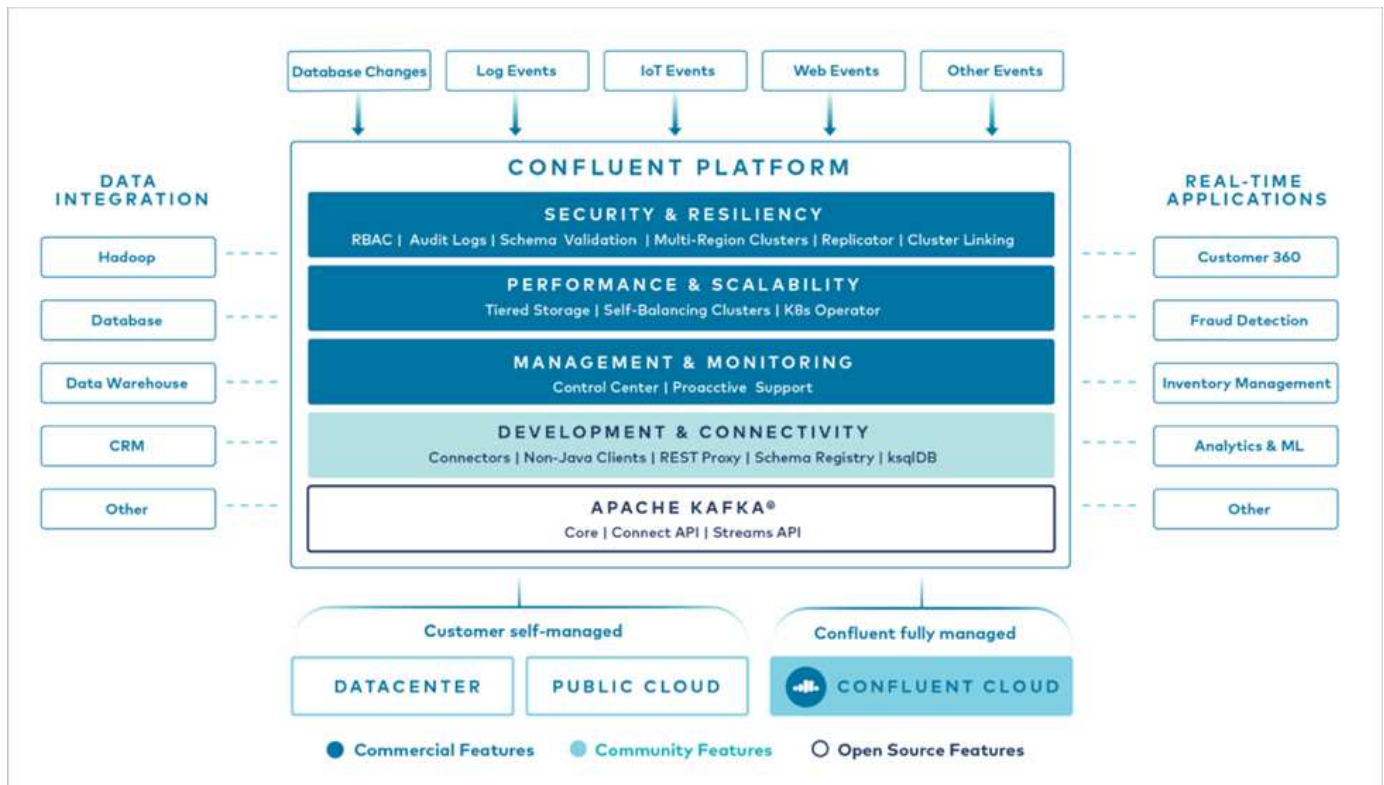
Why Confluent?

By integrating historical and real-time data into a single, central source of truth, Confluent makes it easy to build an entirely new category of modern, event-driven applications, gain a universal data pipeline, and unlock powerful new use cases with full scalability, performance, and reliability.

What is Confluent used for?

Confluent Platform lets you focus on how to derive business value from your data rather than worrying about the underlying mechanics, such as how data is being transported or integrated between disparate systems. Specifically, Confluent Platform simplifies connecting data sources to Kafka, building streaming applications, as well as securing, monitoring, and managing your Kafka infrastructure. Today, Confluent Platform is used for a wide array of use cases across numerous industries, from financial services, omnichannel retail, and autonomous cars, to fraud detection, microservices, and IoT.

The following figure shows Confluent Kafka Platform components.

Overview of Confluent's event streaming technology

At the core of Confluent Platform is Apache Kafka, the most popular open-source distributed streaming platform. The key capabilities of Kafka are as follows:

- Publish and subscribe to streams of records.
- Store streams of records in a fault tolerant way.
- Process streams of records.

Out of the box, Confluent Platform also includes Schema Registry, REST Proxy, a total of 100+ prebuilt Kafka connectors, and ksqlDB.

Overview of Confluent platform's enterprise features

- **Confluent Control Center.** A GUI-based system for managing and monitoring Kafka. It allows you to easily manage Kafka Connect and to create, edit, and manage connections to other systems.
- **Confluent for Kubernetes.** Confluent for Kubernetes is a Kubernetes operator. Kubernetes operators extend the orchestration capabilities of Kubernetes by providing the unique features and requirements for a specific platform application. For Confluent Platform, this includes greatly simplifying the deployment

process of Kafka on Kubernetes and automating typical infrastructure lifecycle tasks.

- **Confluent connectors to Kafka.** Connectors use the Kafka Connect API to connect Kafka to other systems such as databases, key-value stores, search indexes, and file systems. Confluent Hub has downloadable connectors for the most popular data sources and sinks, including fully tested and supported versions of these connectors with Confluent Platform. More details can be found here.

- **Self- balancing clusters.** Provides automated load balancing, failure detection and self-healing. It provides support for adding or decommissioning brokers as needed, with no manual tuning.

- **Confluent cluster linking.** Directly connects clusters together and mirrors topics from one cluster to another over a link bridge. Cluster linking simplifies setup of multi-datacenter, multi-cluster, and hybrid cloud deployments.

- **Confluent auto data balancer.** Monitors your cluster for the number of brokers, the size of partitions, number of partitions, and the number of leaders within the cluster. It allows you to shift data to create an even workload across your cluster, while throttling rebalance traffic to minimize the effect on production workloads while rebalancing.

- **Confluent replicator.** Makes it easier than ever to maintain multiple Kafka clusters in multiple data centers.

- **Tiered storage.** Provides options for storing large volumes of Kafka data using your favorite cloud provider, thereby reducing operational burden and cost. With tiered storage, you can keep data on cost-effective object storage and scale brokers only when you need more compute resources.

- **Confluent JMS client.** Confluent Platform includes a JMS-compatible client for Kafka. This Kafka client implements the JMS 1.1 standard API, using Kafka brokers as the backend. This is useful if you have legacy applications using JMS and you would like to replace the existing JMS message broker with Kafka.

- **Confluent MQTT proxy.** Provides a way to publish data directly to Kafka from MQTT devices and gateways without the need for a MQTT broker in the middle.

- **Confluent security plugins.** Confluent security plugins are used to add security capabilities to various Confluent Platform tools and products. Currently, there is a plugin available for the Confluent REST proxy that helps to authenticate the incoming requests and propagate the authenticated principal to requests to Kafka. This enables Confluent REST proxy clients to utilize the multitenant security features of the Kafka broker.

Confluent verification

We performed verification with Confluent Platform 6.2 Tiered Storage in NetApp StorageGRID. The NetApp and Confluent teams worked on this verification together and ran the test cases required for verification.

Confluent Platform setup

We used the following setup for verification.

For verification, we used three zookeepers, five brokers, five test-script executing servers, named tools servers with 256GB RAM, and 16 CPUs. For NetApp storage, we used StorageGRID with an SG1000 load balancer with four SGF6024s. The storage and brokers were connected via 100GbE connections.

The following figure shows the network topology of configuration used for Confluent verification.

The tools servers act as application clients that send requests to Confluent nodes.

Confluent tiered storage configuration

The tiered storage configuration requires the following parameters in Kafka:

```
Confluent.tier.archiver.num.threads=16
confluent.tier.fetcher.num.threads=32
confluent.tier.enable=true
confluent.tier.feature=true
confluent.tier.backend=S3
confluent.tier.s3.bucket=kafkasgdbucket1-2
confluent.tier.s3.region=us-west-2
confluent.tier.s3.cred.file.path=/data/kafka/.ssh/credentials
confluent.tier.s3.aws.endpoint.override=http://kafkasgd.rtpppe.netapp.com:
10444/
confluent.tier.s3.force.path.style.access=true
```

For verification, we used StorageGRID with the HTTP protocol, but HTTPS also works. The access key and secret key are stored in the file name provided in the `confluent.tier.s3.cred.file.path` parameter.

NetApp object storage - StorageGRID

We configured single-site configuration in StorageGRID for verfication.

Verification tests

We completed the following five test cases for the verification. These tests are executed on the Trogdor framework. The first two were functionality tests and the remaining three were performance tests.

Object store correctness test

This test determines whether all basic operations (for example, get/put/delete) on the object store API work well according to the needs of tiered storage. It is a basic test that every object store service should expect to pass ahead of the following tests. It is an assertive test that either passes or fails.

Tiering functionality correctness test

This test determines if end-to-end tiered storage functionality works well with an assertive test that either passes or fails. The test creates a test topic that by default is configured with tiering enabled and highly a reduced hotset size. It produces an event stream to the newly created test topic, it waits for the brokers to archive the segments to the object store, and it then consumes the event stream and validates that the consumed stream matches the produced stream. The number of messages produced to the event stream is configurable, which lets the user generate a sufficiently large workload according to the needs of testing. The reduced hotset size ensures that the consumer fetches outside the active segment are served only from the object store; this helps test the correctness of the object store for reads. We have performed this test with and without an object-store fault injection. We simulated node failure by stopping the service manager service in one of the nodes in StorageGRID and validating that the end-to-end functionality works with object storage.

Tier fetch benchmark

This test validated the read performance of the tiered object storage and checked the range fetch read requests under heavy load from segments generated by the benchmark. In this benchmark, Confluent developed custom clients to serve the tier fetch requests.

Produce-consume workload benchmark

This test indirectly generated write workload on the object store through the archival of segments. The read workload (segments read) was generated from object storage when consumer groups fetched the segments. This workload was generated by the test script. This test checked the performance of read and write on the object storage in parallel threads. We tested with and without object store fault injection as we did for the tiering functionality correctness test.

Retention workload benchmark

This test checked the deletion performance of an object store under a heavy topic-retention workload. The retention workload was generated using a test script that produces many messages in parallel to a test topic. The test topic was configuring with an aggressive size-based and time-based retention setting that caused the event stream to be continuously purged from the object store. The segments were then archived. This led to a large number of deletions in the object storage by the broker and collection of the performance of the object-store delete operations.

Performance tests with scalability

We performed the tiered storage testing with three to four nodes for producer and consumer workloads with the NetApp StorageGRID setup. According to our tests, the time to completion and the performance results were directly proportional to the number of StorageGRID nodes. The StorageGRID setup required a minimum of three nodes.

- The time to complete the produce and consumer operation decreased linearly when the number of storage nodes increased.

Time to complete trends
(Lower is better)
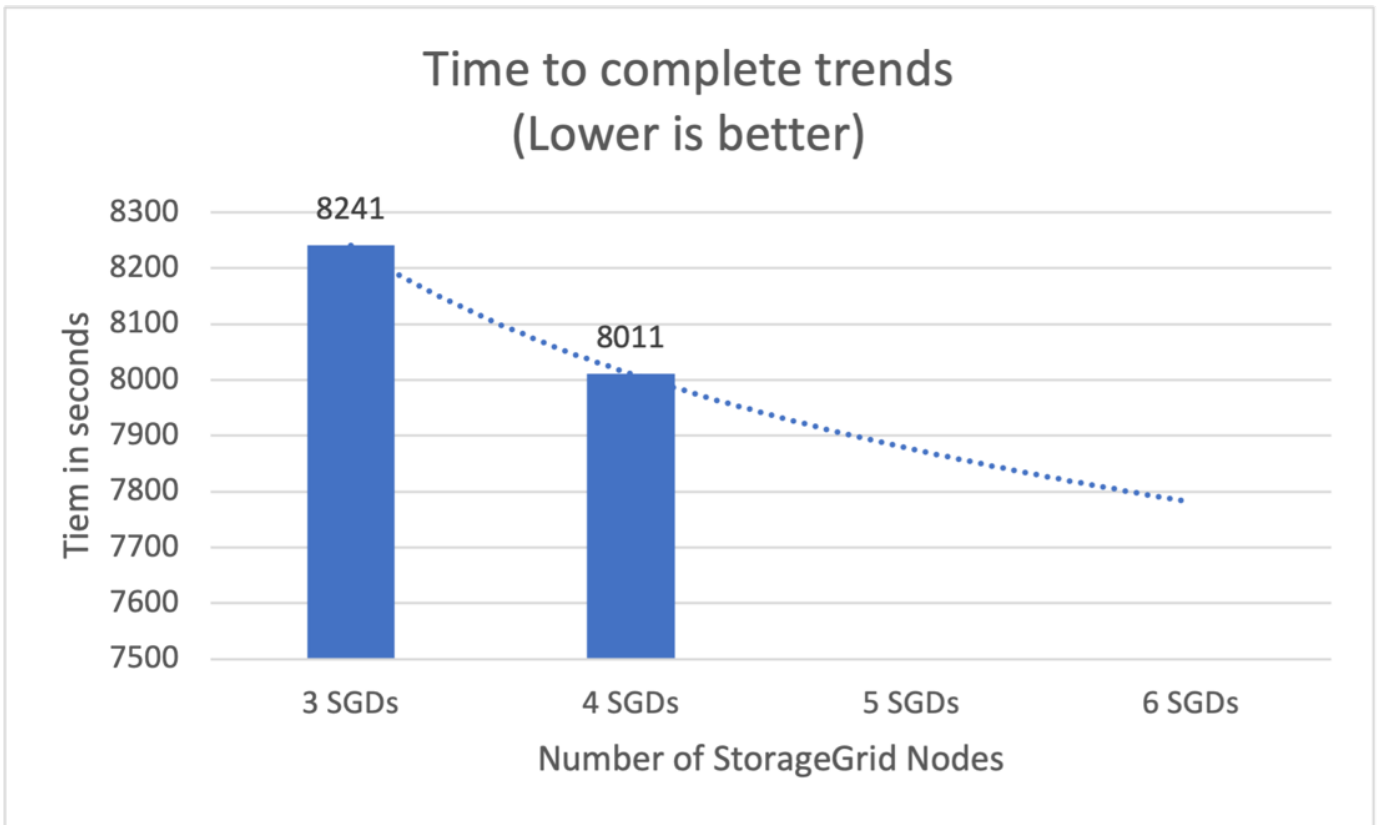
- The performance for the s3 retrieve operation increased linearly based on number of StorageGRID nodes. StorageGRID supports up to 200 StorgeGRID nodes.

