v1.7 | June 2022 | RA-2019

REFERENCE ARCHITECTURE

Splunk on Nutanix

NUTANIX™
YOUR ENTERPRISE CLOUD

Copyright

Contents

1. Executive Summary

A scale-out application like Splunk demands a scale-out infrastructure like Nutanix to grow along with it. This document makes recommendations for designing, optimizing, and scaling Splunk deployments on Nutanix. It shows the scalability of Nutanix and provides detailed performance and configuration information on the scale-out capabilities of the cluster when used for classic Splunk and Splunk with SmartStore deployments.

For some time now the Nutanix distributed scale-out model has been the standard for big data analytics workloads because its ability to scale both compute and storage on demand simplifies footprint expansion in large-scale deployments. For deployment scenarios with an asymmetric demand for storage over compute, the need to disaggregate both components arises. Nutanix allows storage and compute to scale independently of one another, which is more efficient and more cost effective.

With Objects, Nutanix provides both the underlying distributed compute platform and the S3-compliant storage layer for a disaggregated deployment of Splunk and Splunk SmartStore. The provision of both solution layers delivers a high degree of compute and storage elasticity and incredibly cost-efficient long-term data retention at large scale.

Splunk SmartStore is built around the same distributed scale-out model as Nutanix and dynamically places data in local storage, in remote storage, or both, depending on access patterns, data age, and data or index priority. SmartStore uses an AWS (Amazon Web Services) S3 API (application programming interface) to plug into the remote storage tier. Remote storage options are S3 API–compliant object stores.

By decoupling compute and storage you increase resource utilization and flexibility and lower costs. You can then focus budget as required on a high-performance compute tier, where you can invest in new or novel solid-state devices for your hot and recently cached warm Splunk buckets, or separately invest in ultra-dense nodes that can address long-term retention needs with remote storage backed by a Nutanix Objects cluster.

Nutanix Objects software is deployed on a microservices architecture in the Nutanix platform, which allows you to start small and scale to petabytes with billions of objects. Built-in load balancer VMs ensure the even distribution of the workload across the worker

VMs, which provision the S3 management interface. You can define life-cycle policies that allow you to protect, expire, and even tier your data to secondary S3 storage.

The S3 API manages all object data in a global namespace. Typically, to read data, you issue a GET request, and to write data, you issue a PUT request. Nutanix provides an objects browser that allows you to view all your data at any time.

You can further protect your data with additional features like versioning and WORM (write once, read many) settings. The WORM paradigm protects your data against all writes, updates, and deletes for a set period, removing additional vectors against ransomware-style attacks.

Nutanix Objects also supports bidirectional replication between Object stores. Replication happens nearly synchronously (NearSync) and provides an RPO (recovery point objective) of just seconds, which you can configure and monitor in Prism Central. You can drive all workflows using either the GUI or an API that allows you to programmatically create, manage, and query an object store and its bucket storage.

Table: Tested Solution Details

Product Name	Product Version	Nutanix AOS Version	Hypervisor	Hypervisor Version
Splunk Enterprise	8.2.1+	5.15+ LTS	AHV	20170830.301+

2. Introduction

Audience

We wrote this reference architecture for those who design, manage, and support Nutanix infrastructures. Consumers of this document should already be familiar with Splunk, virtualization, and the Nutanix platform.

We have organized this document to address key items for enabling successful design, implementation, and transition to operation.

Purpose

This document covers the following subject areas:

- Overview of the Nutanix solution.
- Overview of Splunk and its use cases.
- The benefits of Splunk on Nutanix.
- Recommendations for architecting a complete Splunk solution on the Nutanix platform, including design and configuration considerations.
- Benchmarks for Splunk performance on Nutanix.

Document Version History

Version Number	Published	Notes
1.0	July 2015	Original publication.
1.1	July 2016	Updated platform overview.

Version Number	Published	Notes
1.2	May 2017	Added consideration of multisite deployments, high availability, customer use cases, and Splunk Enterprise Security sizing.
1.3	June 2018	Updated platform overview.
1.4	August 2019	Minor updates throughout.
1.5	February 2021	Minor updates throughout.
1.6	September 2021	Updated Executive Summary, Provides a Platform for Enterprise Apps, Deployment Options, Validation and Benchmarking, Interpreting the Results, and Results sections, added SmartStore Sizing and Best Practices for Running Splunk on Nutanix sections, and updated the Tested Solution Details and Splunk Indexer VM Sizing tables.
1.7	May 2022	Refreshed content.

3. Nutanix Objects

Nutanix Objects is an object storage service designed to solve the problem of unanticipated data growth—specifically, unstructured data that could start small then grow at a rapid pace, potentially reaching the petabyte scale. Storing unstructured data using traditional block and file data management protocols can result in unmanageably large, complex, and expensive solutions.

Nutanix Objects runs on Nutanix—the industry's most popular hyperconverged solution—natively converging compute and storage into a turnkey appliance you can deploy in minutes to run any application out of the box. With Nutanix Objects, you can start small and scale to store petabyte bursts of new data while still maintaining the benefits of a distributed system, such as flexibility, scalability, and cost effectiveness. The Nutanix solution offers powerful virtualization capabilities that focus on adding value to organizations by providing a robust but easy-to-use product. Simplicity is vital to productivity, and Nutanix Objects users have the freedom to deploy object storage alongside block and file storage for ease of management and transparency because the different storage types are integrated with the same infrastructure stack.

Nutanix Objects integrates with the solution stack through services that run inside the Prism Central VM and manage all the other object storage components and services. The Nutanix cluster deploys worker VMs to handle the multiple components that provide the object storage API and lookups for the objects. These components run as containerized services in a Kubernetes cluster. Objects follows a modular, scale-out design where each component focuses on a single core function, so you can scale out any component independently to match workload demands.

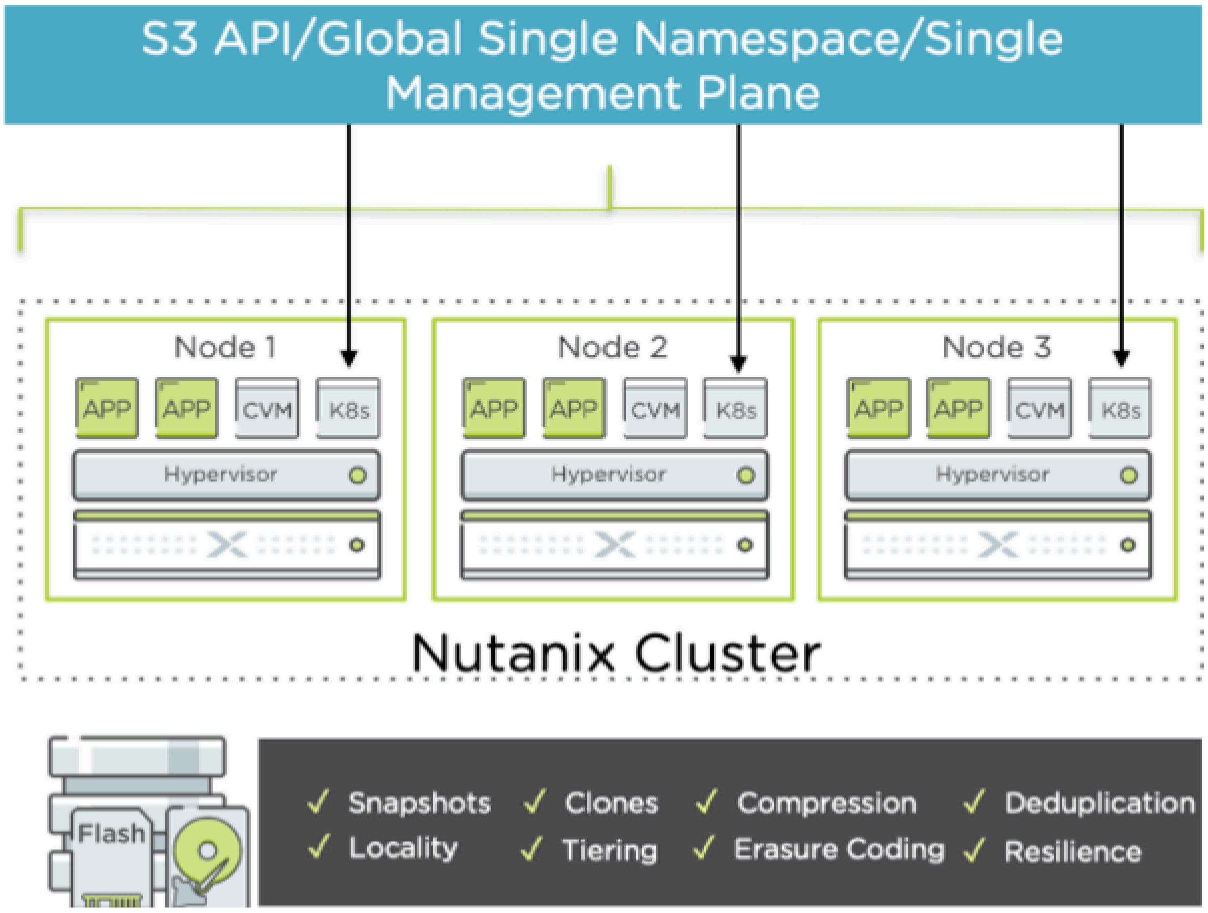


Figure 1: S3-Compatible Object Storage Integrated with Nutanix

4. Splunk Architecture

Indexes Any Data from Any Source

Splunk Enterprise collects and indexes machine-generated data from virtually any source, format, or location in real time. This capability includes data streaming from packaged and custom applications, app servers, web servers, databases, networks, VMs, telecom equipment, operating systems, sensors, and much more. There's no requirement for Splunk Enterprise to "understand" the data up front; it immediately starts collecting and indexing your data so you can start searching and analyzing.

You can find additional product information online.

Forwards Data from Remote Systems

Splunk forwarders are useful in situations where the data you need isn't visible over the network. They deliver reliable, secure, real-time data collection for up to tens of thousands of sources. Splunk forwarders can monitor local application logfiles, capture status command output on a schedule, grab performance metrics from virtual or nonvirtual sources, or watch the file system for configuration, permissions, and attribute changes. They are also lightweight, and you can deploy them quickly at no additional cost.

Correlates Complex Events

Splunk Enterprise enables you to correlate complex events spanning many diverse data sources across your environment. Correlation types include:

- Time-based correlations, which identify relationships based on time, proximity, or distance.
- Transaction-based correlations, which track a series of related events as a single transaction to measure duration, status, and other metrics.
- Subsearches, which take the results of one search and use them in another.

- Lookups, which correlate data with external data sources outside Splunk.
- Joins, which support SQL-like inner and outer joins.

Delivers Enterprise-Class Scale, Resilience, and Interoperability

Splunk Enterprise scales to collect and index dozens of terabytes of data per day. Because the insights drawn from your data are mission-critical, Splunk software's index replication technology provides the availability you need, even as you scale out your distributed computing environment. Automatic load balancing optimizes workloads and response times and provides built-in failover support. Out-of-the-box reporting and analytics capabilities deliver rapid insights from your data.

Splunk DB Connect delivers reliable, scalable, real-time integration between Splunk and traditional relational databases. Splunk Hadoop Connect provides bidirectional integration so you can move data easily and reliably between Splunk Enterprise and Hadoop.

Provides a Platform for Enterprise Apps

Developer teams can leverage Splunk Enterprise in many ways. You can debug and troubleshoot applications during development and test cycles or integrate data from Splunk Enterprise into custom applications. With API versioning, you can output data from any API endpoint in JSON and ensure custom Splunk development over time. Splunk Enterprise ships with the JavaScript SDK as well as additional downloadable SDKs for Java, Python, and PHP, making it easy to customize and extend.

Deployment Scenario: Single Instance to Multi-Indexer Cluster

In single-machine deployments, one instance of Splunk handles the entire end-to-end process, from data input, through indexing, to search. You can use a single-machine deployment for testing and evaluation purposes and perhaps to serve the needs of department-sized environments. For larger environments, however, where data originates on many machines and where many users need to search the data, administrators can distribute functionality across multiple Splunk instances. These larger deployments can also decouple compute and storage using Splunk SmartStore capabilities with Nutanix Objects.

In a typical midsize deployment, for example, you can deploy lightweight versions of Splunk, called forwarders, on the machines where the data originates. The forwarders consume data locally, then advance the data across the network to another Splunk component called the indexer. The indexer does the heavy lifting—it indexes the data and runs searches—and should reside on a machine by itself. The forwarders, on the other hand, can easily coexist on the machines generating the data because the data-consuming function has minimal impact on machine performance.

As you scale up, you can add more forwarders and indexers. A large deployment may have hundreds of forwarders sending data to a few indexers. You can use load balancing on the forwarders so that they distribute their data across some or all indexers. Load balancing helps with scaling and provides a failover capability if one of the indexers goes down because the forwarders automatically send their data to the indexers that remain.

Deployment Scenario: Search Head Cluster

A search head cluster consists of a group of search heads that share configurations, job scheduling, and search artifacts. The search heads are known as the cluster members.

One cluster member has the role of captain, which means it coordinates job scheduling and replication activities among all the members. It also serves as a search head like any other member, running search jobs, serving results, and so on. Over time, the role of captain can shift among the cluster members.

For more information, see the Splunk Enterprise Distributed Deployment Manual.

Deployment Scenario: Multisite Indexer Cluster

A multisite indexer cluster is essentially an indexer cluster that spans multiple physical sites—datacenters, for example. Each site has its own set of peer nodes and search heads. Each site also obeys site-specific replication and search factor rules, providing multisite awareness and delivering improved disaster recovery and search head affinity. The main differences between multisite clusters and single-site deployments are that, in a multisite configuration:

- Each node belongs to an assigned site.
- Bucket copy replication is site-aware.
- Search heads distribute their searches across only local peers whenever possible.

- Multisite indexer configurations can use Splunk SmartStore. Both underlying Objects stores can replicate data between sites to protect against site or remote object store outage.

Bear in mind that you can migrate an indexer cluster from a single site to a multisite configuration. However, after such a migration, the cluster holds both single-site and multisite buckets. It maintains these buckets separately, following these rules:

- Single-site buckets (those existing at the time of migration) continue to respect their single-site replication and search factors. You cannot convert them to multisite.
- Multisite buckets (those created after migration) follow the multisite replication and search factor policies.

If there is a chance that you may wish to deploy across multiple sites in the future, we recommend configuring the indexer cluster to be site-aware from the outset. Consult your local Splunk representative regarding such installations. For further details regarding multisite indexer clusters, refer to Splunk’s multisite indexer cluster deployment overview.

Deployment Scenario: Splunk SmartStore

You can also use SmartStore to decouple compute and storage in all the previous scenarios. SmartStore is an indexer capability that enables you to use remote object stores, such as Nutanix Objects, 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 in a cost-effective manner by scaling those resources separately.

SmartStore introduces a remote storage tier and a cache manager, which allow data to reside either locally on indexers or on the remote storage tier. Data movement between the indexer and the remote storage tier is managed by the cache manager, which resides 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 remote storage, while 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.

You can enable SmartStore for all indexes or for a subset of indexes.

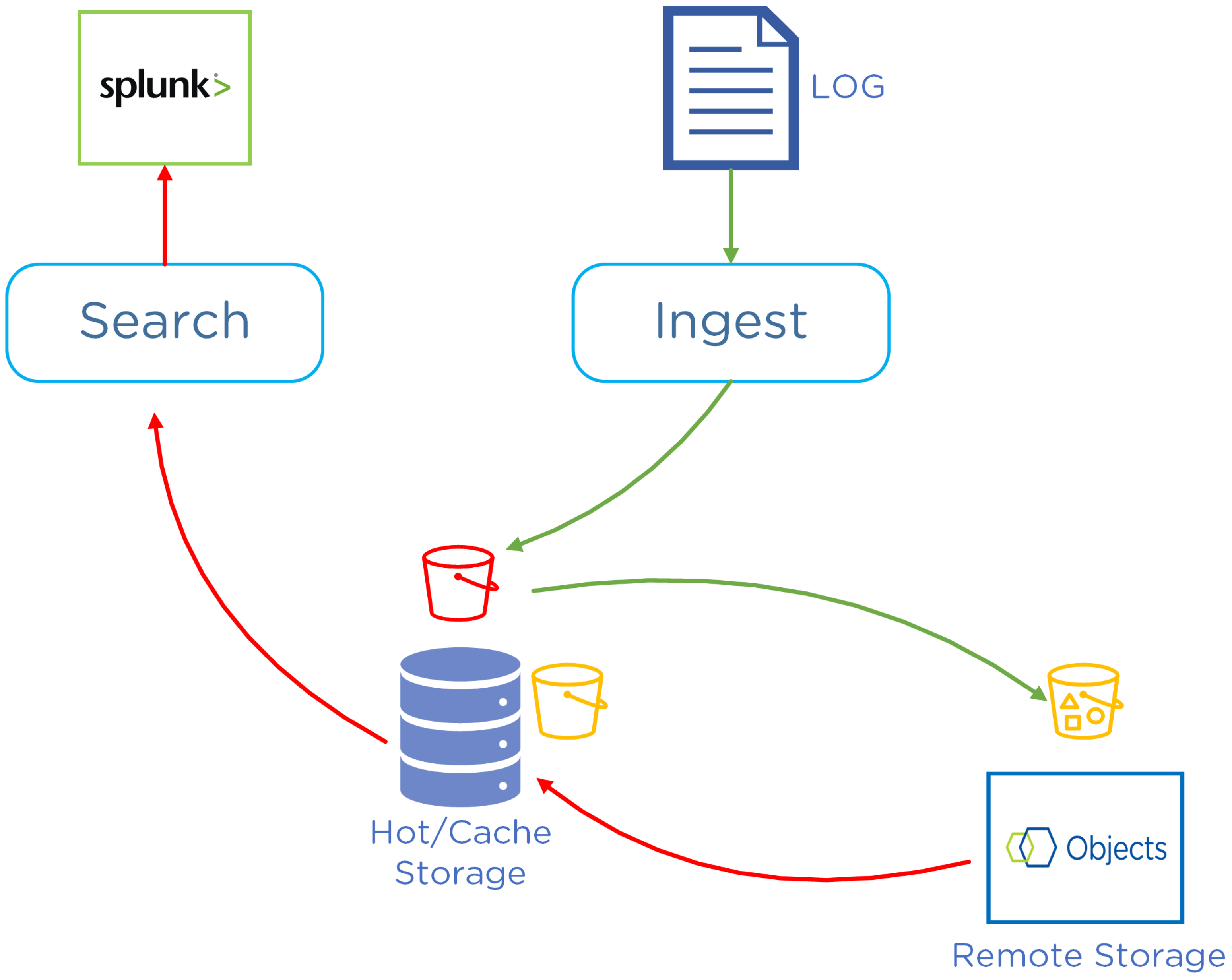


Figure 2: Splunk SmartStore Deployment Overview

5. Using Splunk on Nutanix

The Nutanix platform operates and scales Splunk along with other hosted services, providing a single scalable infrastructure for all deployments. Existing sources can send machine data to the Splunk environment on Nutanix over the network. The following figure shows a high-level view of the Splunk on Nutanix solution.

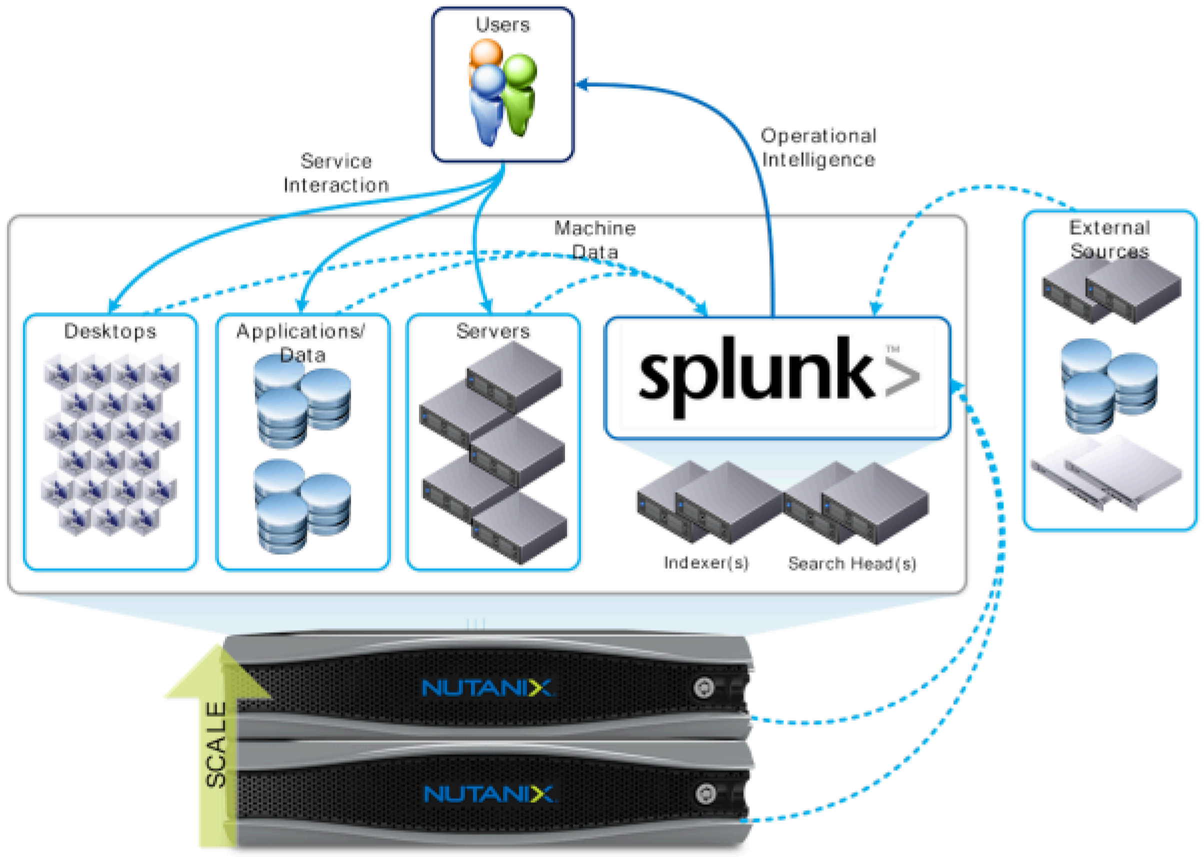


Figure 3: Splunk + Nutanix High-Level Architecture

Why Run Splunk on Nutanix?

Nutanix enables you to run multiple workloads on the same scalable converged infrastructure.

Modular incremental scale

With the Nutanix solution, you can start small and scale. A single Nutanix block (up to four nodes) provides from 20 TB to 240 TB or more of storage and up to 176 cores in a compact footprint. Given the modularity of the solution, you can scale one node at a time, giving you the ability to accurately match supply with demand and minimize the upfront capex.

High performance

The Nutanix solution provides 250,000 or more random read IOPS and up to 5 GB per second of sequential throughput in a compact 2RU (rack unit) cluster. Performance scales linearly as you add new nodes to the cluster.

Data efficiency

The Nutanix solution is truly VM-centric for all compression policies. Unlike traditional solutions that perform compression mainly at the LUN level, the Nutanix solution provides these transformations at the VM and file levels, greatly increasing efficiency and simplicity. These capabilities ensure the highest possible compression and decompression performance on a subblock level. By allowing for both inline and post-process compression, the Nutanix solution breaks the bounds set by traditional data efficiency solutions.

High-density architecture

Nutanix uses an advanced server architecture that integrates 8 Intel CPU (up to 176 cores) and up to 2 TB of memory in a single 2RU appliance. Coupled with data archiving and compression, Nutanix can reduce hardware footprint by as much as 400 percent.

Effective information life-cycle management

The Nutanix architecture supports local disks attached to the CVM (SSD, HDD) as well as remote (NAS) and cloud-based source targets. The tiering logic is fully extensible, allowing you to add and extend new tiers dynamically.

Data Tiering and Management

Splunk has a built-in mechanism to age index data automatically as it grows stale. An index consists of multiple directories, or buckets. These buckets function as data storage mechanisms and progress (or roll) through various stages as the data ages.

Splunk writes all newly indexed data to a searchable hot bucket and, after the bucket reaches a particular size, rolls that data to a warm bucket. Splunk doesn't actively write to warm buckets; warm buckets do, however, remain searchable. As buckets roll to the warm stage, the oldest warm buckets roll to the cold stage. After a set period, the cold buckets roll to frozen, where they are either archived or deleted.

AOS storage has a built-in information life-cycle management (ILM) process that complements Splunk's temporal bucket mechanism. The following figure shows the I/O path and how fingerprinting and the content cache are integrated.