S3 - Retrieve peformance Trend
(Higher is better)
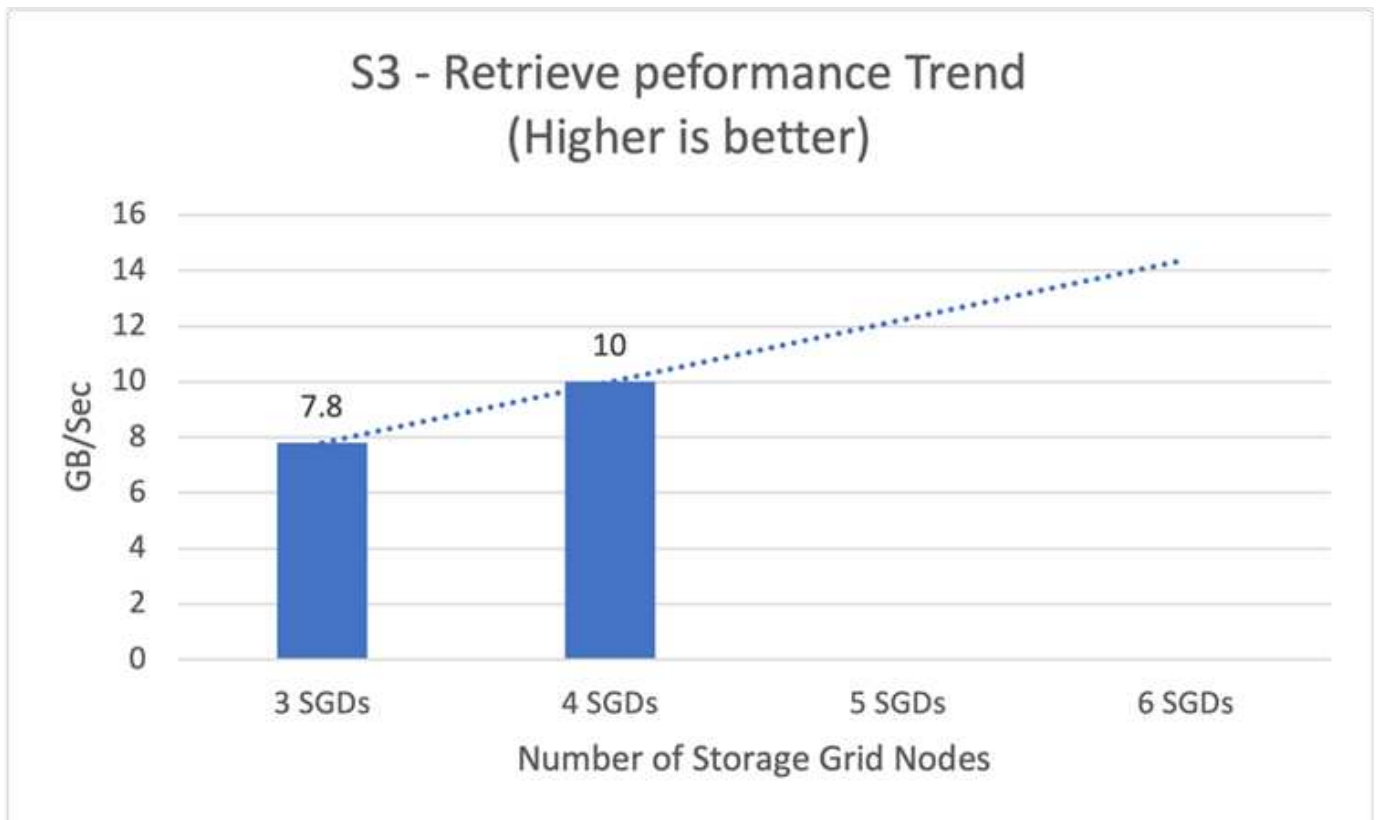
Next: Confluent s3 connector.

Confluent s3 connector

The Amazon S3 Sink connector exports data from Apache Kafka topics to S3 objects in either the Avro, JSON, or Bytes formats. The Amazon S3 sink connector periodically polls data from Kafka and in turn uploads it to S3. A partitioner is used to split the data of every Kafka partition into chunks. Each chunk of data is represented as an S3 object. The key name encodes the topic, the Kafka partition, and the start offset of this data chunk.

In this setup, we show you how to read and write topics in object storage from Kafka directly using the Kafka s3 sink connector. For this test, we used a stand-alone Confluent cluster, but this setup is applicable to a distributed cluster.

1. Download Confluent Kafka from the Confluent website.

2. Unpack the package to a folder on your server.

3. Export two variables.

```
Export CONFLUENT_HOME=/data/confluent/confluent-6.2.0
export PATH=$PATH:/data/confluent/confluent-6.2.0/bin
```

4. For a stand-alone Confluent Kafka setup, the cluster creates a temporary root folder in /tmp. It also creates Zookeeper, Kafka, a schema registry, connect, a ksql-server, and control-center folders and copies their respective configuration files from $CONFLUENT_HOME. See the following example:

```
root@stlrx2540m1-108:~# ls -ltr /tmp/confluent.406980/
total 28
drwxr-xr-x 4 root root 4096 Oct 29 19:01 zookeeper
drwxr-xr-x 4 root root 4096 Oct 29 19:37 kafka
drwxr-xr-x 4 root root 4096 Oct 29 19:40 schema-registry
drwxr-xr-x 4 root root 4096 Oct 29 19:45 kafka-rest
drwxr-xr-x 4 root root 4096 Oct 29 19:47 connect
drwxr-xr-x 4 root root 4096 Oct 29 19:48 ksql-server
drwxr-xr-x 4 root root 4096 Oct 29 19:53 control-center
root@stlrx2540m1-108:~#
```

5. Configure Zookeeper. You don't need to change anything if you use the default parameters.

```
root@stlrx2540m1-108:~# cat
/tmp/confluent.406980/zookeeper/zookeeper.properties   | grep -iv ^#
dataDir=/tmp/confluent.406980/zookeeper/data
clientPort=2181
maxClientCnxns=0
admin.enableServer=false
tickTime=2000
initLimit=5
syncLimit=2
server.179=controlcenter:2888:3888
root@stlrx2540m1-108:~#
```

In the above configuration, we updated the `server. xxx` property. By default, you need three Zookeepers for the Kafka leader selection.

6. We created a myid file in `/tmp/confluent.406980/zookeeper/data` with a unique ID:

```
root@stlrx2540m1-108:~# cat /tmp/confluent.406980/zookeeper/data/myid
179
root@stlrx2540m1-108:~#
```

We used the last number of IP addresses for the myid file. We used default values for the Kafka, connect, control-center, Kafka, Kafka-rest, ksql-server, and schema-registry configurations.

7. Start the Kafka services.

```
root@stlrx2540m1-108:/data/confluent/confluent-6.2.0/bin# confluent
local services  start
The local commands are intended for a single-node development
environment only,
NOT for production usage.

Using CONFLUENT_CURRENT: /tmp/confluent.406980
ZooKeeper is [UP]
Kafka is [UP]
Schema Registry is [UP]
Kafka REST is [UP]
Connect is [UP]
ksqlDB Server is [UP]
Control Center is [UP]
root@stlrx2540m1-108:/data/confluent/confluent-6.2.0/bin#
```

There is a log folder for each configuration, which helps troubleshoot issues. In some instances, services take more time to start. Make sure all services are up and running.

8. Install Kafka connect using `confluent-hub`.

```
root@stlrx2540m1-108:/data/confluent/confluent-6.2.0/bin# ./confluent-
hub install confluentinc/kafka-connect-s3:latest
The component can be installed in any of the following Confluent
Platform installations:
  1. /data/confluent/confluent-6.2.0 (based on $CONFLUENT_HOME)
  2. /data/confluent/confluent-6.2.0 (where this tool is installed)
Choose one of these to continue the installation (1-2): 1
Do you want to install this into /data/confluent/confluent-
6.2.0/share/confluent-hub-components? (yN) y


Component's license:
Confluent Community License
http://www.confluent.io/confluent-community-license
I agree to the software license agreement (yN) y
Downloading component Kafka Connect S3 10.0.3, provided by Confluent,
Inc. from Confluent Hub and installing into /data/confluent/confluent-
6.2.0/share/confluent-hub-components
Do you want to uninstall existing version 10.0.3? (yN) y
Detected Worker's configs:
  1. Standard: /data/confluent/confluent-6.2.0/etc/kafka/connect-
distributed.properties
  2. Standard: /data/confluent/confluent-6.2.0/etc/kafka/connect-
standalone.properties
  3. Standard: /data/confluent/confluent-6.2.0/etc/schema-
registry/connect-avro-distributed.properties
  4. Standard: /data/confluent/confluent-6.2.0/etc/schema-
registry/connect-avro-standalone.properties
  5. Based on CONFLUENT_CURRENT:
/tmp/confluent.406980/connect/connect.properties
  6. Used by Connect process with PID 15904:
/tmp/confluent.406980/connect/connect.properties
Do you want to update all detected configs? (yN) y
Adding installation directory to plugin path in the following files:
  /data/confluent/confluent-6.2.0/etc/kafka/connect-
distributed.properties
  /data/confluent/confluent-6.2.0/etc/kafka/connect-
standalone.properties
  /data/confluent/confluent-6.2.0/etc/schema-registry/connect-avro-
distributed.properties
  /data/confluent/confluent-6.2.0/etc/schema-registry/connect-avro-
standalone.properties
  /tmp/confluent.406980/connect/connect.properties
  /tmp/confluent.406980/connect/connect.properties
```

```
Completed
root@stlrx2540m1-108:/data/confluent/confluent-6.2.0/bin#
```

You can also install a specific version by using `confluent-hub install confluentinc/kafka-connect-s3:10.0.3`.

9. By default, `confluentinc-kafka-connect-s3` is installed in `/data/confluent/confluent-6.2.0/share/confluent-hub-components/confluentinc-kafka-connect-s3`.

10. Update the plug-in path with the new `confluentinc-kafka-connect-s3`.

```
root@stlrx2540m1-108:~# cat /data/confluent/confluent-
6.2.0/etc/kafka/connect-distributed.properties | grep plugin.path
#
plugin.path=/usr/local/share/java,/usr/local/share/kafka/plugins,/opt/co
nnectors,
plugin.path=/usr/share/java,/data/zookeeper/confluent/confluent-
6.2.0/share/confluent-hub-components,/data/confluent/confluent-
6.2.0/share/confluent-hub-components,/data/confluent/confluent-
6.2.0/share/confluent-hub-components/confluentinc-kafka-connect-s3
root@stlrx2540m1-108:~#
```

11. Stop the Confluent services and restart them.

```
confluent local services  stop
confluent local services  start
root@stlrx2540m1-108:/data/confluent/confluent-6.2.0/bin# confluent
local services  status
The local commands are intended for a single-node development
environment only,
NOT for production usage.

Using CONFLUENT_CURRENT: /tmp/confluent.406980
Connect is [UP]
Control Center is [UP]
Kafka is [UP]
Kafka REST is [UP]
ksqlDB Server is [UP]
Schema Registry is [UP]
ZooKeeper is [UP]
root@stlrx2540m1-108:/data/confluent/confluent-6.2.0/bin#
```

12. Configure the access ID and secret key in the `/root/.aws/credentials` file.

```
root@stlrx2540m1-108:~# cat /root/.aws/credentials
[default]
aws_access_key_id = xxxxxxxxxxxx
aws_secret_access_key = xxxxxxxxxxxxxxxxxxxxxxxxxxxx
root@stlrx2540m1-108:~#
```

13. Verify that the bucket is reachable.

```
root@stlrx2540m4-01:~# aws s3 –endpoint-url
http://kafkasgd.rtpppe.netapp.com:10444 ls kafkasgdbucket1-2
2021-10-29 21:04:18      1388 1
2021-10-29 21:04:20      1388 2
2021-10-29 21:04:22      1388 3
root@stlrx2540m4-01:~#
```

14. Configure the s3-sink properties file for s3 and bucket configuration.

```
root@stlrx2540m1-108:~# cat /data/confluent/confluent-
6.2.0/share/confluent-hub-components/confluentinc-kafka-connect-
s3/etc/quickstart-s3.properties | grep -v ^#
name=s3-sink
connector.class=io.confluent.connect.s3.S3SinkConnector
tasks.max=1
topics=s3_testtopic
s3.region=us-west-2
s3.bucket.name=kafkasgdbucket1-2
store.url=http://kafkasgd.rtpppe.netapp.com:10444/
s3.part.size=5242880
flush.size=3
storage.class=io.confluent.connect.s3.storage.S3Storage
format.class=io.confluent.connect.s3.format.avro.AvroFormat
partitioner.class=io.confluent.connect.storage.partitioner.DefaultPartit
ioner
schema.compatibility=NONE
root@stlrx2540m1-108:~#
```

15. Import a few records to the s3 bucket.

```
kafka-avro-console-producer --broker-list localhost:9092 --topic
s3_topic \
--property
value.schema='{"type":"record","name":"myrecord","fields":[{"name":"f1",
"type":"string"}]}'
{"f1": "value1"}
{"f1": "value2"}
{"f1": "value3"}
{"f1": "value4"}
{"f1": "value5"}
{"f1": "value6"}
{"f1": "value7"}
{"f1": "value8"}
{"f1": "value9"}
```

16. Load the s3-sink connector.

```
root@stlrx2540m1-108:~# confluent local services connect connector load
s3-sink  --config /data/confluent/confluent-6.2.0/share/confluent-hub-
components/confluentinc-kafka-connect-s3/etc/quickstart-s3.properties
The local commands are intended for a single-node development
environment only,
NOT for production usage.
https://docs.confluent.io/current/cli/index.html
{
   "name": "s3-sink",
   "config": {
     "connector.class": "io.confluent.connect.s3.S3SinkConnector",
     "flush.size": "3",
     "format.class": "io.confluent.connect.s3.format.avro.AvroFormat",
     "partitioner.class":
"io.confluent.connect.storage.partitioner.DefaultPartitioner",
     "s3.bucket.name": "kafkasgdbucket1-2",
     "s3.part.size": "5242880",
     "s3.region": "us-west-2",
     "schema.compatibility": "NONE",
     "storage.class": "io.confluent.connect.s3.storage.S3Storage",
     "store.url": "http://kafkasgd.rtpppe.netapp.com:10444/",
     "tasks.max": "1",
     "topics": "s3_testtopic",
     "name": "s3-sink"
   },
   "tasks": [],
   "type": "sink"
}
root@stlrx2540m1-108:~#
```

17. Check the s3-sink status.

```
root@stlrx2540m1-108:~# confluent local services connect connector
status s3-sink
The local commands are intended for a single-node development
environment only,
NOT for production usage.
https://docs.confluent.io/current/cli/index.html
{
  "name": "s3-sink",
  "connector": {
    "state": "RUNNING",
    "worker_id": "10.63.150.185:8083"
  },
  "tasks": [
    {
      "id": 0,
      "state": "RUNNING",
      "worker_id": "10.63.150.185:8083"
    }
  ],
  "type": "sink"
}
root@stlrx2540m1-108:~#
```

18. Check the log to make sure that s3-sink is ready to accept topics.

```
root@stlrx2540m1-108:~# confluent local services connect log
```

19. Check the topics in Kafka.

```
kafka-topics --list --bootstrap-server localhost:9092
…
connect-configs
connect-offsets
connect-statuses
default_ksql_processing_log
s3_testtopic
s3_topic
s3_topic_new
root@stlrx2540m1-108:~#
```

20. Check the objects in the s3 bucket.

```
root@stlrx2540m1-108:~# aws s3 --endpoint-url
http://kafkasgd.rtpppe.netapp.com:10444 ls --recursive kafkasgdbucket1-
2/topics/
2021-10-29 21:24:00        213
topics/s3_testtopic/partition=0/s3_testtopic+0+0000000000.avro
2021-10-29 21:24:00        213
topics/s3_testtopic/partition=0/s3_testtopic+0+0000000003.avro
2021-10-29 21:24:00        213
topics/s3_testtopic/partition=0/s3_testtopic+0+0000000006.avro
2021-10-29 21:24:08        213
topics/s3_testtopic/partition=0/s3_testtopic+0+0000000009.avro
2021-10-29 21:24:08        213
topics/s3_testtopic/partition=0/s3_testtopic+0+0000000012.avro
2021-10-29 21:24:09        213
topics/s3_testtopic/partition=0/s3_testtopic+0+0000000015.avro
root@stlrx2540m1-108:~#
```

21. To verify the contents, copy each file from S3 to your local filesystem by running the following command:

```
root@stlrx2540m1-108:~# aws s3 --endpoint-url
http://kafkasgd.rtpppe.netapp.com:10444 cp s3://kafkasgdbucket1-
2/topics/s3_testtopic/partition=0/s3_testtopic+0+0000000000.avro
tes.avro
download: s3://kafkasgdbucket1-
2/topics/s3_testtopic/partition=0/s3_testtopic+0+0000000000.avro to
./tes.avro
root@stlrx2540m1-108:~#
```

22. To print the records, use avro-tools-1.11.0.1.jar (available in the Apache Archives).

```
root@stlrx2540m1-108:~# java -jar /usr/src/avro-tools-1.11.0.1.jar
tojson tes.avro
21/10/30 00:20:24 WARN util.NativeCodeLoader: Unable to load native-
hadoop library for your platform... using builtin-java classes where
applicable
{"f1":"value1"}
{"f1":"value2"}
{"f1":"value3"}
root@stlrx2540m1-108:~#
```

Confluent Self-balancing Clusters

If you have managed a Kafka cluster before, you are likely familiar with the challenges that come with manually reassigning partitions to different brokers to make sure that the workload is balanced across the cluster. For organizations with large Kafka deployments, reshuffling large amounts of data can be daunting, tedious, and risky, especially if mission-critical applications are built on top of the cluster. However, even for the smallest Kafka use cases, the process is time consuming and prone to human error.

In our lab, we tested the Confluent self-balancing clusters feature, which automates rebalancing based on cluster topology changes or uneven load. The Confluent rebalance test helps to measure the time to add a new broker when node failure or the scaling node requires rebalancing data across brokers. In classic Kafka configurations, the amount of data to be rebalanced grows as the cluster grows, but, in tiered storage, rebalancing is restricted to a small amount of data. Based on our validation, rebalancing in tiered storage takes seconds or minutes in a classic Kafka architecture and grows linearly as the cluster grows.

In self-balancing clusters, partition rebalances are fully automated to optimize Kafka's throughput, accelerate broker scaling, and reduce the operational burden of running a large cluster. At steady-state, self-balancing clusters monitor the skew of data across the brokers and continuously reassigns partitions to optimize cluster performance. When scaling the platform up or down, self-balancing clusters automatically recognize the presence of new brokers or the removal of old brokers and trigger a subsequent partition reassignment. This enables you to easily add and decommission brokers, making your Kafka clusters fundamentally more elastic. These benefits come without any need for manual intervention, complex math, or the risk of human error that partition reassignments typically entail. As a result, data rebalances are completed in far less time, and you are free to focus on higher-value event-streaming projects rather than needing to constantly supervise your clusters.

Best practice guidelines

- Based on our validation, S3 object storage is best for Confluent to keep data.
- We can use high-throughput SAN (specifically FC) to keep the broker hot data or local disk, because, in the Confluent tiered storage configuration, the size of the data held in the brokers data directory is based on the segment size and retention time when the data is moved to object storage.
- Object stores provide better performance when segment.bytes is higher; we tested 512MB.
- In Kafka, the length of the key or value (in bytes) for each record produced to the topic is controlled by the `length.key.value` parameter. For StorageGRID, S3 object ingest and retrieve performance increased to higher values. For example, 512 bytes provided a 5.8GBps retrieve, 1024 bytes provided a 7.5GBps s3 retrieve, and 2048 bytes provided close to 10GBps.

The following figure presents the S3 object ingest and retrieve based on `length.key.value`.

S3 Ingest and Retrieve

- **Kafka tuning.** To improve the performance of tiered storage, you can increase TierFetcherNumThreads and TierArchiverNumThreads. As a general guideline, you want to increase TierFetcherNumThreads to match the number of physical CPU cores and increase TierArchiverNumThreads to half the number of CPU cores. For example, in server properties, if you have a machine with eight physical cores, set confluent.tier.fetcher.num.threads = 8 and confluent.tier.archiver.num.threads = 4.

- **Time interval for topic deletes.** When a topic is deleted, deletion of the log segment files in object storage does not immediately begin. Rather, there is a time interval with a default value of 3 hours before deletion of those files takes place. You can modify the configuration, confluent.tier.topic.delete.check.interval.ms, to change the value of this interval. If you delete a topic or cluster, you can also manually delete the objects in the respective bucket.

- **ACLs on tiered storage internal topics.** A recommended best practice for on-premises deployments is to enable an ACL authorizer on the internal topics used for tiered storage. Set ACL rules to limit access on this data to the broker user only. This secures the internal topics and prevents unauthorized access to tiered storage data and metadata.

```
kafka-acls --bootstrap-server localhost:9092 --command-config adminclient-
configs.conf \
--add --allow-principal User:<kafka> --operation All --topic "_confluent-
tier-state"
```

> ℹ️ Replace the user `<kafka>` with the actual broker principal in your deployment.

For example, the command `confluent-tier-state` sets ACLs on the internal topic for tiered storage. Currently, there is only a single internal topic related to tiered storage. The example creates an ACL that provides the principal Kafka permission for all operations on the internal topic.

Next: Sizing.

Sizing

Previous: Best practice guidelines.

Kafka sizing can be performed with four configuration modes: simple, granular, reverse, and partitions.

Simple

The simple mode is appropriate for the first-time Apache Kafka users or early state use cases. For this mode, you provide requirements such as throughput MBps, read fanout, retention, and the resource utilization percentage (60% is default). You also enter the environment, such as on-premises (bare-metal, VMware, Kubernetes, or OpenStack) or cloud. Based on this information, the sizing of a Kafka cluster provides the number of servers required for the broker, the zookeeper, Apache Kafka connect workers, the schema registry, a REST Proxy, ksqlDB, and the Confluent control center.

For tiered storage, consider the granular configuration mode for sizing a Kafka cluster. Granular mode is appropriate for experienced Apache Kafka users or well-defined use cases. This section describes sizing for producers, stream processors, and consumers.

Producers

To describe the producers for Apache Kafka (for example a native client, REST proxy, or Kafka connector), provide the following information:

- **Name.** Spark.
- **Producer type.** Application or service, proxy (REST, MQTT, other), and existing database (RDBMS, NOSQL, other). You can also select "I don't know."
- **Average throughput.** In events per second (1,000,000 for example).
- **Peak throughput.** In events per second (4,000,000 for example).
- **Average message size.** In bytes, uncompressed (max 1MB; 1000 for example).
- **Message format.** Options include Avro, JSON, protocol buffers, binary, text, "I don't know," and other.
- **Replication factor.** Options are 1, 2, 3 (Confluent recommendation), 4, 5, or 6.
- **Retention time.** One day (for example). How long do you want your data to be stored in Apache Kafka? Enter -1 with any unit for an infinite time. The calculator assumes a retention time of 10 years for infinite retention.
- Select the check box for "Enable Tiered Storage to Decrease Broker Count and Allow for Infinite Storage?"
- When tiered storage is enabled, the retention fields control the hot set of data that is stored locally on the broker. The archival retention fields control how long data is stored in archival object storage.
- **Archival Storage Retention.** One year (for example). How long do you want your data to be stored in archival storage? Enter -1 with any unit for an infinite duration. The calculator assumes a retention of 10 years for infinite retention.
- **Growth Multiplier.** 1 (for example). If the value of this parameter is based on current throughput, set it to 1. To size based on additional growth, set this parameter to a growth multiplier.
- **Number of producer instances.** 10 (for example). How many producer instances will be running? This input is required to incorporate the CPU load into the sizing calculation. A blank value indicates that CPU load is not incorporated into the calculation.

Based on this example input, sizing has the following effect on producers:

- Average throughput in uncompressed bytes: 1GBps. Peak throughput in uncompressed bytes: 4GBps. Average throughput in compressed bytes: 400MBps. Peak throughput in compressed bytes: 1.6GBps. This is based on a default 60% compression rate (you can change this value).