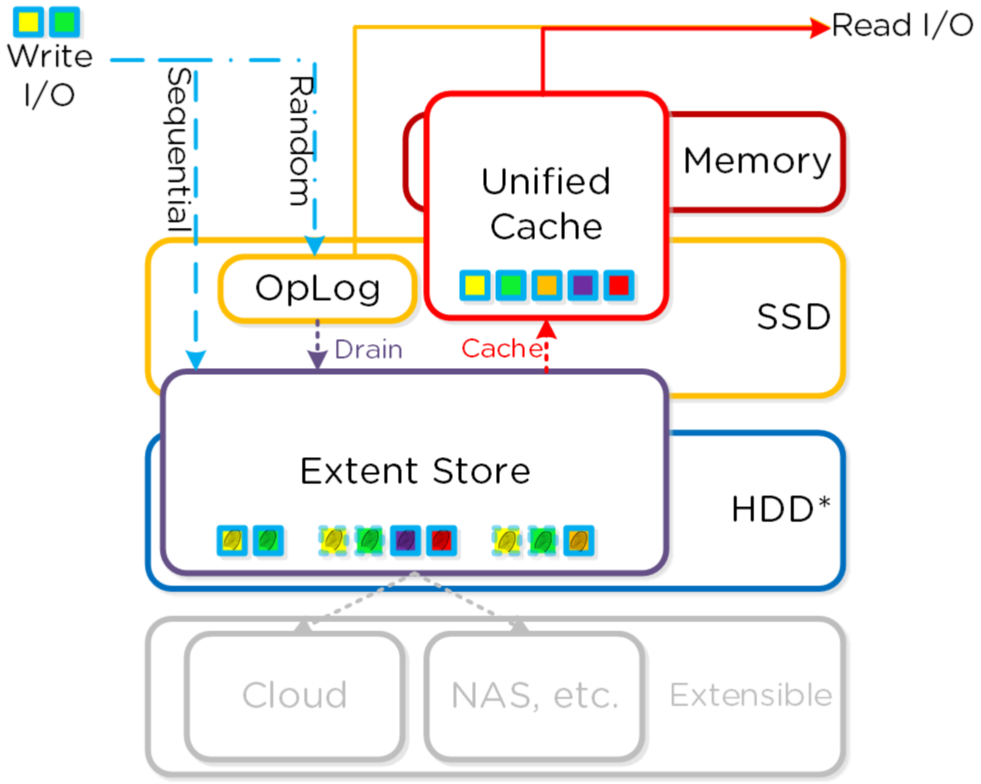


Figure 4: I/O Path, Fingerprinting, and the Content Cache

Just as Splunk rolls data from hot, to warm, and then to cold and frozen tiers, Nutanix ILM automatically detects that data is cooling down and migrates that data from the SSD to the higher-capacity HDD tier to free SSD space for new hot data. You also have the option to configure this data to bypass the SSD tier automatically and write directly to the HDD. The system can automatically compress this data (above and beyond native Splunk compression) after a specified period to increase storage capacity. If users access the compressed cold data, the system automatically decompresses it and places it on the appropriate tier.

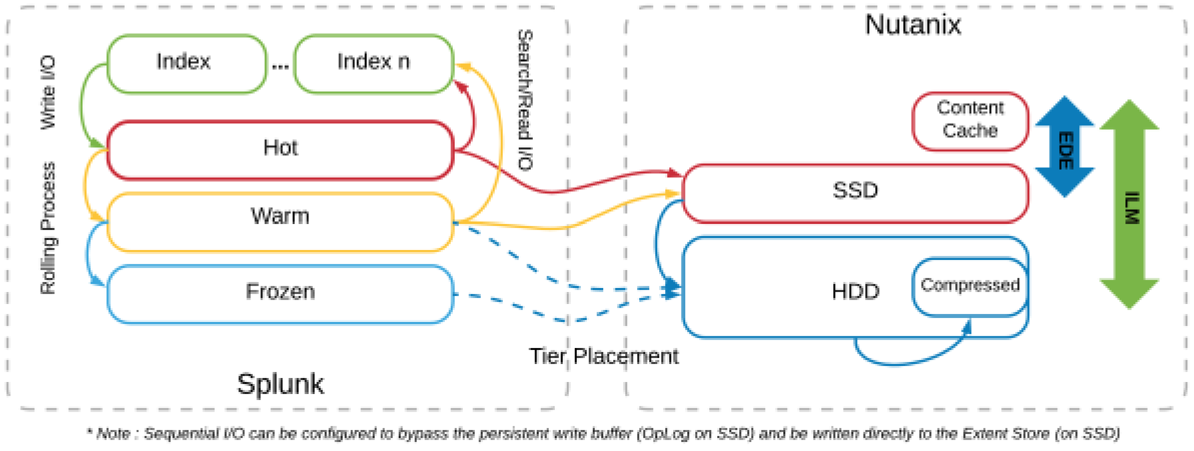
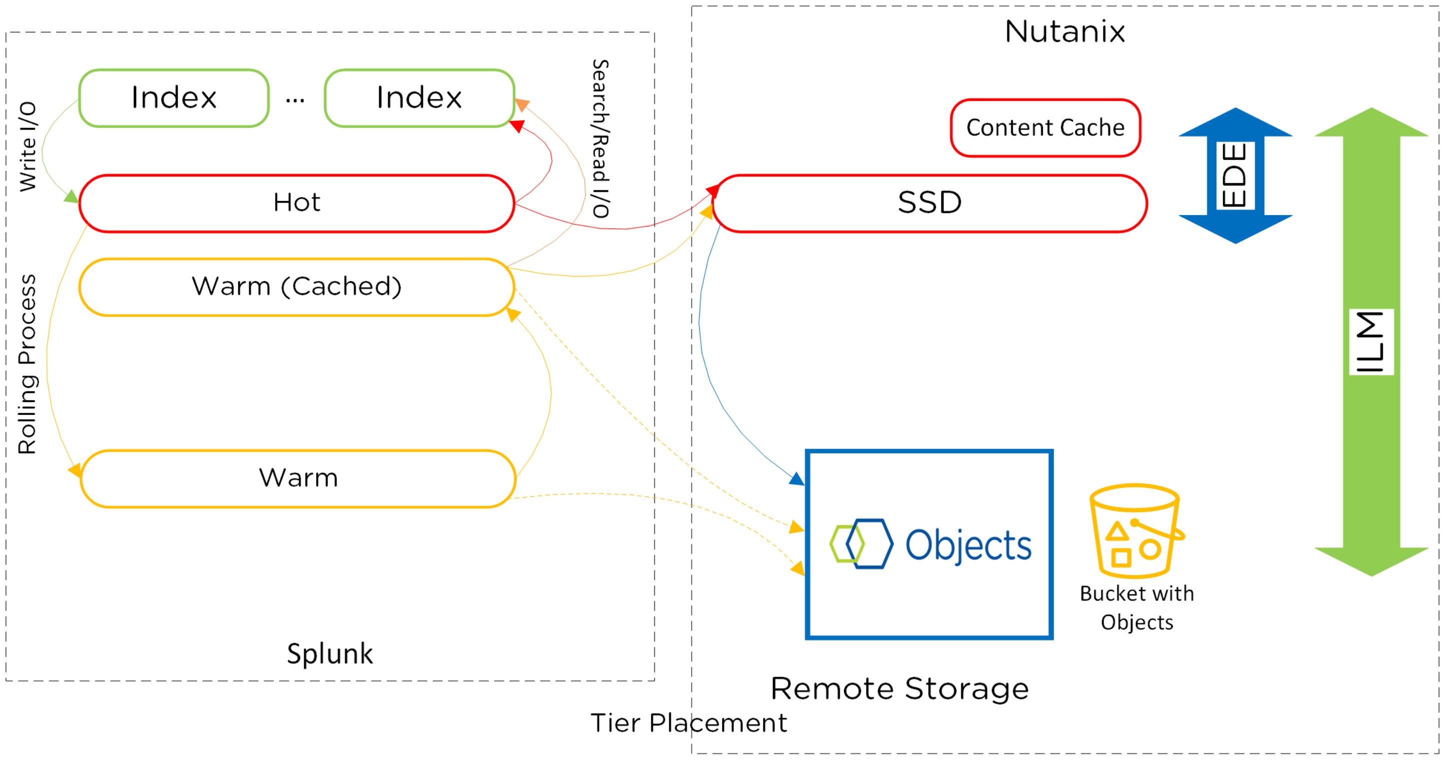

Nutanix
Index
...
Index n
Write I/O
Rolling Process
Search/Read I/O
Hot
Warm
Frozen
Content Cache
EDE
ILM
SSD
HDD
Compressed
Tier Placement
Splunk
* Note : Sequential I/O can be configured to bypass the persistent write buffer (OpLog on SSD) and be written directly to the Extent Store (on SSD)

Figure 5: Mapping Splunk Index Buckets to Nutanix ILM



Note: Sequential I/O can be configured to bypass the persistent write buffer (OpLog on SSD) and be written directly to the Extent Store (on SSD)

Figure 6: Mapping Splunk SmartStore Index buckets to Nutanix ILM

AOS distributed fabric makes a large pool of clustered storage available to VMs running on the Nutanix platform. Given Splunk's bucketed tiering, we recommend that you use Nutanix nodes with high-performance storage for the hot and warm tiers and Nutanix nodes with high-capacity storage for the cold Splunk bucket data. High-capacity storage nodes can be a separate Nutanix Objects cluster to hold cold tier Splunk data in SmartStore scenarios.

Note: The Splunk VM only needs a single mount point configured for all buckets.

This arrangement simplifies configuration and management, as the Nutanix platform automatically handles bucket placement without multiple storage platforms or clusters.

Compression

Nutanix recommends post-process compression with at least a 48-hour delay for Splunk on Nutanix deployments. When Splunk ingests data into the indexers, it uses transient files to collect data in 128 KB slices, then coalesces and compresses those slices into the final index file. Because the transient files only exist for as long as it takes to collect 128 KB of data ingest, you don't need to spend Nutanix CPU resources on compressing them inline.

Nutanix uses a two-phase post-process compression methodology. The initial compression pass (LZ4) works on data that has been inactive for longer than one day, then a further compression scheme (LZ4HC) transforms data that has been cold for three days. When compared to earlier algorithms such as Snappy, the LZ4 compression family offers an improved tradeoff between compression and decompression speeds versus CPU utilization.

Erasure Coding

The Nutanix platform relies on a replication factor for data protection and availability. This method provides the highest degree of availability because it doesn't require the system to recompute data on failure or to read from more than one storage location. However, this feature requires full copies and thus occupies additional storage resources. Nutanix strikes a balance between providing availability and reducing required storage by transforming data using erasure coding (EC-X).

Like the concept of RAID (levels 4, 5, 6, and so on), EC-X encodes a strip of data blocks on different nodes and calculates parity. In the event of a host or disk failure, the system can use the parity to decode any missing data blocks. In the AOS distributed fabric, the data block is an extent group, and each data block must be on a different node and belong to a different vDisk.

You can configure the number of data and parity blocks based on how many failures you need to tolerate. In most cases, the configuration is <number of data blocks> / <number of parity blocks>.

For example, replication factor 2 availability (n + 1) could consist of three or four data blocks and one parity block in a strip (3:1 or 4:1). Replication factor 3 availability (n + 2) could consist of three or four data blocks and two parity blocks in a strip (3:2 or 4:2).

We can calculate the expected overhead as <number of parity blocks> / <number of parity blocks + number of data blocks>. For example, a 4:1 strip has 20 percent overhead, or 1.25x, compared to the 2x of replication factor 2.

EC-X is a post-process framework that doesn't affect the traditional write I/O path. The encoding uses the Curator MapReduce framework for task distribution.

The following figure depicts a normal environment using replication factors.

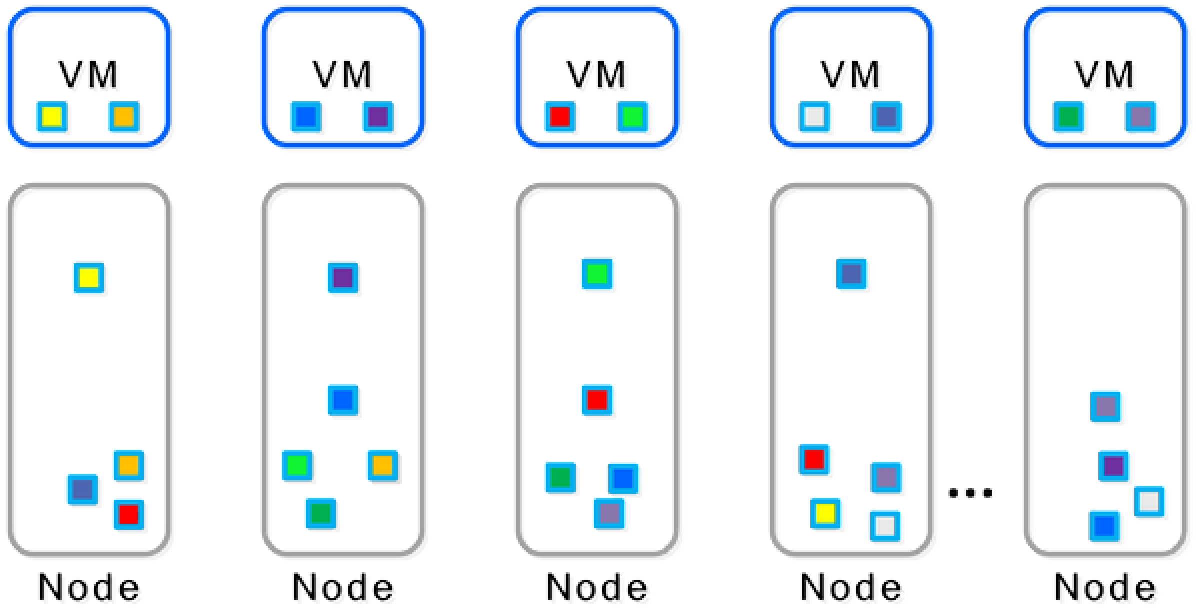


Figure 7: Data Protection: Traditional Replication Factors

In this scenario, we have a mix of both replication factor 2 and replication factor 3 data, with primary copies stored locally and replicas distributed to other nodes throughout the cluster.

When a Curator full scan runs, it finds eligible extent groups available for EC-X and distributes and throttles the encoding tasks using Chronos.

The following figure shows an example 4:1 and 3:2 strip.