- Total on-broker hotset storage required: 31,104TB, including replication, compressed. Total off-broker archival storage required: 378,432TB, compressed. Use https://fusion.netapp.com for StorageGRID sizing.

Stream Processors must describe their applications or services that consume data from Apache Kafka and produce back into Apache Kafka. In most cases these are built in ksqlDB or Kafka Streams.

- **Name.** Spark streamer.
- **Processing time.** How long does this processor take to process a single message?
  - 1 ms (simple, stateless transformation) [example], 10ms (stateful in-memory operation).
  - 100ms (stateful network or disk operation), 1000ms (3rd party REST call).
  - I have benchmarked this parameter and know exactly how long it takes.
- **Output Retention.** 1 day (example). A stream processor produces its output back to Apache Kafka. How long do you want this output data to be stored in Apache Kafka? Enter -1 with any unit for an infinite duration.
- Select the check box "Enable Tiered Storage to Decrease Broker Count and Allow for Infinite Storage?"
- **Archival Storage Retention.** 1 year (for example). How long do you want your data to be stored in archival storage? Enter -1 with any unit for an infinite duration. The calculator assumes a retention of 10 years for infinite retention.
- **Output Passthrough Percentage.** 100 (for example). A stream processor produces its output back to Apache Kafka. What percentage of inbound throughput will be outputted back into Apache Kafka? For example, if inbound throughput is 20MBps and this value is 10, the output throughput will be 2MBps.
- From which applications does this read from? Select "Spark," the name used in producer type-based sizing.
  Based on the above input, you can expect the following effects of sizing on stream processer instances and topic partition estimates:
- This stream processor application requires the following number of instances. The incoming topics likely require this many partitions as well. Contact Confluent to confirm this parameter.
  - 1,000 for average throughput with no growth multiplier
  - 4,000 for peak throughput with no growth multiplier
  - 1,000 for average throughput with a growth multiplier
  - 4,000 for peak throughput with a growth multiplier

Consumers

Describe your applications or services that consume data from Apache Kafka and do not produce back into Apache Kafka; for example, a native client or Kafka Connector.

- **Name.** Spark consumer.
- **Processing time.** How long does this consumer take to process a single message?
  - 1ms (for example, a simple and stateless task like logging)
  - 10ms (fast writes to a datastore)
  - 100ms (slow writes to a datastore)
  - 1000ms (third party REST call)
  - Some other benchmarked process of known duration.

- **Consumer type.** Application, proxy, or sink to an existing datastore (RDBMS, NoSQL, other).
- From which applications does this read from? Connect this parameter with producer and stream sizing determined previously.

Based on the above input, you must determine the sizing for consumer instances and topic partition estimates. A consumer application requires the following number of instances.

- 2,000 for average throughput, no growth multiplier
- 8,000 for peak throughput, no growth multiplier
- 2,000 for average throughput, including growth multiplier
- 8,000 for peak throughput, including growth multiplier

The incoming topics likely need this number of partitions as well. Contact Confluent to confirm.

In addition to the requirements for producers, stream processors, and consumers, you must provide the following additional requirements:

- **Rebuild time.** For example, 4 hours. If an Apache Kafka broker host fails, its data is lost, and a new host is provisioned to replace the failed host, how fast must this new host rebuild itself? Leave this parameter blank if the value is unknown.
- **Resource utilization target (percentage).** For example, 60. How utilized do you want your hosts to be during average throughput? Confluent recommends 60% utilization unless you are using Confluent self-balancing clusters, in which case utilization can be higher.

Describe your environment

- **What environment will your cluster be running in?** Amazon Web Services, Microsoft Azure, Google cloud platform, bare-metal on premises, VMware on premises, OpenStack on premises, or Kubernates on premises?
- **Host details.** Number of cores: 48 (for example), network card type (10GbE, 40GbE, 16GbE, 1GbE, or another type).
- **Storage volumes.** Host: 12 (for example). How many hard drives or SSDs are supported per host? Confluent recommends 12 hard drives per host.
- **Storage capacity/volume (in GB).** 1000 (for example). How much storage can a single volume store in gigabytes? Confluent recommends 1TB disks.
- **Storage configuration.** How are storage volumes configured? Confluent recommends RAID10 to take advantage of all Confluent features. JBOD, SAN, RAID 1, RAID 0, RAID 5, and other types are also supported.
- **Single volume throughput (MBps).** 125 (for example). How fast can a single storage volume read or write in megabytes per second? Confluent recommends standard hard drives, which typically have 125MBps throughput.
- **Memory capacity (GB).** 64 (for example).

After you have determined your environmental variables, select Size my Cluster. Based on the example parameters indicated above, we determined the following sizing for Confluent Kafka:

- **Apache Kafka.** Broker count: 22. Your cluster is storage-bound. Consider enabling tiered storage to decrease your host count and allow for infinite storage.
- **Apache ZooKeeper.** Count: 5; Apache Kafka Connect Workers: Count: 2; Schema Registry: Count: 2;

REST Proxy: Count: 2; ksqlDB: Count: 2; Confluent Control Center: Count: 1.

Use reverse mode for platform teams without a use case in mind. Use partitions mode to calculate how many partitions a single topic requires. See https://eventsizer.io for sizing based on the reverse and partitions modes.

Conclusion

This document provides best practice guidelines for using Confluent Tiered Storage with NetApp storage, including verification tests, tiered storage performance results, tuning, Confluent S3 connectors, and the self-balancing feature. Considering ILM policies, Confluent performance with multiple performance tests for verification, and industry-standard S3 APIs, NetApp StorageGRID object storage is an optimal choice for Confluent tiered storage.

Where to find additional information

To learn more about the information that is described in this document, review the following documents and/or websites:

- What is Apache Kafka

  https://www.confluent.io/what-is-apache-kafka/

- NetApp Product Documentation

  https://www.netapp.com/support-and-training/documentation/

- S3-sink parameter details

  https://docs.confluent.io/kafka-connect-s3-sink/current/configuration_options.html#s3-configuration-options

- Apache Kafka

  https://en.wikipedia.org/wiki/Apache_Kafka

- Infinite Storage in Confluent Platform

  https://www.confluent.io/blog/infinite-kafka-storage-in-confluent-platform/

- Confluent Tiered Storage - Best practices and sizing

  https://docs.confluent.io/platform/current/kafka/tiered-storage.html#best-practices-and-recommendations

- Amazon S3 sink connector for Confluent Platform

  https://docs.confluent.io/kafka-connect-s3-sink/current/overview.html

- Kafka sizing

  https://eventsizer.io

- StorageGRID sizing

https://fusion.netapp.com/

- Kafka use cases

  https://kafka.apache.org/uses

- Self-balancing Kafka clusters in confluent platform 6.0

  https://www.confluent.io/blog/self-balancing-kafka-clusters-in-confluent-platform-6-0/

  https://www.confluent.io/blog/confluent-platform-6-0-delivers-the-most-powerful-event-streaming-platform-to-date/

Version history

Version	Date	Document version history
Version 1.0	December 2021	Initial release.

NetApp hybrid cloud data solutions - Spark and Hadoop based on customer use cases

TR-4657: NetApp hybrid cloud data solutions - Spark and Hadoop based on customer use cases

Karthikeyan Nagalingam and Sathish Thyagarajan, NetApp

This document describes hybrid cloud data solutions using NetApp AFF and FAS storage systems, NetApp Cloud Volumes ONTAP, NetApp connected storage, and NetApp FlexClone technology for Spark and Hadoop. These solution architectures allow customers to choose an appropriate data protection solution for their environment. NetApp designed these solutions based on interaction with customers and their business use-cases. This document provides the following detailed information:

- Why we need data protection for Spark and Hadoop environments and customer challenges.
- The data fabric powered by NetApp vision and its building blocks and services.
- How these building blocks can be used to architect flexible data protection workflows.
- The pros and cons of several architectures based on real-world customer use cases. Each use case provides the following components:
  - Customer scenarios
  - Requirements and challenges
  - Solutions
  - Summary of the solutions

Why Hadoop data protection?

In a Hadoop and Spark environment, the following concerns must be addressed:

- **Software or human failures.** Human error in software updates while carrying out Hadoop data operations can lead to faulty behavior that can cause unexpected results from the job. In such case, we need to protect the data to avoid failures or unreasonable outcomes. For example, as the result of a poorly

executed software update to a traffic signal analysis application, a new feature that fails to properly analyze traffic signal data in the form of plain text. The software still analyzes JSON and other non- text file formats, resulting in the real-time traffic control analytics system producing prediction results that are missing data points. This situation can cause faulty outputs that might lead to accidents at the traffic signals. Data protection can address this issue by providing the capability to quickly roll back to the previous working application version.

- **Size and scale.** The size of the analytics data grows day by day due to the ever-increasing numbers of data sources and volume. Social media, mobile apps, data analytics, and cloud computing platforms are the main sources of data in the current big data market, which is increasing very rapidly, and therefore the data needs to be protected to ensure accurate data operations.

- **Hadoop's native data protection.** Hadoop has a native command to protect the data, but this command does not provide consistency of data during backup. It only supports directory-level backup. The snapshots created by Hadoop are read-only and cannot be used to reuse the backup data directly.

Data protection challenges for Hadoop and Spark customers

A common challenge for Hadoop and Spark customers is to reduce the backup time and increase backup reliability without negatively affecting performance at the production cluster during data protection.

Customers also need to minimize recovery point objective (RPO) and recovery time objective (RTO) downtime and control their on-premises and cloud-based disaster recovery sites for optimal business continuity. This control typically comes from having enterprise-level management tools.

The Hadoop and Spark environments are complicated because not only is the data volume huge and growing, but the rate this data arrives is increasing. This scenario makes it difficult to rapidly create efficient, up-to-date DevTest and QA environments from the source data. NetApp recognizes these challenges and offers the solutions presented in this paper.

Data fabric powered by NetApp for big data architecture

The data fabric powered by NetApp simplifies and integrates data management across cloud and on-premises environments to accelerate digital transformation.

The data fabric powered by NetApp delivers consistent and integrated data management services and applications (building blocks) for data visibility and insights, data access and control, and data protection and security, as shown in the figure below.
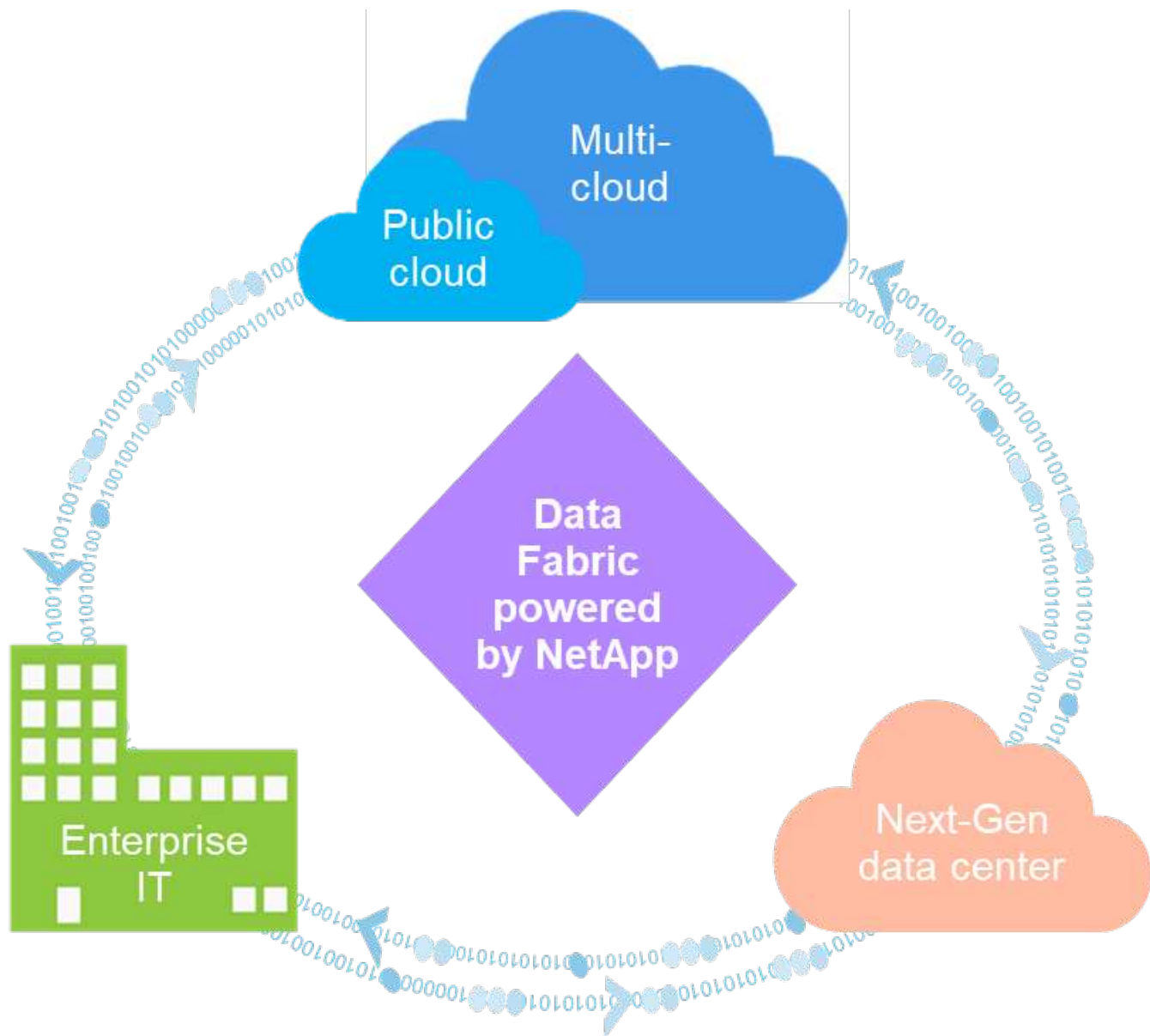
Proven data fabric customer use cases

The data fabric powered by NetApp provides the following nine proven use cases for customers:

- Accelerate analytics workloads
- Accelerate DevOps transformation
- Build cloud hosting infrastructure
- Integrate cloud data services
- Protect and secure data
- Optimize unstructured data
- Gain data center efficiencies
- Deliver data insights and control
- Simplify and automate

This document covers two of the nine use cases (along with their solutions):

- Accelerate analytics workloads
- Protect and secure data

NetApp NFS direct access

The NetApp NFS direct access (formerly known as NetApp In-Place Analytics Module) (shown in the figure below) allows customers to run big data analytics jobs on their existing or new NFSv3 or NFSv4 data without moving or copying the data. It prevents multiple copies of data and eliminates the need to sync the data with a source. For example, in the financial sector, the movement of data from one place to another place must meet legal obligations, which is not an easy task. In this scenario, the NetApp NFS direct access analyzes the financial data from its original location. Another key benefit is that using the NetApp NFS direct access simplifies protecting Hadoop data by using native Hadoop commands and enables data protection workflows leveraging NetApp's rich data management portfolio.
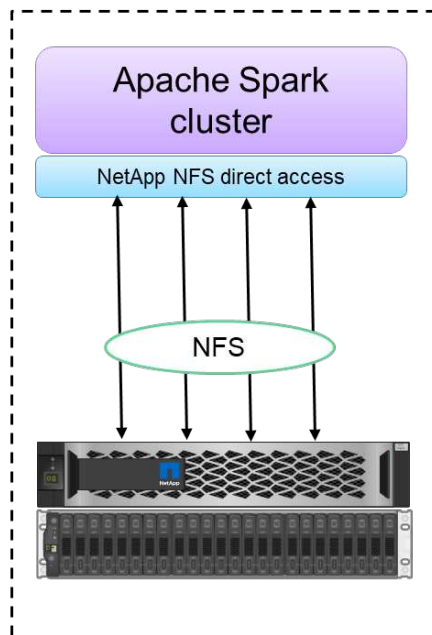
Configuration 1: NFS as primary storage
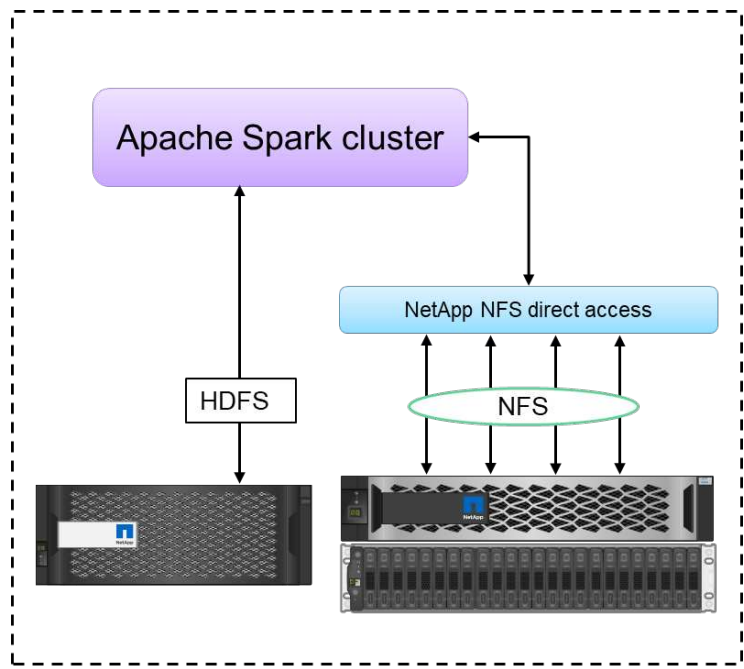
Configuration 2: HDFS and NFS in single Spark cluster

The NetApp NFS direct access provides two kinds of deployment options for Hadoop/Spark clusters:

- By default, the Hadoop/Spark clusters use Hadoop Distributed File System (HDFS) for data storage and the default file system. The NetApp NFS direct access can replace the default HDFS with NFS storage as the default file system, enabling direct analytics operations on NFS data.
- In another deployment option, the NetApp NFS direct access supports configuring NFS as additional storage along with HDFS in a single Hadoop/Spark cluster. In this case, the customer can share data through NFS exports and access it from the same cluster along with HDFS data.

The key benefits of using the NetApp NFS direct access include:

- Analyzes the data from its current location, which prevents the time- and performance-consuming task of moving analytics data to a Hadoop infrastructure such as HDFS.
- Reduces the number of replicas from three to one.
- Enables users to decouple the compute and storage to scale them independently.
- Provides enterprise data protection by leveraging the rich data management capabilities of ONTAP.

- Is certified with the Hortonworks data platform.

- Enables hybrid data analytics deployments.

- Reduces the backup time by leveraging dynamic multithread capability.

Building blocks for big data

The data fabric powered by NetApp integrates data management services and applications (building blocks) for data access, control, protection, and security, as shown in the figure below.
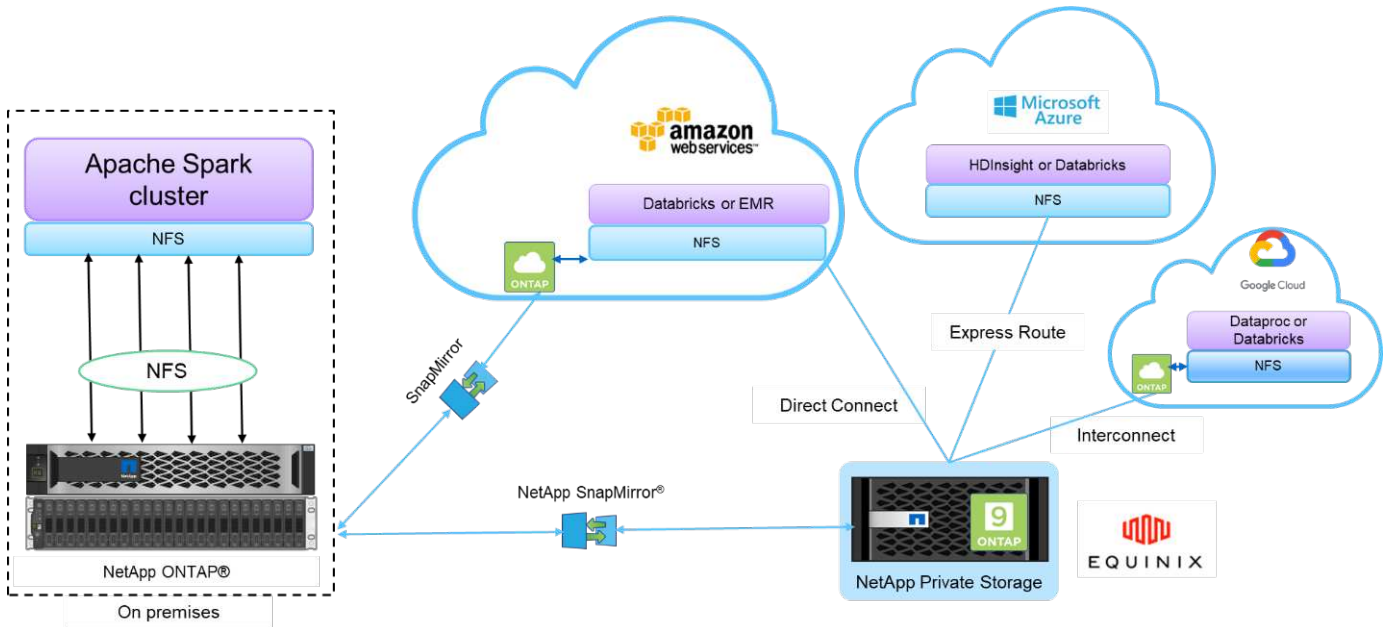
The building blocks in the figure above include:

- **NetApp NFS direct access.** Provides the latest Hadoop and Spark clusters with direct access to NetApp NFS volumes without additional software or driver requirements.

- **NetApp Cloud Volumes ONTAP and Cloud Volume Services.** Software-defined connected storage based on ONTAP running in Amazon Web Services (AWS) or Azure NetApp Files (ANF) in Microsoft Azure cloud services.

- **NetApp SnapMirror technology**. Provides data protection capabilities between on-premises and ONTAP Cloud or NPS instances.

- **Cloud service providers.** These providers include AWS, Microsoft Azure, Google Cloud, and IBM Cloud.

- **PaaS.** Cloud-based analytics services such as Amazon Elastic MapReduce (EMR) and Databricks in AWS as well as Microsoft Azure HDInsight and Azure Databricks.

Hadoop data protection and NetApp

Hadoop DistCp is a native tool used for large intercluster and intracluster copying. The Hadoop DistCp basic process shown in the figure below is a typical backup workflow using Hadoop native tools such as MapReduce to copy Hadoop data from an HDFS source to a corresponding target. The NetApp NFS direct access enables customers to set NFS as the target destination for the Hadoop DistCp tool to copy the data from HDFS source

into an NFS share through MapReduce. The NetApp NFS direct access acts as an NFS driver for the DistCp tool.
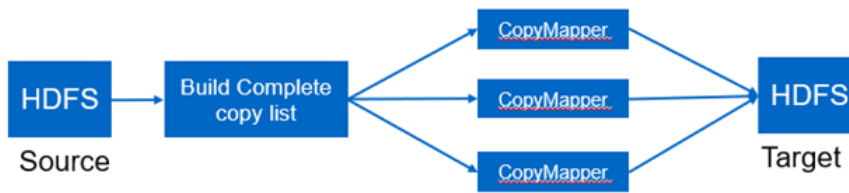
Hadoop Distcp Basic Process
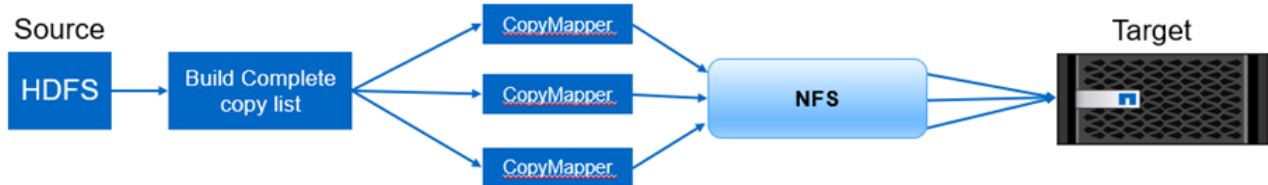
Hadoop Distcp and NetApp

Overview of Hadoop data protection use cases

This section provides a high-level description of the data protection use cases, which constitute the focus of this paper. The remaining sections provide more details for each use case, such as the customer problem (scenario), requirements and challenges, and solutions.

Use case 1: Backing up Hadoop data

For this use case, the In-Place Analytics Module helped a large financial institution reduce the long backup window time from more than 24 hours to just under a few hours.

Use case 2: Backup and disaster recovery from the cloud to on-premises

By using the data fabric powered by NetApp as building blocks, a large broadcasting company was able to fulfill its requirement of backing up cloud data into its on-premise data center depending on the different modes of data transfers, such as on demand, instantaneous, or based on the Hadoop/Spark cluster load.

Use case 3: Enabling DevTest on existing Hadoop data

NetApp solutions helped an online music distributor to rapidly build multiple space-efficient Hadoop clusters in different branches to create reports and run daily DevTest tasks by using scheduled policies.

Use case 4: Data protection and multicloud connectivity

A large service provider used the data fabric powered by NetApp to provide multicloud analytics to its customers from different cloud instances.

Use case 5: Accelerate analytic workloads

One of the largest financial services and investment banks used the NetApp network-attached storage solution to reduce I/O wait time and accelerate its quantitative financial analytics platform.

Use case 1: Backing up Hadoop data

Scenario

In this scenario, the customer has a large on-premises Hadoop repository and wants to back it up for disaster recovery purposes. However, the customer's current backup solution is costly and is suffering from a long backup window of more than 24 hours.

Requirements and challenges

The main requirements and challenges for this use case include:

- Software backward compatibility:

    ◦ The proposed alternative backup solution should be compatible with the current running software versions used in the production Hadoop cluster.

- To meet the committed SLAs, the proposed alternative solution should achieve very low RPOs and RTOs.

- The backup created by the NetApp backup solution can be used in the Hadoop cluster built locally in the data center as well as the Hadoop cluster running in the disaster recovery location at the remote site.

- The proposed solution must be cost effective.

- The proposed solution must reduce the performance effect on the currently running, in-production analytics jobs during the backup times.