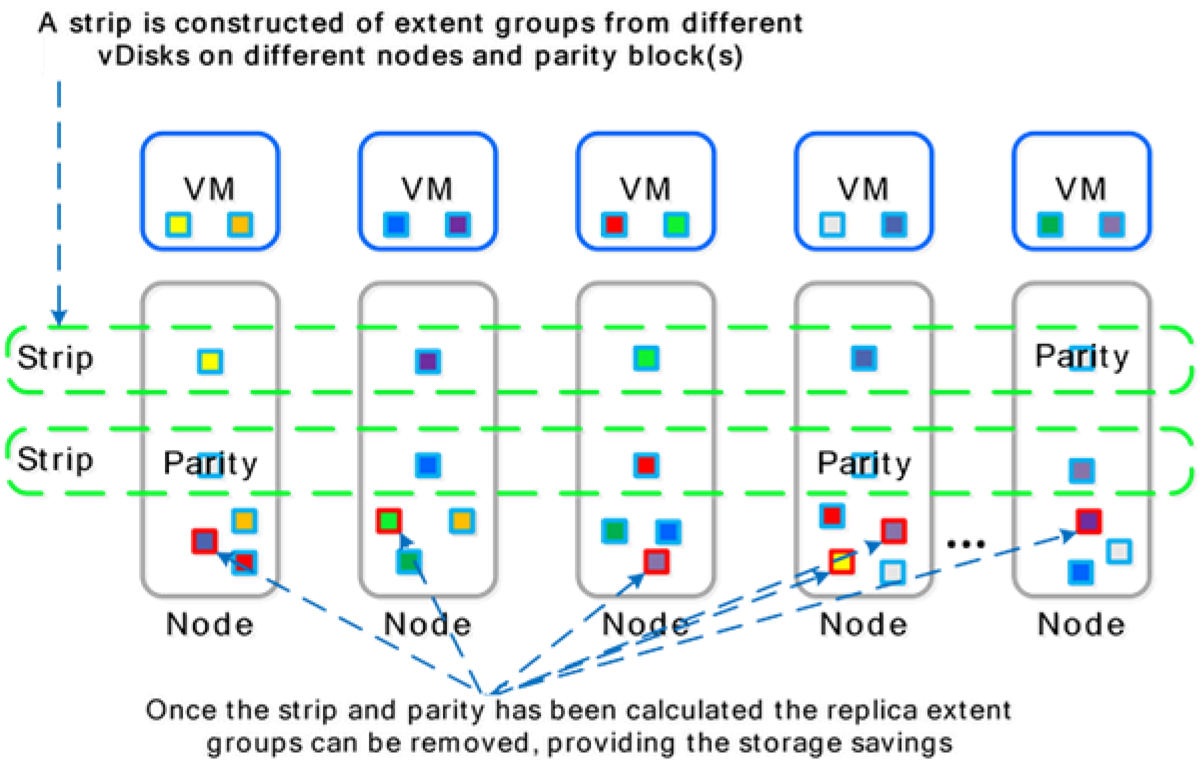

A strip is constructed of extent groups from different vDisks on different nodes and parity block(s)
VM
VM
VM
VM
VM
Strip
Strip
Parity
Parity
Parity
Node
Node
Node
Node
Node
Once the strip and parity has been calculated the replica extent groups can be removed, providing the storage savings

Figure 8: Erasure-Coded Strip

Once EC-X has successfully calculated the strips and parity, the system removes the replica extent groups. The following figure shows the storage savings in the environment after running EC-X.

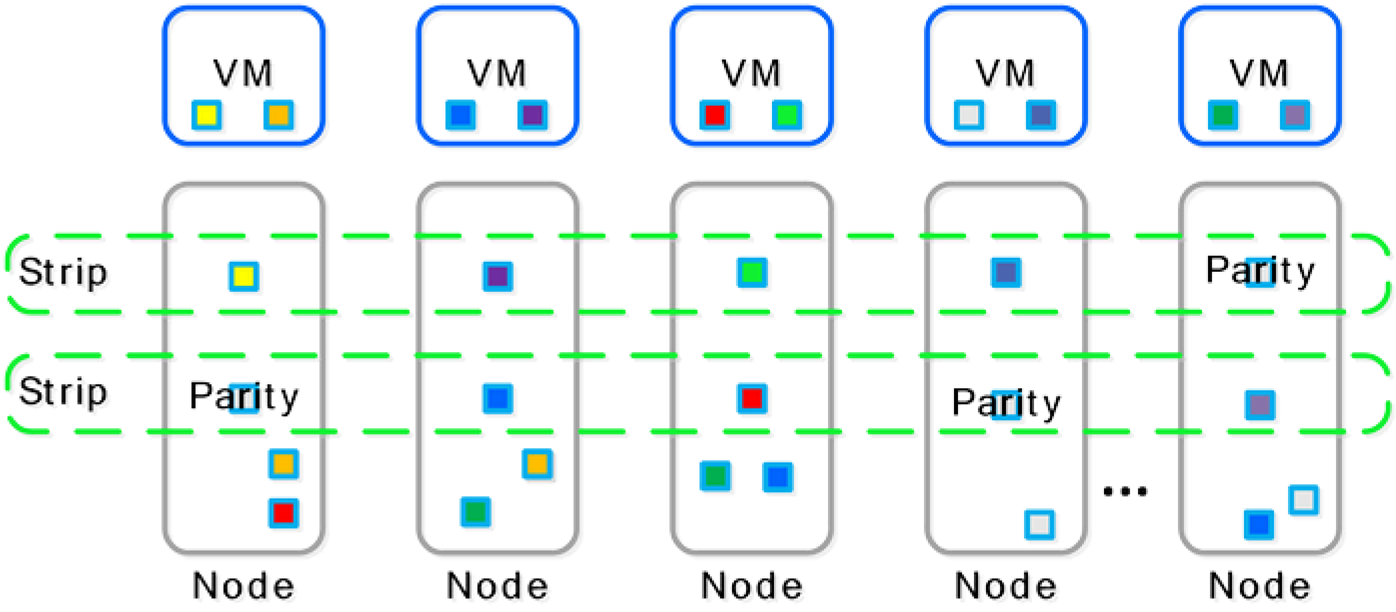


Figure 9: Erasure-Coded Strip Savings

Disk Balancing

In the following section, we describe disk balancing in some detail, including sample scenarios.

The AOS distributed fabric is a dynamic platform that can react to various workloads and allow single clusters to mix heterogeneous node types (for example, the compute-heavy NX-3000 and the storage-heavy NX-8000). When mixing nodes with larger storage capacities, it’s important to ensure uniform data distribution throughout the cluster, which the native disk balancing feature can accomplish. Integrated with the ILM, disk balancing is based on a node’s utilization of its local storage capacity. Its goal is to keep utilization uniform across nodes once the utilization has breached a certain threshold.

The following figure shows an example of a mixed cluster (NX-3000 + NX-8000) in an unbalanced state.

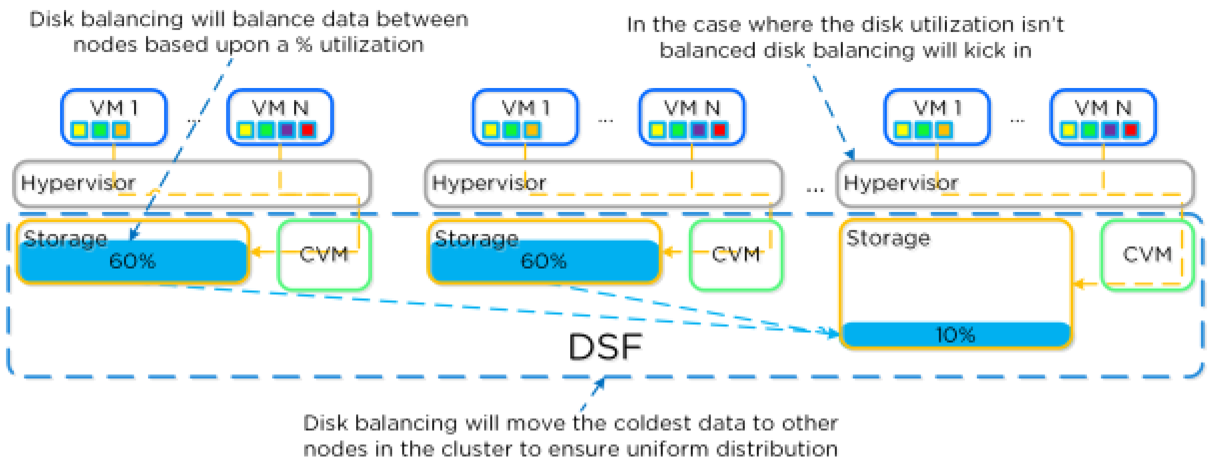

Disk balancing will balance data between nodes based upon a % utilization
In the case where the disk utilization isn't balanced disk balancing will kick in
VM 1
...
VM N
VM 1
...
VM N
VM 1
...
VM N
Hypervisor
Hypervisor
...
Hypervisor
Storage
60%
CVM
Storage
60%
CVM
Storage
10%
CVM
DSF
Disk balancing will move the coldest data to other nodes in the cluster to ensure uniform distribution

Figure 10: Disk Balancing: Unbalanced

Disk balancing uses the Curator framework and runs both as a scheduled process and when a node has breached a set threshold (in other words, when local node capacity utilization is greater than n percent). When the data isn't balanced, Curator determines which data the nodes need to move and distributes the movement tasks to nodes in the cluster. When a cluster's node types are homogeneous (for example, all NX-3000s), utilization should be fairly uniform. However, if there are VMs writing much more data than others, the per-node capacity utilization can become skewed. In this case, disk balancing runs, moving the coldest data on the overused node to other nodes in the cluster. When the node types are heterogeneous (for example, NX-3000 + NX-8000) or when you use a node in a storage-only mode (not running any VMs), you're more likely to need to move data.

The following figure shows an example of a mixed cluster in a balanced state after running disk balancing.

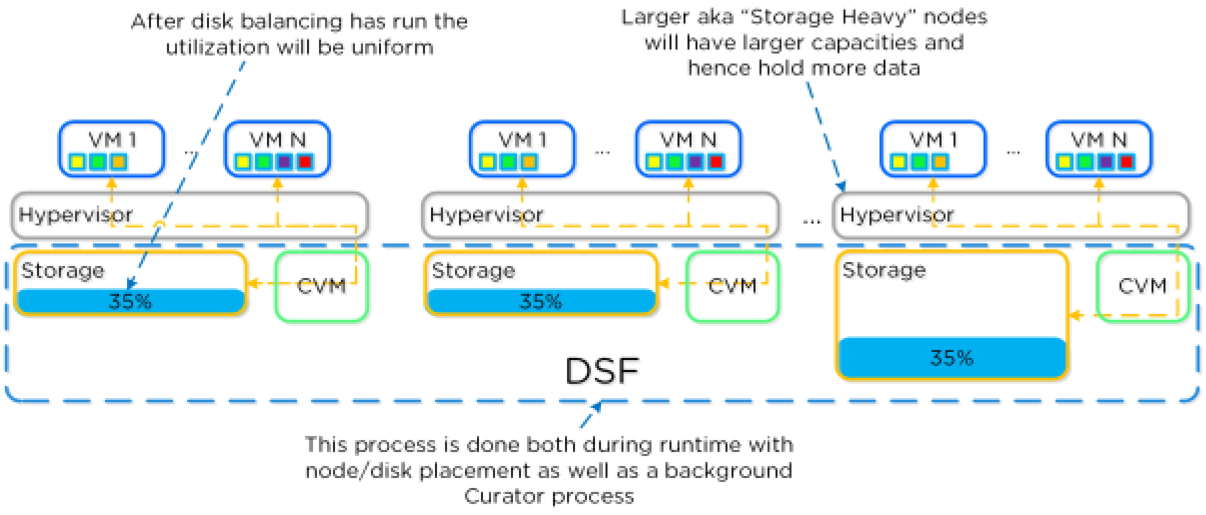

After disk balancing has run the utilization will be uniform
Larger aka "Storage Heavy" nodes will have larger capacities and hence hold more data
VM 1
VM N
...
Hypervisor
Storage
35%
CVM
DSF
This process is done both during runtime with node/disk placement as well as a background Curator process

Figure 11: Disk Balancing: Balanced

Users can also run some nodes in a storage-only state, where only the CVM runs on nodes whose primary purpose is bulk storage capacity. In this case, the CVM can access the full node's memory to provide a much larger read cache. The following figure shows an example of disk balancing moving data to a storage-only node in a mixed cluster from the active VM nodes.

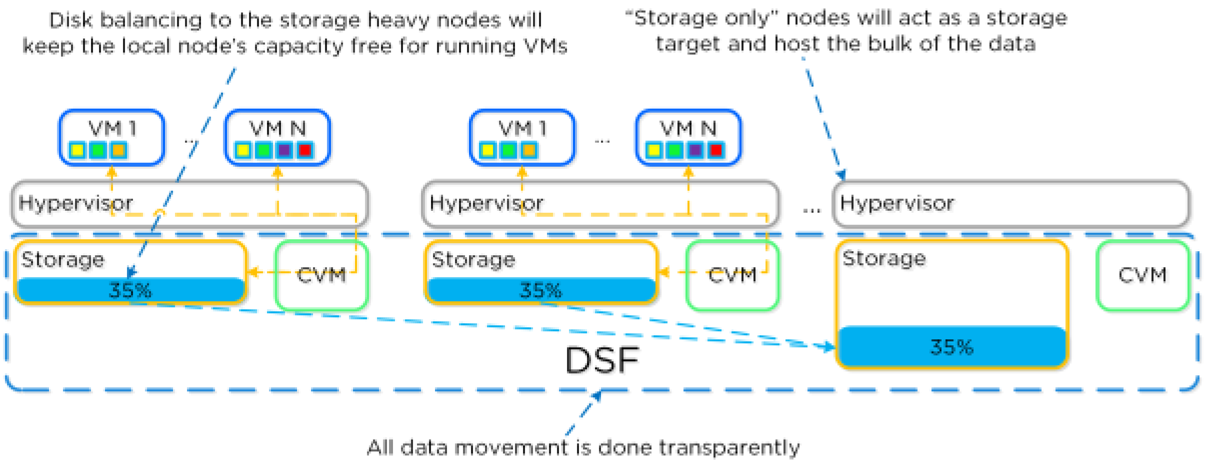

Disk balancing to the storage heavy nodes will keep the local node's capacity free for running VMs
"Storage only" nodes will act as a storage target and host the bulk of the data
VM 1
VM N
...
Hypervisor
Storage
35%
CVM
DSF
All data movement is done transparently

Figure 12: Disk Balancing: Storage-Only Node

Data I/O Detail

The following figure represents the high-level I/O path for VMs and Splunk instances running on Nutanix. The AOS distributed fabric handles all I/O operations, which occur on the local node to provide the fastest possible I/O performance. Machine data writes to the Splunk indexer locally for all VMs on the same hypervisor node and over 10 GbE for VMs and sources hosted remotely or on another node.

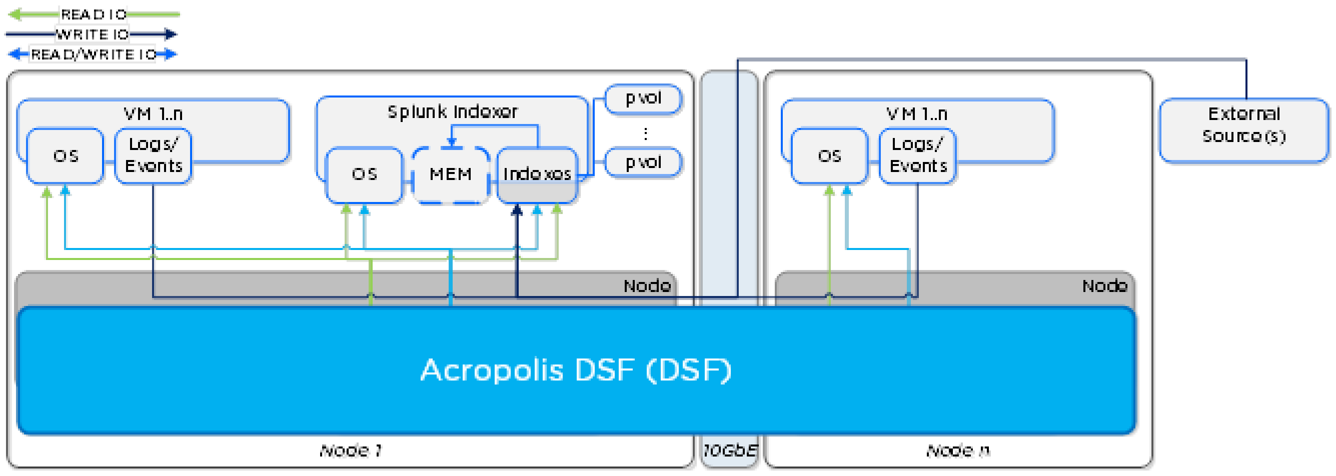


Figure 13: High-Level Data I/O Path

The next figure describes the detailed I/O path for VMs and Splunk instances running on Nutanix. All write I/O—including data input to the Splunk indexer—occurs on the local node's SSD tier to provide the highest possible performance. Read requests for the Splunk indexes occur locally and are served from the high-performance in-memory read cache (if cached) or from the SSD or HDD tier, depending on placement. Each node also saves frequently accessed local data in the read cache (for example, VM data or Splunk indexes). Nutanix ILM constantly monitors data and I/O patterns to choose the appropriate tier placement.

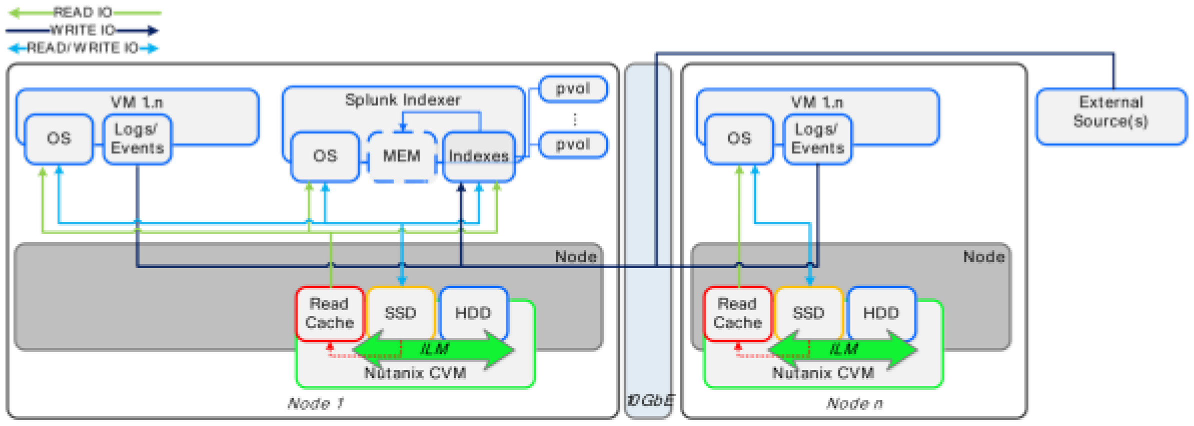


Figure 14: Detailed Data I/O Path

Deployment Options

With the Splunk on Nutanix solution, you can start with a single block and then scale up incrementally, one node, one block, or multiple blocks at a time. This design provides the best of both worlds—the ability to start small and grow to massive scale without any impact on performance.

Note: You can have either a dedicated (Splunk only) or shared (Splunk + other VMs) deployment on Nutanix. For larger deployments, we recommend a dedicated environment; with proper resource planning, you can share your smaller deployments.

Nonclustered Deployment

In a nonclustered deployment, you can run dedicated search heads or individual users can use each indexer separately.

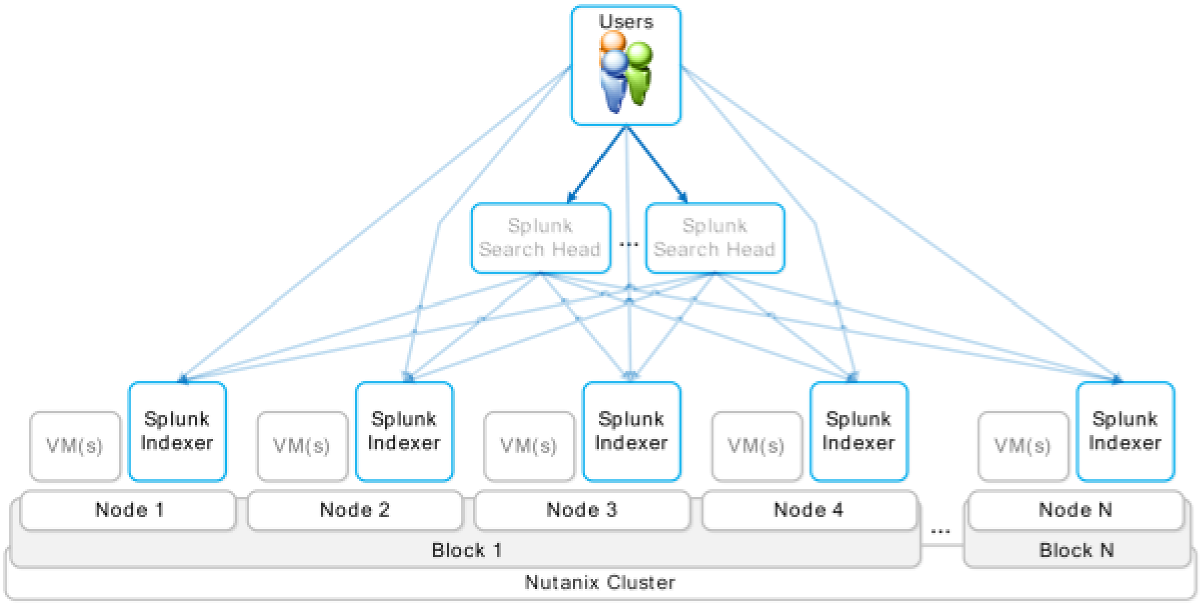


Figure 15: Nonclustered Deployment

Search Head Cluster Deployment

A search head cluster deployment brings together multiple Splunk search heads to form a distributed search platform. You can then use a load balancer to distribute the user load across the search heads.

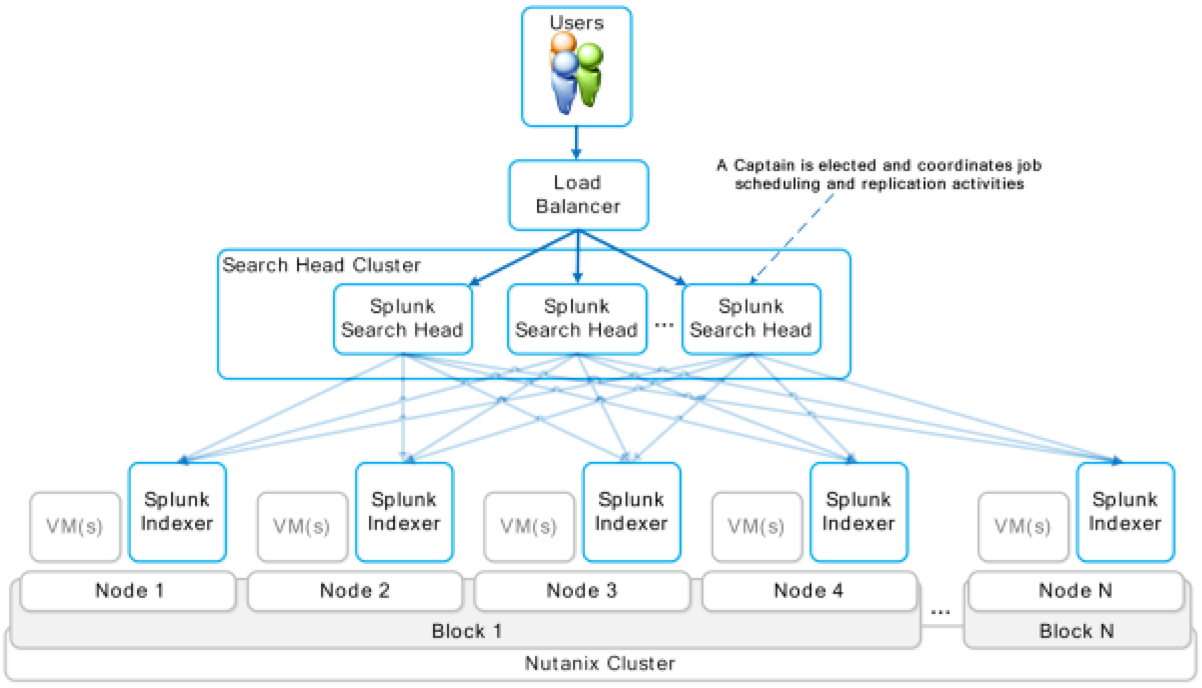


Figure 16: Search Head Cluster Deployment

Notice that we don’t include a clustered Splunk indexer or search head pooling deployment. We don’t need to because Nutanix provides storage high availability (HA), and the hypervisor HA capabilities handle indexer availability as well. Because search head clustering is more flexible than search head pooling, we recommend it when deploying larger Splunk instances on Nutanix.

Multiple-Indexer Cluster Deployment

Multisite clusters consist of a manager, peer indexers, and search head nodes. There is exactly one manager node and any number of indexer peers and search heads.

Every node is part of a specific site determined by physical location. For example, your San Jose data center (DC) is Site 1 and your Phoenix DC is Site 2. Typically, each site also hosts search heads. By employing search head affinity in this way, you increase search efficiency by ensuring that the search accesses data locally.

The following diagram is an example of a two-site cluster.