Customer's existing backup solution

The figure below shows the original Hadoop native backup solution.
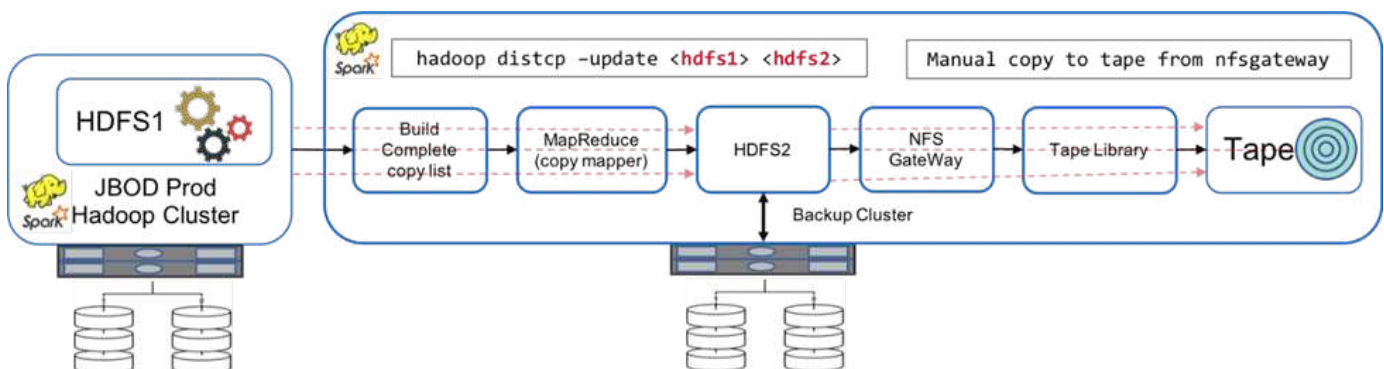
The production data is protected to tape through the intermediate backup cluster:

- HDFS1 data is copied to HDFS2 by running the `hadoop distcp -update <hdfs1> <hdfs2>` command.

- The backup cluster acts as an NFS gateway, and the data is manually copied to tape through the Linux `cp`

command through the tape library.

The benefits of the original Hadoop native backup solution include:

- The solution is based on Hadoop native commands, which saves the user from having to learn new procedures.
- The solution leverages industry-standard architecture and hardware.

The disadvantages of the original Hadoop native backup solution include:

- The long backup window time exceeds 24 hours, which makes the production data vulnerable.
- Significant cluster performance degradation during backup times.
- Copying to tape is a manual process.
- The backup solution is expensive in terms of the hardware required and the human hours required for manual processes.

Backup solutions

Based on these challenges and requirements, and taking into consideration the existing backup system, three possible backup solutions were suggested. The following subsections describe each of these three different backup solutions, labeled solution A through solution C.

Solution A

Solution A adds the In-Place Analytics Module to the backup Hadoop cluster, which allows secondary backups to NetApp NFS storage systems, eliminating the tape requirement, as shown in the figure below.
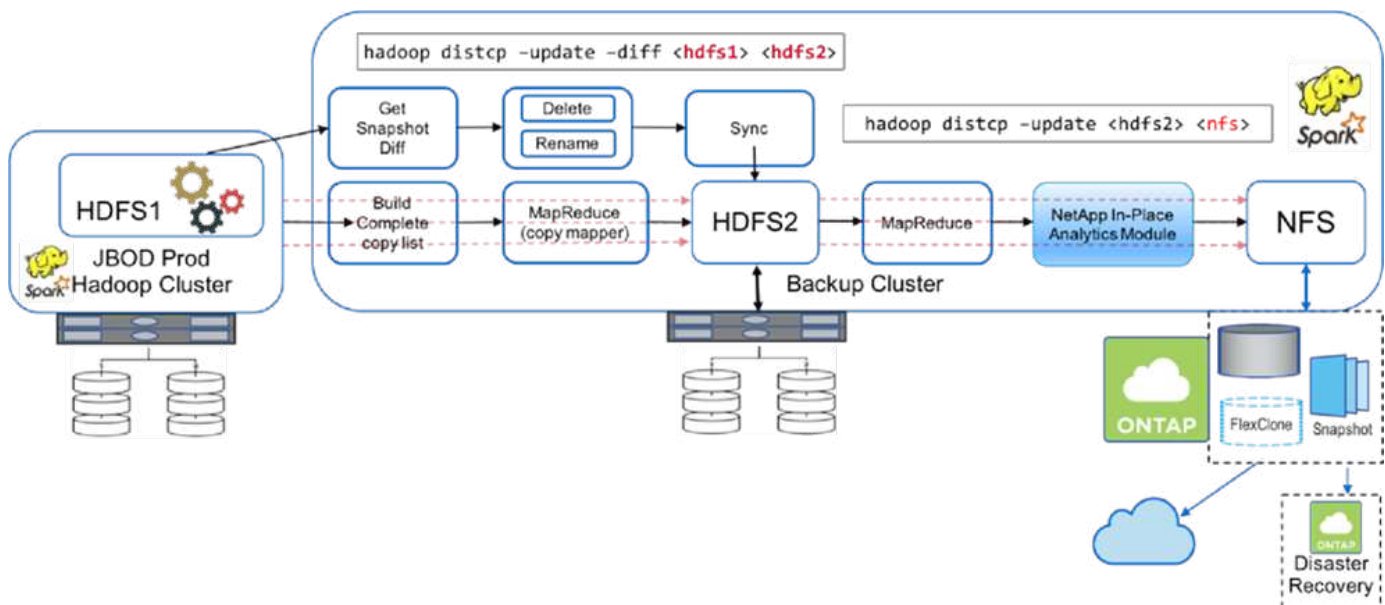
The detailed tasks for solution A include:

- The production Hadoop cluster has the customer's analytics data in the HDFS that requires protection.
- The backup Hadoop cluster with HDFS acts as an intermediate location for the data. Just a bunch of disks (JBOD) provides the storage for HDFS in both the production and backup Hadoop clusters.
- Protect the Hadoop production data is protected from the production cluster HDFS to the backup cluster HDFS by running the `Hadoop distcp -update -diff <hdfs1> <hdfs2>` command.

> **ⓘ**   The Hadoop snapshot is used to protect the data from production to the backup Hadoop cluster.

- The NetApp ONTAP storage controller provides an NFS exported volume, which is provisioned to the backup Hadoop cluster.

- By running the `Hadoop distcp` command leveraging MapReduce and multiple mappers, the analytics data is protected from the backup Hadoop cluster to NFS by using the In-Place Analytics Module.

  After the data is stored in NFS on the NetApp storage system, NetApp Snapshot, SnapRestore, and FlexClone technologies are used to back up, restore, and duplicate the Hadoop data as needed.

> **ⓘ**   Hadoop data can be protected to the cloud as well as disaster recovery locations by using SnapMirror technology.

The benefits of solution A include:

- Hadoop production data is protected from the backup cluster.
- HDFS data is protected through NFS enabling protection to cloud and disaster recovery locations.
- Improves performance by offloading backup operations to the backup cluster.
- Eliminates manual tape operations
- Allows for enterprise management functions through NetApp tools.
- Requires minimal changes to the existing environment.
- Is a cost-effective solution.

The disadvantage of this solution is that it requires a backup cluster and additional mappers to improve performance.

The customer recently deployed solution A due to its simplicity, cost, and overall performance.

In this solution, SAN disks from ONTAP can be used instead of JBOD. This option offloads the backup cluster storage load to ONTAP; however, the downside is that SAN fabric switches are required.

Solution B

Solution B adds the In-Place Analytics Module to the production Hadoop cluster, which eliminates the need for the backup Hadoop cluster, as shown in the figure below.
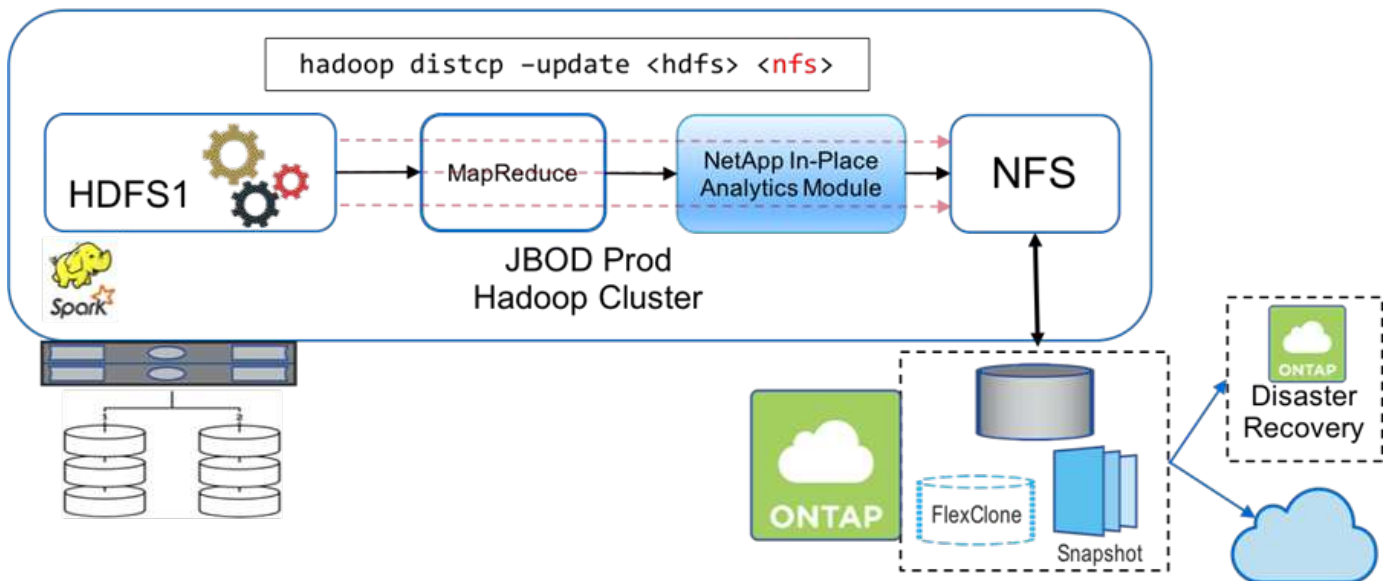
The detailed tasks for solution B include:

- The NetApp ONTAP storage controller provisions the NFS export to the production Hadoop cluster.

  The Hadoop native `hadoop distcp` command protects the Hadoop data from the production cluster HDFS to NFS through the In-Place Analytics Module.

- After the data is stored in NFS on the NetApp storage system, Snapshot, SnapRestore, and FlexClone technologies are used to back up, restore, and duplicate the Hadoop data as needed.

The benefits of solution B include:

- The production cluster is slightly modified for the backup solution, which simplifies implementation and reduces additional infrastructure cost.
- A backup cluster for the backup operation is not required.
- HDFS production data is protected in the conversion to NFS data.
- The solution allows for enterprise management functions through NetApp tools.

The disadvantage of this solution is that it's implemented in the production cluster, which can add additional administrator tasks in the production cluster.

Solution C

In solution C, the NetApp SAN volumes are directly provisioned to the Hadoop production cluster for HDFS storage, as shown in the figure below.
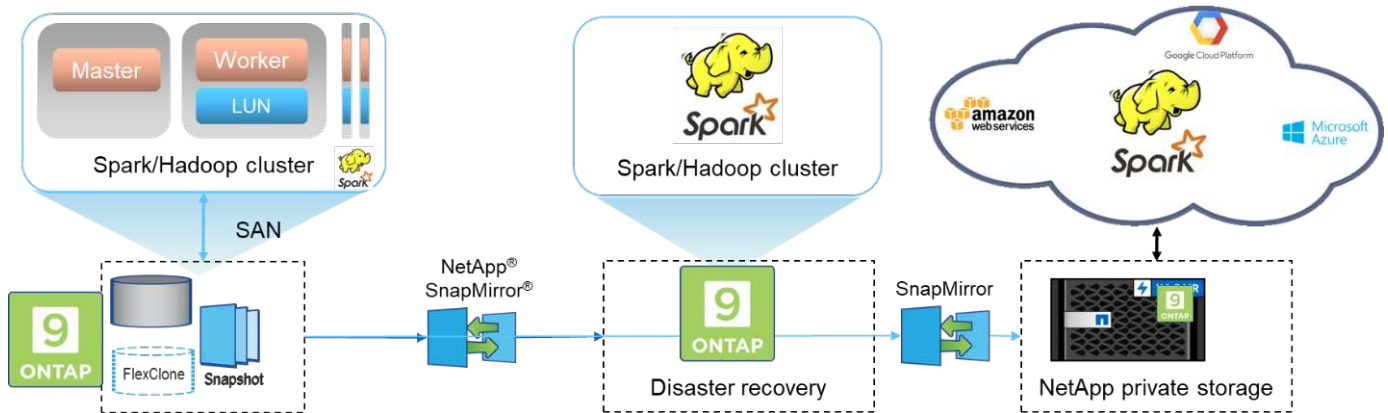
The detailed steps for solution C include:

- NetApp ONTAP SAN storage is provisioned at the production Hadoop cluster for HDFS data storage.
- NetApp Snapshot and SnapMirror technologies are used to back up the HDFS data from the production Hadoop cluster.
- There is no performance effect to production for the Hadoop/Spark cluster during the Snapshot copy backup process because the backup is at the storage layer.