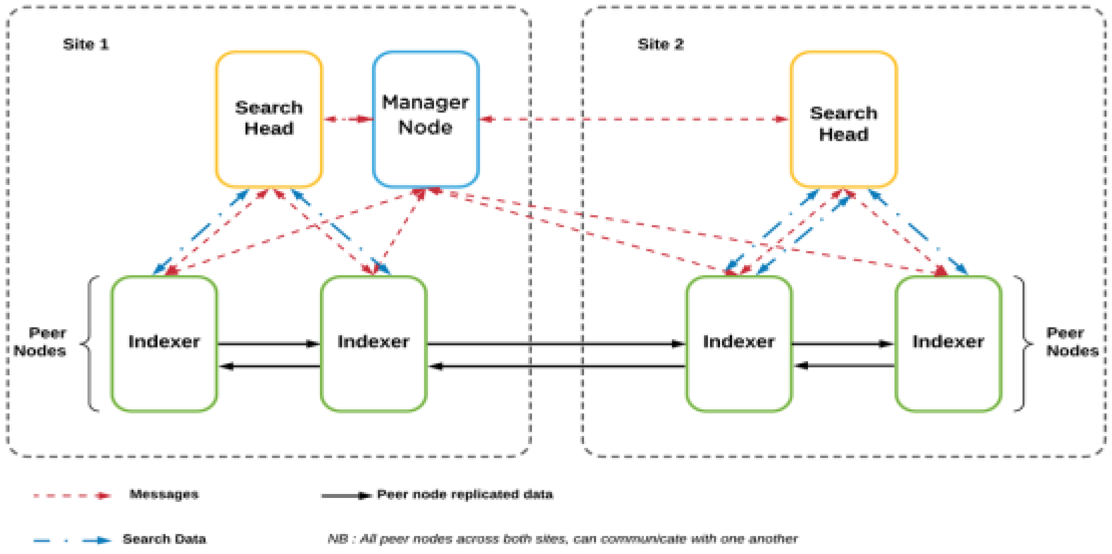


Figure 17: Multiple-Indexer Cluster Architecture Across Two Sites

The manager node controls the entire cluster. Though the manager node resides on a site, it isn't actually a member of any site. It does, however, have a built-in search head that requires you to specify a site for the manager as a whole. The search head functionality in the manager is for testing only, not for production. The previous example shows how the search heads have been configured for site affinity. Each site has a search head that searches the set of peer nodes on that site. In turn, the indexer peer nodes replicate data across the site boundaries, which allows site-affinity-based searching to work.

SmartStore with Nutanix Objects Deployment

To deploy SmartStore with Nutanix Objects, perform the following steps:

- Configure Nutanix Objects as a remote store for SmartStore.
- Configure the Splunk indexer (or cluster peer nodes) to access the remote store.
- Test the SmartStore deployment with Objects.

For more information on the deployment steps, refer to Splunk documentation and the Nutanix University video “How To Setup a Splunk Smart Store with Nutanix.”

SmartStore minimizes the amount of data on local storage while maintaining the fast indexing and search capabilities characteristic of Splunk Enterprise deployments. The following architectural diagram shows the flow of data for a SmartStore-enabled index in an indexer cluster.

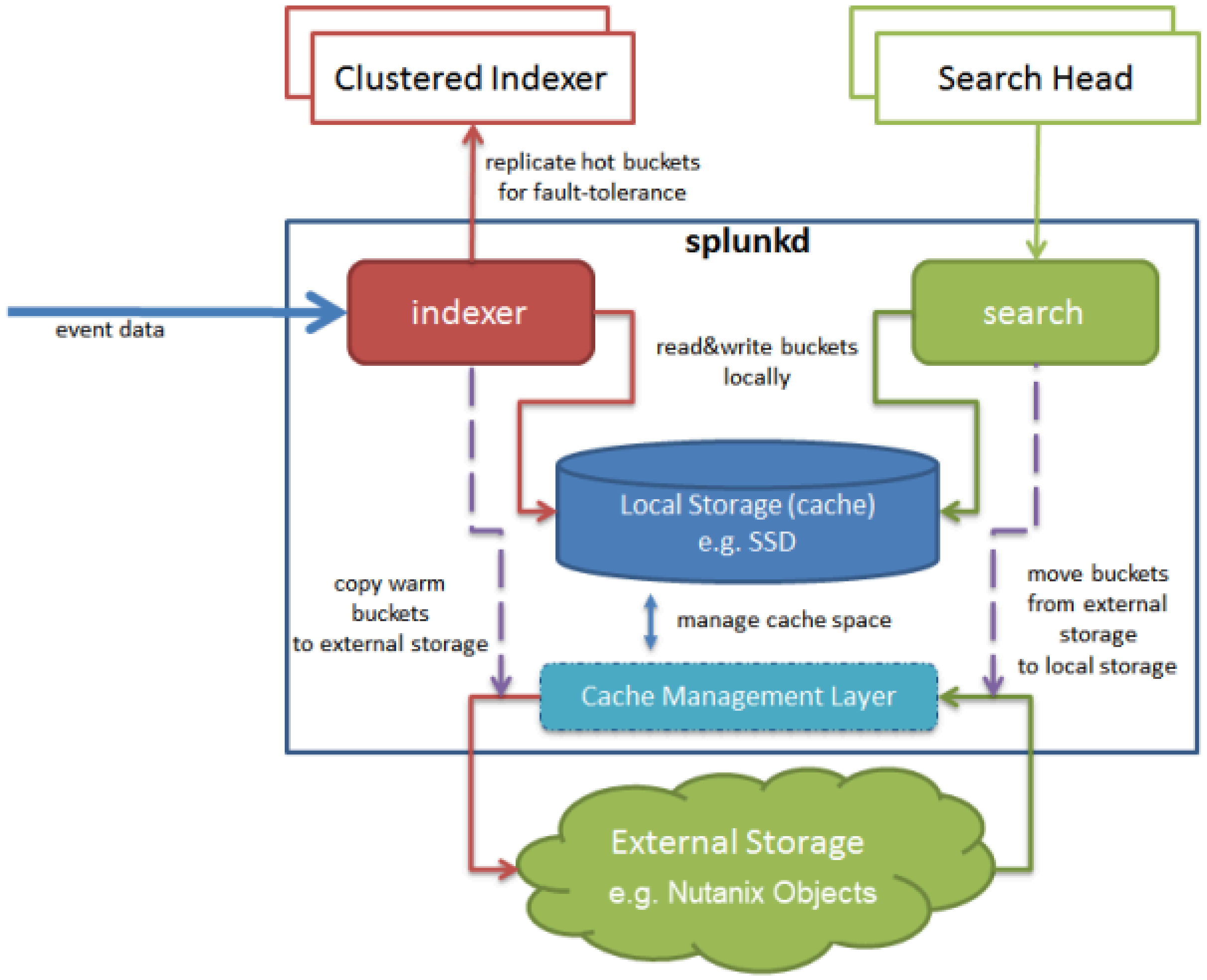

Clustered Indexer
Search Head
replicate hot buckets for fault-tolerance
splunkd
event data
indexer
search
read&write buckets locally
Local Storage (cache) e.g. SSD
copy warm buckets to external storage
manage cache space
move buckets from external storage to local storage
Cache Management Layer
External Storage e.g. Nutanix Objects

Figure 18: Data Flow for SmartStore-Enabled Index

The following image provides a step-by-step data flow diagram.

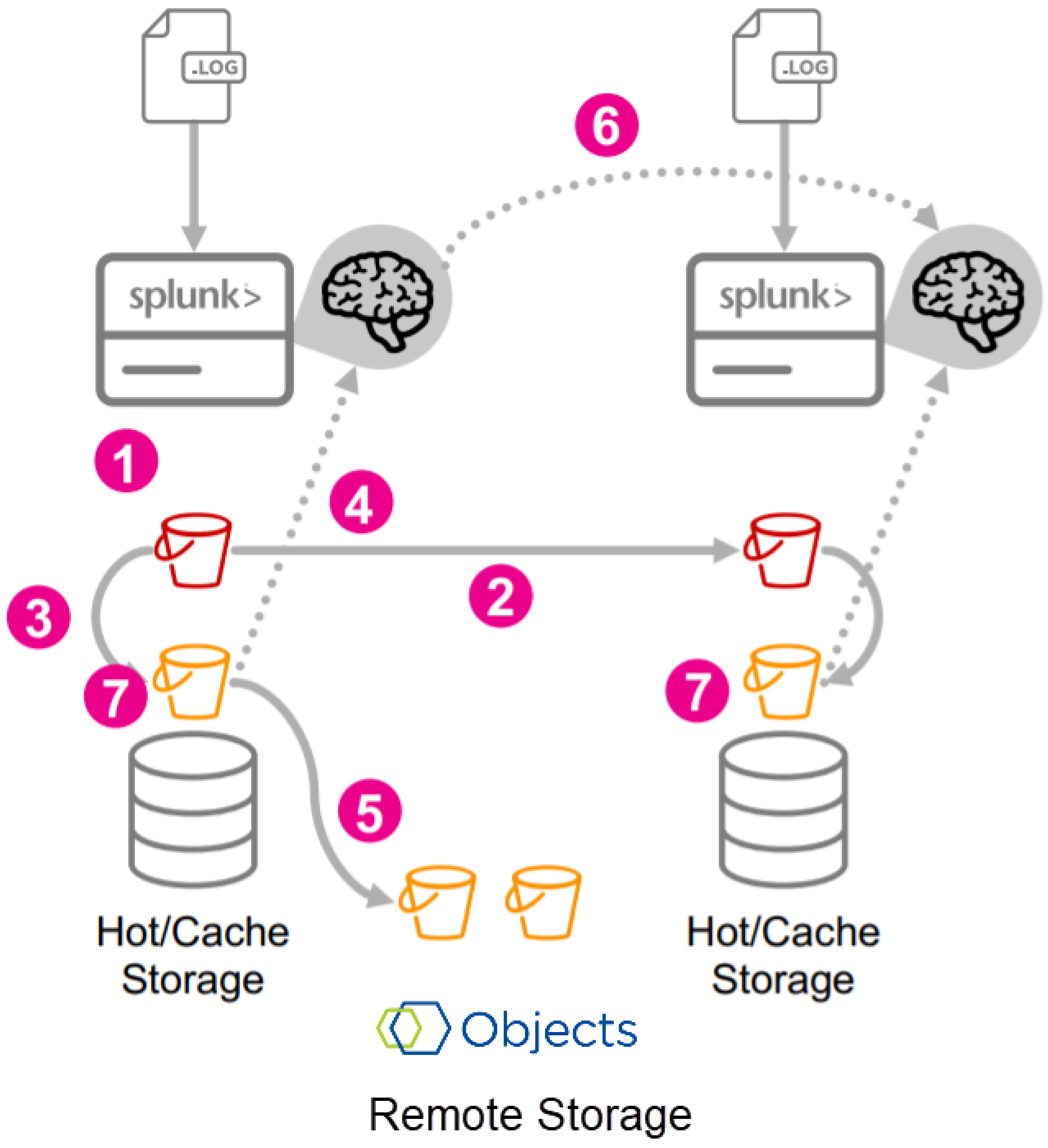


Figure 19: Detailed Data Flow for SmartStore-Enabled Index with Nutanix Objects

(Source: Splunk Documentation)

- Data arrives at source indexer and is written to a hot bucket locally.
- The hot bucket streams to cluster peers according to replication factor.
- Replication completes and the hot buckets roll to warm.
- Buckets are registered with their cache managers.
- Cache manager on source peer uploads the bucket to the remote object store (Nutanix Objects) and leaves the existing copy in its cache.
- The source peer notifies replication peers that the bucket has uploaded successfully.
- Cached copies remain on the peers until evicted by the local cache manager.

The indexer clusters maintain replication and search factor copies of hot buckets only. Remote storage (Nutanix Objects) ensures the high availability, data fidelity, and disaster recovery of the warm buckets.

A search head is used to search the data. Search heads get all the traffic from the end users. End users log on to the UI using the search head and run their searches, reports, alerts, and dashboards.

SmartStore Multisite Deployment

The architecture of single- and multisite clusters is similar, but there are a few differences for multisite clusters:

- Each node (manager, peer, search head) has an assigned site. For example, if you want your cluster to span servers in San Jose and Idaho, you assign all nodes in San Jose to site1 and all nodes in Idaho to site2.
- Replication of bucket copies occurs in a site-aware manner.
- Search heads distribute their searches across local peers only whenever possible, which allows you to configure a cluster so that search heads perform their searches only on data stored on their local sites. This configuration reduces network traffic while still providing access to the entire dataset, because each site contains a full copy of the data. This benefit is known as search affinity.
- Bucket-fixing activities respect site boundaries when applicable.

The following architecture diagram shows a two-site cluster. In this example, the cluster has been configured for search affinity. Each site has its own search head, which searches the set of peer nodes on its site. However, a search head might also search peers outside its own site. The peers replicate data across site boundaries for both disaster recovery and search affinity.

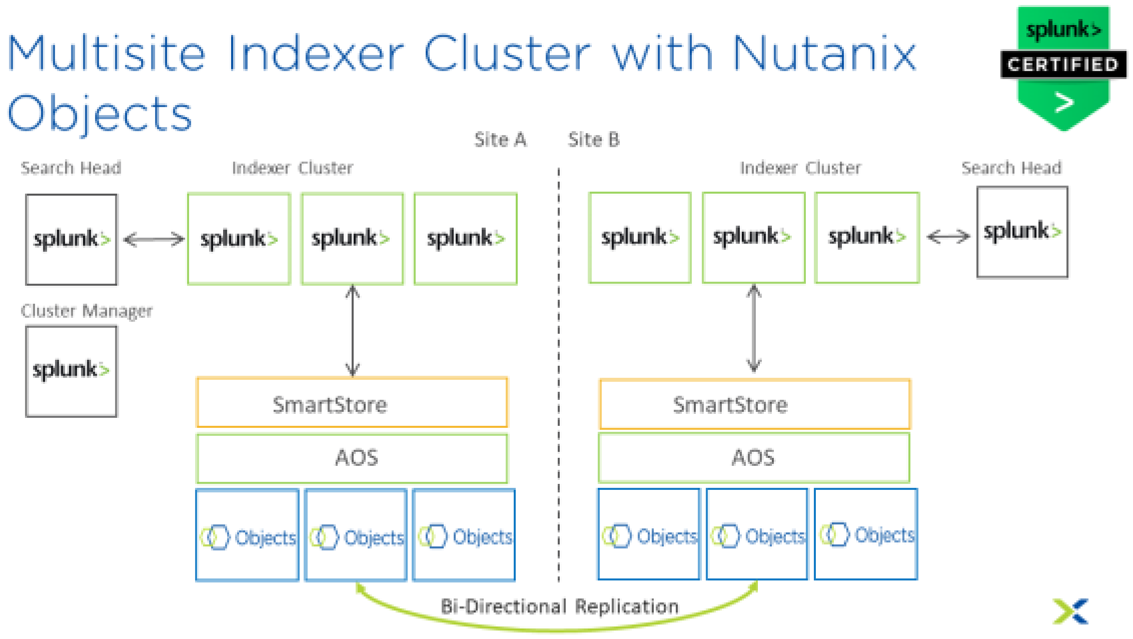


Figure 20: Overview of a Multisite Indexer Cluster with Nutanix Objects

SmartStore supports on-premise multisite clusters. The topology is limited to two sites with an intersite latency between 100 and 300 ms (100 ms preferred for best performance). Each site has an active object store and each indexer node points to the same object store URI through a third-party virtual IP or load balancer. The virtual IP or load balancer routes traffic to the object store at its site location. In the case of site or object store failure, the virtual IP reroutes traffic to the remaining object store. In these scenarios there may be increased traffic across the WAN and some queries may return partial results due to replication lag between the data stores, particularly if they aren’t in complete sync with each other because of the outage.

Note: You need a standby cluster manager on the other site in case of a site failure that affects the active cluster manager.

Nutanix Features for Increased Splunk Uptime

AOS Distributed Fabric

Multisite Splunk configurations that run on Nutanix set both the site replication and site search factors to maintain a single copy of an index bucket at the site originating the data (in other words, the site that is receiving external data). The alternate site maintains an additional index copy. As the Nutanix platform always maintains block-level redundancy, both the originating and remote site buckets replicate per the overall Nutanix cluster-level redundancy factor, which is 2 by default.

The following diagram shows the server.conf settings you need to configure multisite indexing on the Nutanix platform. Search factor, sites, and so on require additional settings. Please refer to Splunk documentation for the most up-to-date information.

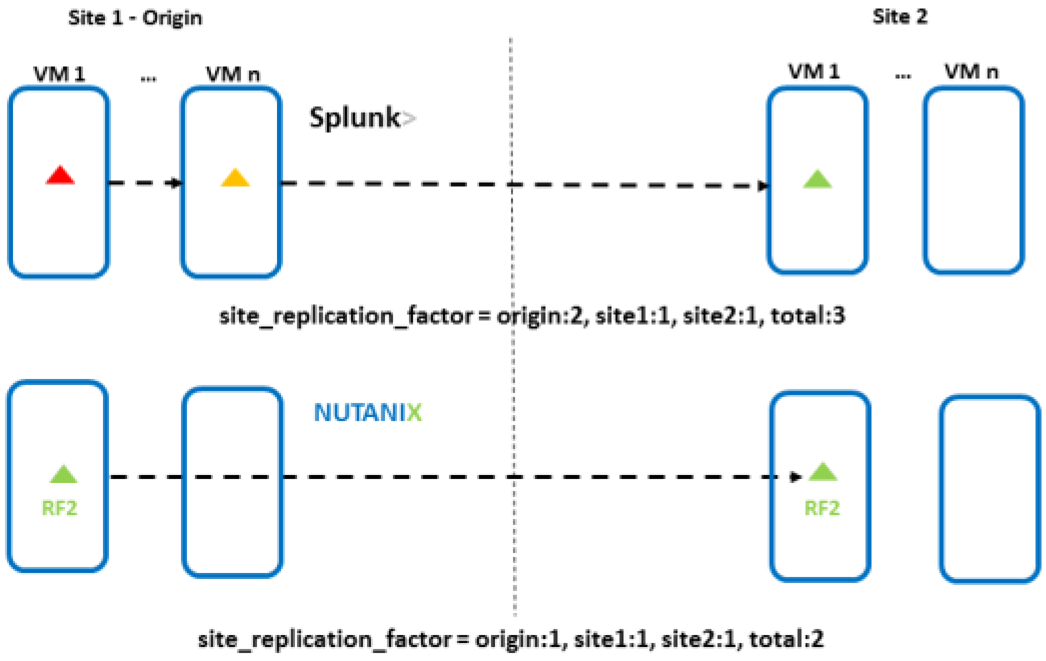

Site 1 - Origin
Site 2
VM 1
...
VM n
VM 1
...
VM n
Splunk>
site_replication_factor = origin:2, site1:1, site2:1, total:3
NUTANIX
RF2
RF2
site_replication_factor = origin:1, site1:1, site2:1, total:2

Figure 21: Splunk Site Replication Factor Changes on Nutanix

Nutanix High Availability

The following diagrams cover various common failure scenarios. In them, we demonstrate how the Acropolis management software protects against outages and show the potential benefits the Nutanix self-healing mechanisms can bring to Splunk cluster components.

In the following figure, we show the Nutanix cluster in a steady state, in which both platform- and application-level components are working as expected.

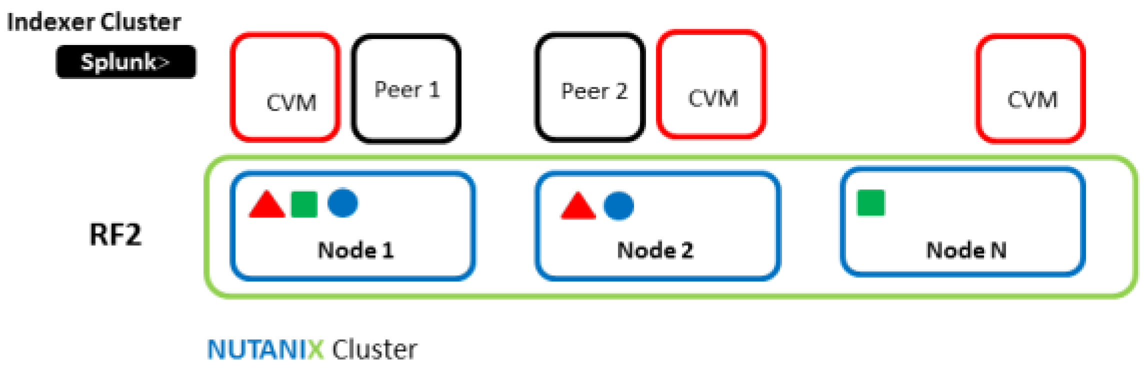


Figure 22: Nutanix Cluster Hosting Splunk VMs

If the CVM on any host reboots or fails, the autopath functionality redirects the application VM's I/O to an alternate CVM. With this redirection, a Splunk indexer or search head, for example, doesn't need to migrate to another hypervisor host. Hence, there's no need to take any remedial action (such as a postmigration index rebalance or maintenance of site-specific bucket replication levels) at the Splunk level. Once the CVM outage ends (for example, a reboot completes after software upgrade), the I/O path returns to the local CVM. For more details on CVM failure handling, refer to the Controller VM Failure section of the Prism Web Console Guide (6.0).

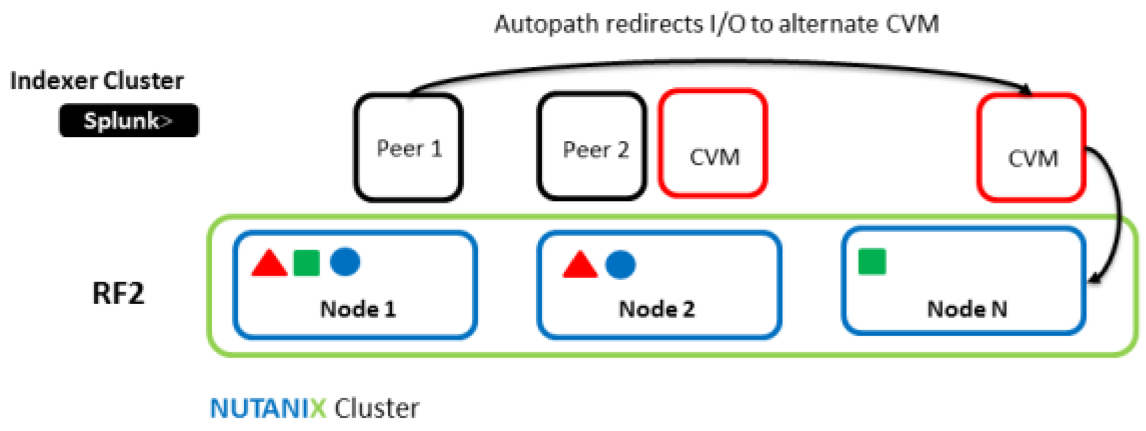


Figure 23: Nutanix Autopath Redirects I/O on CVM Failure

During a host failure, Nutanix HA can live-migrate Splunk application VMs to an alternate host. In a multisite Splunk deployment, the cluster leader exists on one of the sites. To protect this important component, Nutanix enables live migration to preserve cluster leader uptime. The same protection is available for both indexer and search head components. When a Splunk indexer peer node fails, an administrator can decide between:

1. Allowing live migration and preserving data locality using the Nutanix distributed processes.
2. Letting the Splunk application handle what Splunk calls bucket fixup. In this operation, when Splunk loses an indexer, the system creates additional index bucket copies across the remaining nodes, making additional indexes searchable.

For more details on handling host failure, refer to the Host Failure section of the Prism Web Console Guide (6.0) and Splunk's documentation on component recovery in multisite indexer scenarios.

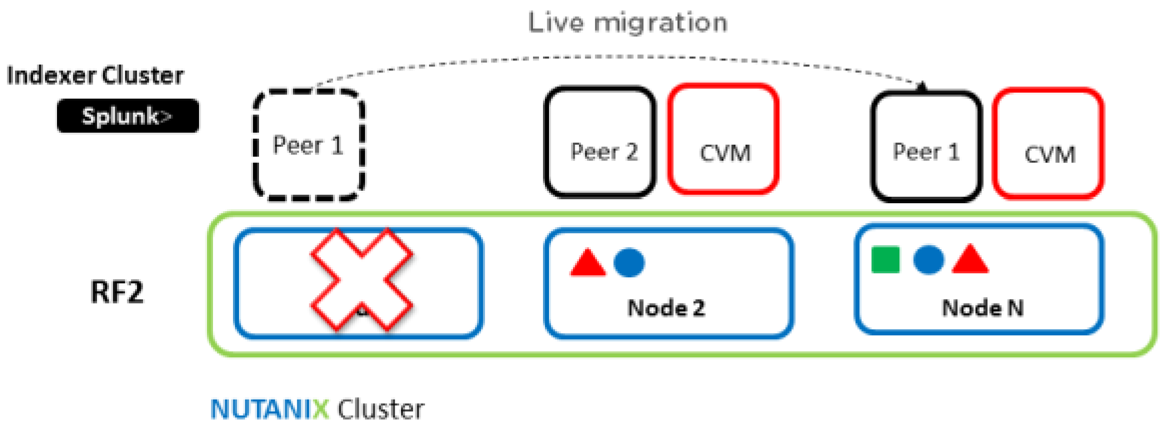


Figure 24: Nutanix HA Live Migrates Application VMs on Cluster Host Failure

6. Solution Design

Design Decisions

The following tables cover design decisions and rationale for Splunk on Nutanix.

Note: If your setup is a Splunk Enterprise Security (Splunk ES) deployment, please see the Additional Sizing Considerations When Running Splunk Enterprise Security section.