> ⓘ Snapshot technology provides backups that complete in seconds regardless of the size of the data.

The benefits of solution C include:

- Space-efficient backup can be created by using Snapshot technology.
- Allows for enterprise management functions through NetApp tools.

Use case 2: Backup and disaster recovery from the cloud to on-premises

This use case is based on a broadcasting customer that needs to back up cloud-based analytics data to its on-premises data center, as illustrated in the figure below.
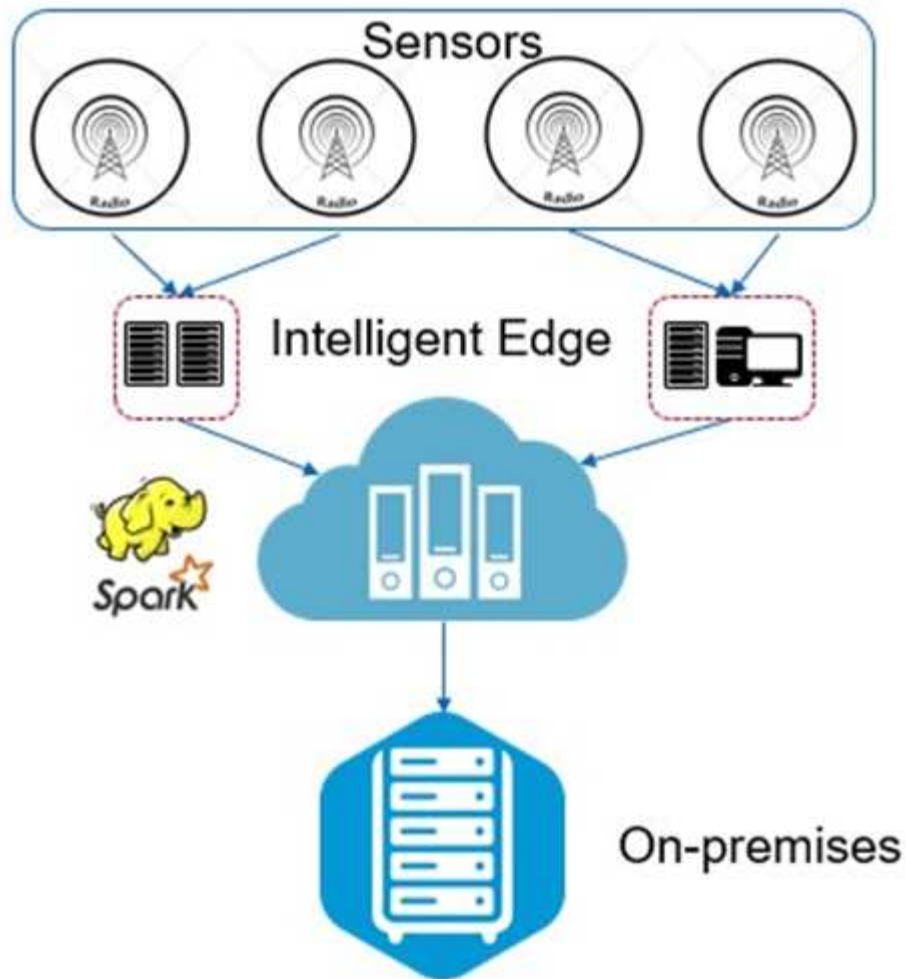
Scenario

In this scenario, the IoT sensor data is ingested into the cloud and analyzed by using an open source Apache Spark cluster within AWS. The requirement is to back up the processed data from the cloud to on-premises.

Requirements and challenges

The main requirements and challenges for this use case include:

- Enabling data protection should not cause any performance effect on the production Spark/Hadoop cluster in the cloud.
- Cloud sensor data needs to be moved and protected to on-premises in an efficient and secure way.
- Flexibility to transfer data from the cloud to on-premises under different conditions, such as on-demand, instantaneous, and during low-cluster load times.

Solution

The customer uses AWS Elastic Block Store (EBS) for its Spark cluster HDFS storage to receive and ingest data from remote sensors through Kafka. Consequently, the HDFS storage acts as the source for the backup data.

To fulfill these requirements, NetApp ONTAP Cloud is deployed in AWS, and an NFS share is created to act as the backup target for the Spark/Hadoop cluster.

After the NFS share is created, the In-Place Analytics Module is leveraged to copy the data from the HDFS EBS storage into the ONTAP NFS share. After the data resides in NFS in ONTAP Cloud, SnapMirror technology can be used to mirror the data from the cloud into on-premises storage as needed in a secure and efficient way.

This image shows the backup and disaster recovery from cloud to on-premises solution.
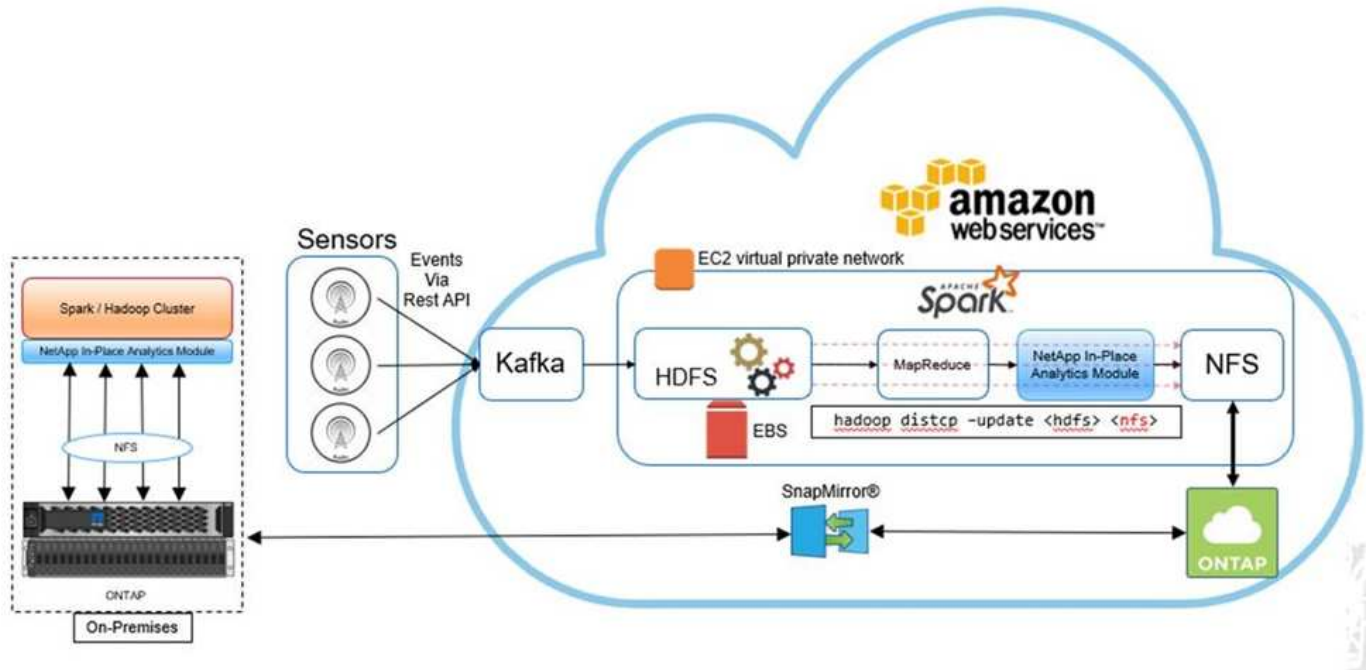
Use case 3: Enabling DevTest on existing Hadoop data

In this use case, the customer's requirement is to rapidly and efficiently build new Hadoop/Spark clusters based on an existing Hadoop cluster containing a large amount of analytics data for DevTest and reporting purposes in the same data center as well as remote locations.

Scenario

In this scenario, multiple Spark/Hadoop clusters are built from a large Hadoop data lake implementation on-premises as well as at disaster recovery locations.

Requirements and challenges

The main requirements and challenges for this use case include:

- Create multiple Hadoop clusters for DevTest, QA, or any other purpose that requires access to the same production data. The challenge here is to clone a very large Hadoop cluster multiple times instantaneously and in a very space-efficient manner.
- Sync the Hadoop data to DevTest and reporting teams for operational efficiency.
- Distribute the Hadoop data by using the same credentials across production and new clusters.

- Use scheduled policies to efficiently create QA clusters without affecting the production cluster.

Solution

FlexClone technology is used to answer the requirements just described. FlexClone technology is the read/write copy of a Snapshot copy. It reads the data from parent Snapshot copy data and only consumes additional space for new/modified blocks. It is fast and space-efficient.

First, a Snapshot copy of the existing cluster was created by using a NetApp consistency group.

Snapshot copies within NetApp System Manager or the storage admin prompt. The consistency group Snapshot copies are application-consistent group Snapshot copies, and the FlexClone volume is created based on consistency group Snapshot copies. It is worth mentioning that a FlexClone volume inherits the parent volume's NFS export policy. After the Snapshot copy is created, a new Hadoop cluster must be installed for DevTest and reporting purposes, as shown in the figure below. The In-Place Analytics Module accesses the cloned NFS volume from the new Hadoop cluster through In-Place Analytics Module users and group authorization for the NFS data.

To have proper access, the new cluster must have the same UID and GUID for the users configured in the In-Place Analytics Module users and group configurations.

This image shows the Hadoop cluster for DevTest.
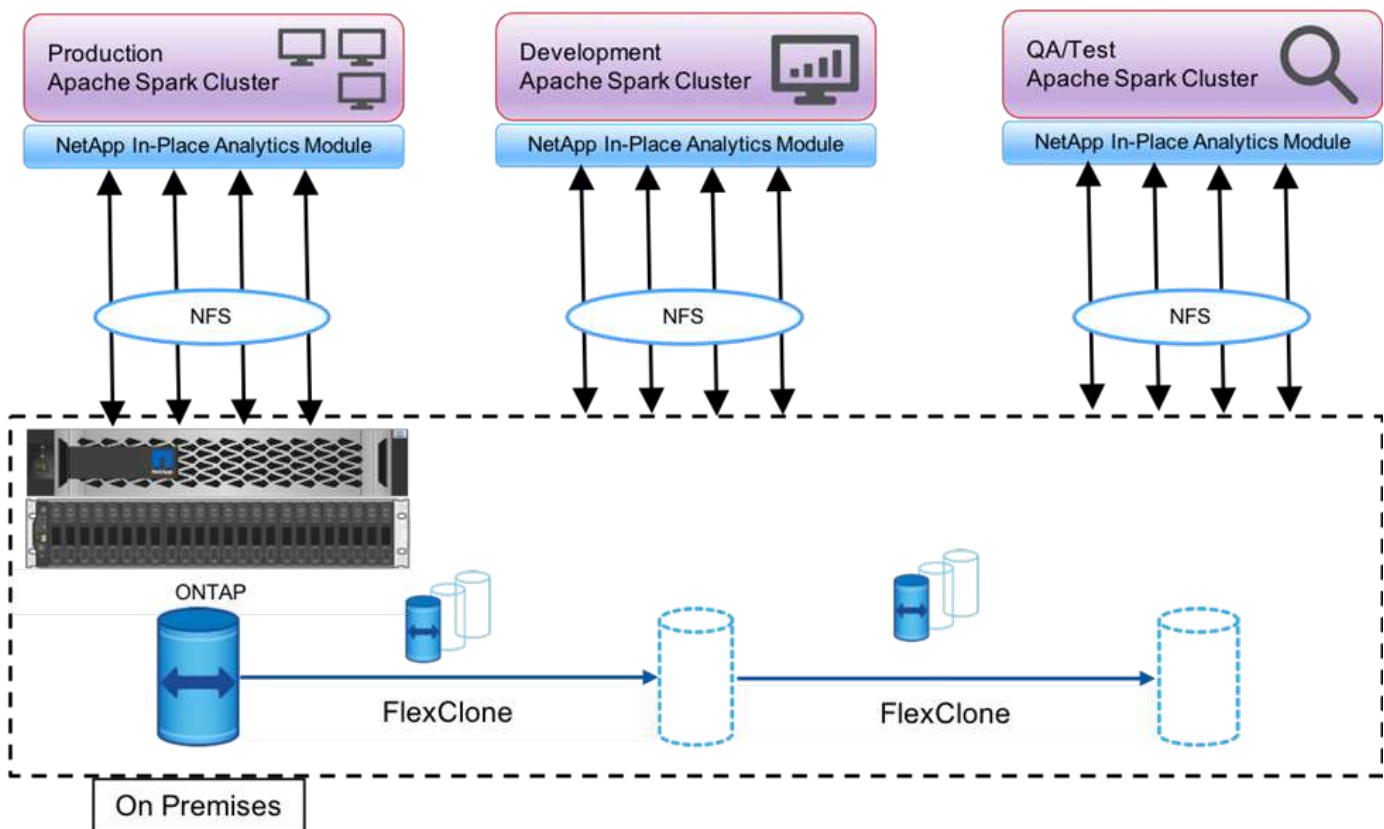
Use case 4: Data protection and multicloud connectivity

This use case is relevant for a cloud service partner tasked with providing multicloud connectivity for customers' big data analytics data.

Scenario

In this scenario, IoT data received in AWS from different sources is stored in a central location in NPS. The NPS storage is connected to Spark/Hadoop clusters located in AWS and Azure enabling big data analytics applications running in multiple clouds accessing the same data.

Requirements and challenges

The main requirements and challenges for this use case include:

- Customers want to run analytics jobs on the same data using multiple clouds.
- Data must be received from different sources such as on-premises and cloud through different sensors and hubs.
- The solution must be efficient and cost-effective.
- The main challenge is to build a cost-effective and efficient solution that delivers hybrid analytics services between on-premises and across different clouds.

Solution

This image illustrates the data protection and multicloud connectivity solution.
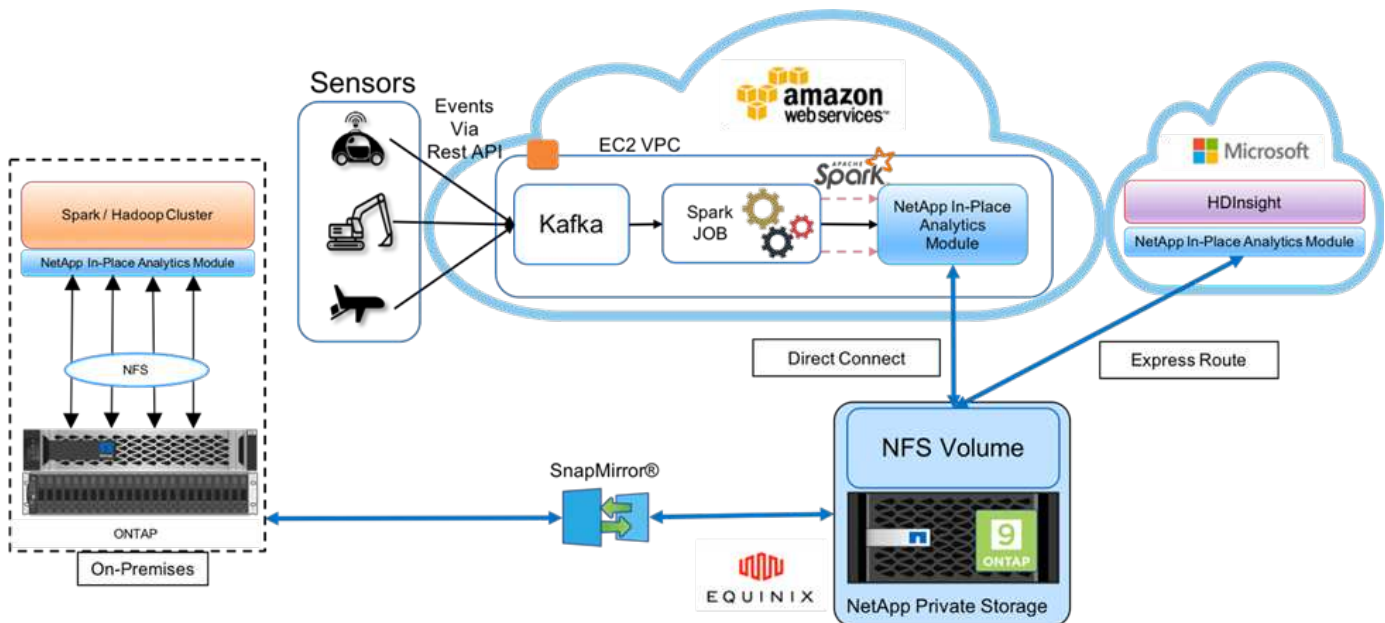
As shown in the figure above, data from sensors is streamed and ingested into the AWS Spark cluster through Kafka. The data is stored in an NFS share residing in NPS, which is located outside of the cloud provider within an Equinix data center. Because NetApp NPS is connected to Amazon AWS and Microsoft Azure through Direct Connect and Express Route connections, respectively, customers can leverage the In-Place Analytics Module to access the data from both Amazon and AWS analytics clusters. This approach solves having cloud analytics across multiple hyperscalers.

Consequently, because both on-premises and NPS storage runs ONTAP software, SnapMirror can mirror the NPS data into the on-premises cluster, providing hybrid cloud analytics across on-premises and multiple clouds.

For the best performance, NetApp typically recommends using multiple network interfaces and direct connection/express routes to access the data from cloud instances.

Use case 5: Accelerate analytic workloads

In this scenario, a large financial services and investment bank's analytics platform was modernized using the NetApp NFS storage solution to achieve significant improvement in analyzing investment risks and derivatives for its asset management and quantitative business unit.
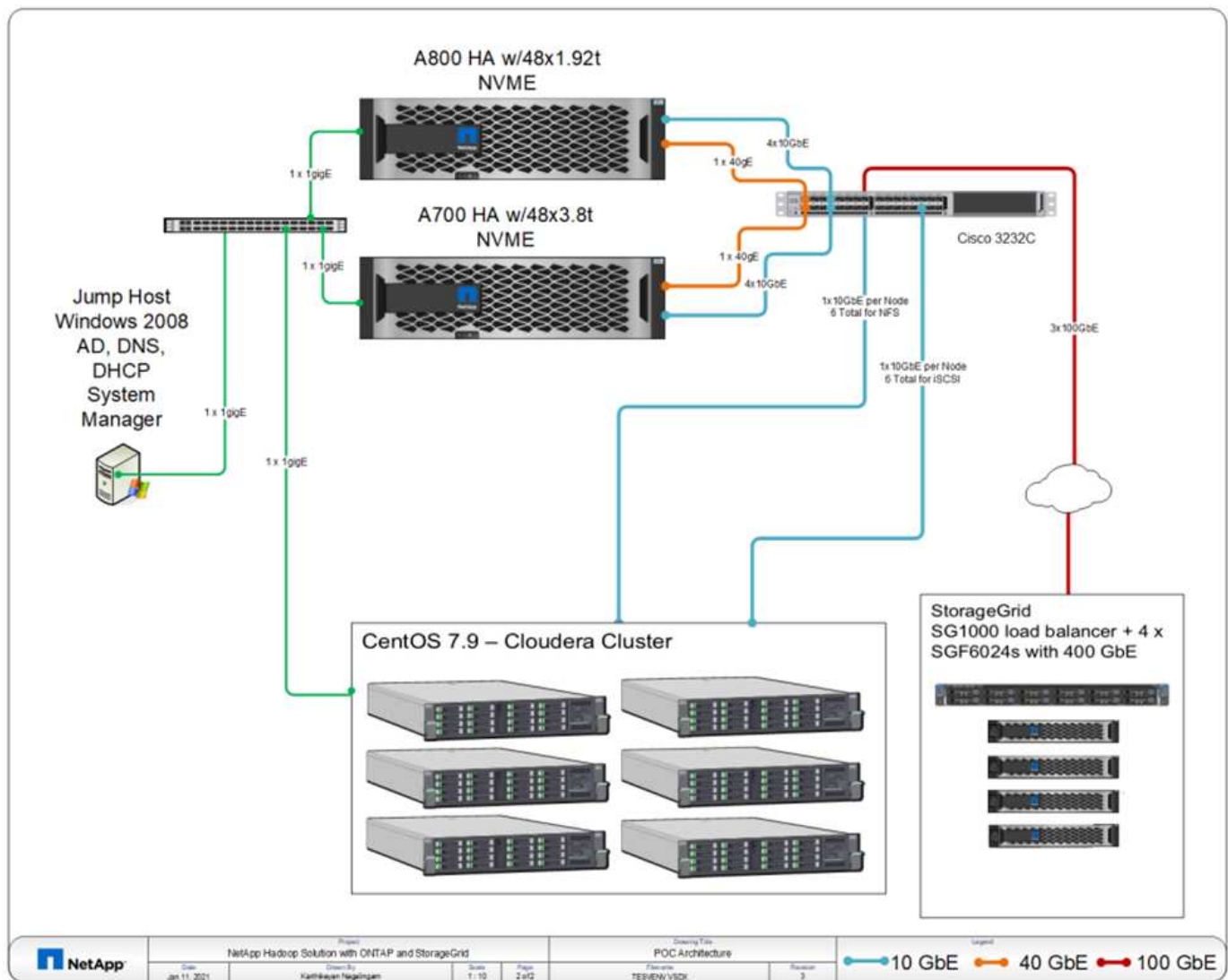
Scenario

In the customer's existing environment, the Hadoop infrastructure used for the analytics platform leveraged internal storage from the Hadoop servers. Due to proprietary nature of JBOD environment, many internal customers within the organization were unable to take advantage of their Monte Carlo quantitative model, a simulation that relies on the recurring samples of real-time data. The suboptimal ability to understand the effects of uncertainty in market movements was serving unfavorably for the quantitative asset management business unit.

Requirements and challenges

The quantitative business unit at the bank wanted an efficient forecasting method to attain accurate and timely predictions. To do so, the team recognized the need to modernize the infrastructure, reduce existing I/O wait time and improve performance on the analytic applications such as Hadoop and Spark to efficiently simulate investment models, measure potential gains and analyze risks.

Solution

The customer had JBOD for their existing Spark solution. NetApp ONTAP, NetApp StorageGRID, and MinIO Gateway to NFS was then leveraged to reduce the I/O wait time for the bank's quantitative finance group that runs simulation and analysis on investment models that assess potential gains and risks. This image shows the Spark solution with NetApp storage.

A800 HA w/48x1.92t
NVME

A700 HA w/48x3.8t
NVME

Jump Host
Windows 2008
AD, DNS,
DHCP
System
Manager

Cisco 3232C

4x 10GbE
1 x 40gE
1 x 40gE
4x 10GbE

1x 10GbE per Node
6 Total for NFS

1x 10GbE per Node
6 Total for iSCSI

3x 100GbE

CentOS 7.9 – Cloudera Cluster

StorageGrid
SG1000 load balancer + 4 x
SGF6024s with 400 GbE

NetApp	Project NetApp Hadoop Solution with ONTAP and StorageGrid			Drawing Title POC Architecture		Legend 10 GbE 40 GbE 100 GbE
	Date Jan 11, 2021	Drawn By Karthikeyan Nagalingam	Scale 1:10	Page 2 of 2	Filename TEBVENV VSDX	Revision 3

As shown in figure above, AFF A800, A700 systems, and StorageGRID were deployed to access parquet files through NFS and S3 protocols in a six-node Hadoop cluster with Spark, and YARN and Hive metadata services for data analytic operations.

A direct-attached storage (DAS) solution in the customer's old environment had the disadvantage to scale compute and storage independently. With NetApp ONTAP solution for Spark, the bank's financial analytics business unit was able to decouple storage from compute and seamlessly bring infrastructure resources more effectively as needed.

By using ONTAP with NFS, the compute server CPUs were almost fully utilized for Spark SQL jobs and the I/O wait time was reduced by nearly 70%, therefore providing better compute power and performance boost to Spark workloads. Subsequently, increasing CPU utilization also enabled the customer to leverage GPUs, such as GPUDirect, for further platform modernization. Additionally, StorageGRID provides a low-cost storage option for Spark workloads and MinIO Gateway provides secure access to NFS data through the S3 protocol. For data in the cloud, NetApp recommends Cloud Volumes ONTAP, Azure NetApp Files, and NetApp Cloud Volumes Service.

Conclusion

This section provides a summary of the use cases and solutions provided by NetApp to fulfill various Hadoop data protection requirements. By using the data fabric powered by NetApp, customers can:

- Have the flexibility to choose the right data protection solutions by leveraging NetApp's rich data management capabilities and integration with Hadoop native workflows.
- Reduce their Hadoop cluster backup window time by almost 70%.
- Eliminate any performance effect resulting from Hadoop cluster backups.
- Provide multicloud data protection and data access from different cloud providers simultaneously to a single source of analytics data.
- Create fast and space-efficient Hadoop cluster copies by using FlexClone technology.

Where to find additional information

To learn more about the information described in this document, see the following documents and/or websites:

- NetApp Big Data Analytics Solutions

  https://www.netapp.com/us/solutions/applications/big-data-analytics/index.aspx

- Apache Spark Workload with NetApp Storage

  https://www.netapp.com/pdf.html?item=/media/26877-nva-1157-deploy.pdf

- NetApp Storage Solutions for Apache Spark

  https://www.netapp.com/media/16864-tr-4570.pdf

- Apache Hadoop on data fabric enabled by NetApp

  https://www.netapp.com/media/16877-tr-4529.pdf

- NetApp In-Place Analytics Module

  https://library.netapp.com/ecm/ecm_download_file/ECMLP2854071

Acknowledgements

- Paul Burland, Sales Rep, ANZ Victoria District Sales, NetApp
- Hoseb Dermanilian, Business Development Manager, NetApp
- Lee Dorrier, Director MPSG, NetApp
- David Thiessen, Systems Engineer, ANZ Victoria District SE, NetApp

Version history

Version	Date	Document version history
Version 1.0	January 2018	Initial release

Version	Date	Document version history
Version 2.0	October 2021	Updated with use case #5: Accelerate analytic workload