Table: General Design Decisions: General

Item	Detail	Rationale
Minimum size	3 Nutanix nodes (3 hypervisor hosts)	Minimum size requirement
Scale approach	Incremental modular scale	Allow for growth from PoC to massive scale
Scale unit	Nodes, blocks, or pods	Granular scale to precisely meet the capacity demands
Infrastructure services	Small deployments: shared cluster; Large deployments: dedicated cluster (Node A from 3 blocks or a 1350)	Dedicated infrastructure cluster for larger deployments (best practice)

Table: General Design Decisions: Nutanix

Item	Detail	Rationale
Cluster size	<32 nodes: 1 cluster; >32 nodes: 2 clusters scaled up to 48 nodes each; >96 nodes: 1 additional cluster every 48 nodes	Isolated fault domains; Keeping single hop between nodes
Storage pool	1 storage pool per cluster	Standard practice; ILM handles tiering

Item	Detail	Rationale
Container	1 container for VMs and data; Replication factor: 2; Features: compression + EC-X	Standard practice; Compression for index files
Features and enhancements	Increase CVM memory to 32 GB	Best practice

Table: General Design Decisions: Splunk Platform

Item	Detail	Rationale
Distributed deployment	No	VM HA for availability
Search head pooling	No	Search head clustering used instead
Search head clustering	Yes, for larger deployments	

Table: General Design Decisions: Miscellaneous

Item	Detail	Rationale
LVM stripe size	32 KB or 64 KB	Best performance
Disks in LVM volume	6	Best performance

The Nutanix solution supports deployments on multiple hypervisors. We’ve outlined any hypervisor-specific design decisions in the following tables.

Table: Hypervisor-Specific Decisions: AHV

Item	Detail	Rationale
Cluster size	Same as Nutanix cluster size (minimum of 3 hosts)	Isolated fault domains
Infrastructure services hosting	N/A	Built into Nutanix platform
Datastore	N/A	iSCSI disks used

Table: Hypervisor-Specific Decisions: vSphere

Item	Detail	Rationale
Cluster size	Up to 48 hypervisor hosts (minimum of 3 hosts)	Isolated fault domains

Item	Detail	Rationale
Clusters per vCenter	Up to 4 x 16 host clusters	Task parallelization
Infrastructure services hosting	Small deployments: shared cluster; Large deployments: dedicated cluster	Dedicated infrastructure cluster for larger deployments (best practice)
Datastore	1 Nutanix NFS datastore per pod for VMs and data	Nutanix handles I/O distribution and localization

We've determined Splunk component VM sizes based on their reference hardware and internal testing.

Table: Splunk Indexer VM Sizing

VM	CPU Resources	Memory Resources	Disk Resources	Capability
Minimum	16 vCPU*	12 GB+	OS: 1 x n GB; DATA: 6 x n GB	Up to approximately 300 GB per day indexing
Midrange	Up to 24 vCPU*	64 GB+	OS: 1 x n GB; DATA: 6 x n TB	Up to approximately 300 GB per day indexing with additional compute for search concurrency
Large or Splunk Enterprise Security (ES)	Up to 48 vCPU*	128 GB+	OS: 1 x n GB; DATA: 6 x n TB	Up to approximately 300 GB per day indexing; 100 GB per day ingest for ES. With additional compute for sustained search. Critical for Premium Splunk Apps.

* If you see the CPU pegged at 100 percent, you can increase the vCPU count up to the suggested maximum. Keep vCPU count within a NUMA boundary and make sure that the vCPU count is fewer than the total number of cores minus the cores used for the Nutanix CVM.

The next table offers Splunk search head VM sizing recommendations for minimum and large deployments. In a small configuration, you don't need a separate search head.

Table: Splunk Search Head VM Sizing

VM	CPU Resources	Memory Resources	Disk Resources	Capability
Minimum	8 vCPU	12 GB+	OS: 1 x n GB; DATA: N/A	Up to 16 users per 750 GB–1 TB of daily index data
Large or Splunk ES	16 vCPU	32 GB+	OS: 1 x n GB; DATA: N/A	Up to 24 users per 750 GB–1 TB of daily index data

Note: Splunk suggests a maximum of eight indexers per search head. Search head clustering requires a minimum of three search heads. In our design, we used the minimum search head specification.

Splunk Sizing

The following section covers the Splunk sizing and considerations for running Splunk on Nutanix. We assume at least one Splunk indexer per Nutanix node.

Note: It's always a good practice to add a buffer for contingency and growth.

Splunk Indexer Sizing

Step 1a: Calculate the Estimated Required Hot and Warm Storage

- Required hot and warm storage = (daily average indexing rate in GB x hot and warm retention policy in days) / 2

Note: Splunk assumes a 50 percent compression ratio.

For example, if there are 500 GB of data indexed daily and you need to keep hot and warm data for a minimum of two months:

- Required hot and warm storage = (500 GB per day x 60 days) / 2 = 15,000 GB = 15 TB

Step 1b: Calculate the Estimated Required Cold and Frozen Storage

- Required cold and frozen storage = ((daily average indexing rate in GB x cold and frozen retention policy in days) / 2) – hot and warm storage

Note: Splunk assumes a 50 percent compression ratio.

For example, if you index 500 GB of data daily and have 15 TB of hot and warm data, and you keep cold and frozen data for one year:

- Required cold and frozen storage = ((500 GB per day x 365 days) / 2) – 15,000 GB = 76,250 GB = ~76 TB

Step 1c: Calculate the Estimated Required Total Storage

- Required total storage = required hot and warm storage + required cold and frozen storage

For example, if there are 15 TB of required hot and warm data and 76 TB of required cold and frozen data:

- Required total storage = 15 TB + 76 TB = 91 TB

Step 2a: Calculate the Total Required Number of Hot and Warm Nutanix Nodes by Storage Capacity

- Total number of Nutanix nodes for hot and warm by storage capacity = (required hot and warm storage) / (storage per hot and warm node / Nutanix data protection overhead)

For example, if there are 15 TB of required hot and warm storage, the Nutanix data protection overhead is two (assuming replication factor 2 with no EC-X or compression savings), and there are 8 TB of raw storage per hot and warm node:

- Number of Nutanix nodes for hot and warm by storage capacity = ROUNDUP (15 TB / (8 TB / 2)) = 4 nodes

If we take the same inputs while leveraging EC-X, the result is an effective data protection overhead of 1.2 (EC 4:1 strip).

- Number of Nutanix nodes for hot and warm by storage capacity = ROUNDUP (15 TB / (8 TB / 1.2)) = 3 nodes

Step 2b: Calculate the Total Required Number of Cold and Frozen Nutanix Nodes by Storage Capacity

- Total number of Nutanix nodes for cold and frozen by storage capacity = (required cold and frozen storage) / (storage per cold and frozen node / Nutanix data protection overhead)

If there are 76 TB of required cold and frozen storage, the Nutanix data protection overhead is two (assuming replication factor 2 with no EC-X or compression savings), and there are 16 TB of raw storage per cold and frozen node:

- Number of Nutanix nodes for cold and frozen by storage capacity = ROUNDUP (76 TB / (16 TB / 2)) = 10 nodes

If we take the same inputs while using EC-X, the result is an effective data protection overhead of 1.2 (EC 4:1 strip).

- Number of Nutanix nodes for cold and frozen by storage capacity = ROUNDUP (76 TB / (16 TB / 1.2)) = 6 nodes

Step 2c: Calculate the Total Required Number of Nutanix Nodes by Storage Capacity

- Total number of Nutanix nodes by storage capacity = total number of Nutanix nodes for hot and warm by storage capacity + total number of Nutanix nodes for cold and frozen by storage capacity

Taking the outputs from above and assuming we’re using EC-X and compression, we have three nodes for hot and warm data and six nodes for cold and frozen data:

- Total number of Nutanix nodes by storage capacity = 3 nodes + 6 nodes = 9 nodes

Step 3: Calculate the Total Required Number of Nutanix Nodes for Hot and Warm Data by Indexer Capability

- Total number of Nutanix nodes for hot and warm data by indexer capability = daily average indexing rate in GB / daily indexer ingest capability

For example, if there are 500 GB of data indexed daily and each indexer is capable of handling 150 GB of data ingest:

- Number of Nutanix nodes for hot and warm data by indexer capability = ROUNDUP (500 GB / 150 GB) = 4 nodes

Step 4: Calculate the Real Required Number of Nutanix Nodes for Hot and Warm Data

- Total number of Nutanix nodes for hot and warm data = MAX (2a, 3)

If we take the examples above, in which the number of nodes by storage is three and the number of nodes by indexer capability is four:

- Total number of Nutanix nodes / indexers for hot and warm data = MAX (three nodes, four nodes) = four nodes

Step 5: Aggregate Results

- Hot and warm nodes: four nodes (from outcome of step 4)
- Cold and frozen nodes: six nodes (from outcome of step 2b)

Splunk SmartStore Sizing

Rationale

When you initially size the PoC for a new project, you typically only ingest a limited subset of data sources or feeds. In PoC environments, a compute-intensive platform is often sufficient to handle the Splunk workload. However, once the project goes into production, the data ingested and the compute needed for concurrent searches increases, encroaching on indexing requirements. At this point, you need to migrate the indexer infrastructure to a more I/O-intensive or spindle-heavy platform. You can recycle the initial compute-based nodes by using them for search head clusters and management VMs.

As the deployment grows, your need for storage often increases more rapidly than your need for compute. As mentioned previously, SmartStore solves this problem by decoupling storage and compute resources so you can scale each as needed. The following section presents SmartStore architectures with a high-performance hot cache tier for local search and a separate Objects cluster for retaining all nonwritable data in remote storage.

Indexer Sizing

The following is an example calculation of estimated required cache storage, where:

- Daily ingest rate (I) = 500 GB
- Cache retention (C) = 90 days
- Available disk space (D) on each indexer host (Nutanix architecture provides homogenous disk space)
- Replication factor (R) = 2

We can calculate minimum cache requirement as follows:

(I x R + ((C – 1) x I)) = 500 x 2 + ((90 – 1) x 500) = 1,000 GB + 44,500 GB = ~46 TB

So, the minimum required number of indexers is minimum cache requirement size / D.

Assuming we used a current appliance model with an all-flash storage capacity of around 31 TB per node:

(46 TB x 2 for Nutanix redundancy factor) / D = (46 TB x 2) / 31 TB = ROUND (92 TB / 31 TB) = ~3 nodes

You can verify this number by comparing ingest throughput requirements per indexer (~300 GB per day per indexer for Splunk Enterprise Search and 100 GB per day per indexer for Splunk Enterprise Security). In the Enterprise Security context, you need at least five indexer nodes. We discuss Enterprise Security sizing in greater detail later in this document.

SmartStore Sizing

There is no concept of cold, frozen, or archive buckets in SmartStore. All long-term retention data is simply remote storage, namely a local Nutanix Objects cluster. Warm buckets are migrated to remote storage and then cached back to the cache storage tier, which is our indexer cluster, based on their appearance in recent searches.

- Calculate the required SmartStore storage capacity:
  - SmartStore required capacity = (daily average indexing rate in GB x warm retention policy in days) / 2
  - For a seven-year retention policy: (500 GB x (2,555 – 90)) / 2 = ~1,200 TB = 1.2 PB
- Calculate the total required number of Nutanix Objects nodes by storage capacity:
  - Total number of Nutanix Objects nodes for SmartStore capacity = SmartStore required capacity / (storage per node / Nutanix data protection overhead)
  - 1,200 TB / ((180 TB HDD + 15.36 TB SSD) / 2) = 1,200 TB / (195.36 TB / 2) = 1,200 TB / 97.68 TB = ~13 Objects nodes

Note: We calculated storage per node from 10 x 18 TB HDDs + 2 x 7.68 TB SSDs, specifically for dense storage Objects clusters.

Search Head Sizing

Step 1a: Calculate the Estimated Number of Search Heads by Users

- Estimated search heads by users = number of search users / 16 (assuming 16 users per search head)

For example, if there are 48 search users:

- Estimated search heads by users = 48 users / 16 = 3 search heads

Step 1b: Calculate the Estimated Number of Search Heads by Number of Indexers

- Estimated search heads by number of indexers = number of indexers / 8

Note: Splunk recommends a ratio of eight indexers per search head.

For example, if there are 16 indexers:

- Estimated search heads by number of indexers = 16 indexers / 8 = 2 search heads

Step 2: Calculate the Required Number of Minimum Search Heads

- Required minimum search heads = MAX (1a, 1b, 3 (if using search head clustering))

Note: You need a minimum of three search heads for search head clustering.

If we take the examples above, in which the number of search heads by users is three, number of search heads by indexers is two, and search head clustering is desired:

- Required minimum search heads = MAX (3, 2, 3) = 3 search heads

Solution Application

You can use the calculations from the previous section, which are based on the various Splunk component sizes, to determine the quantity and size of the required components. For example, you can calculate the number of indexers by taking the total amount of daily index data divided by the indexing capability per VM size.

Small Sample Scenario

The following tables contain the small sample scenario inputs.

Table: Small Sample Scenario Indexing Inputs

Item	Detail	Rationale
Total index data	Up to 150 GB per day	Sample size
Data days in hot and warm buckets	30	Sample input
Data days in cold and frozen buckets	30	Sample input
Min. # of hot and warm indexers	1 small or 1 medium	Based on daily indexing and hot and warm bucket storage
Min. # of cold and frozen nodes and indexers	N/A	Based on daily indexing and cold and frozen bucket storage

Table: Small Sample Scenario Search Inputs

Item	Detail	Rationale
Search users	Up to 16	Sample size
# of search heads	N/A	N/A at this scale

The following tables contain the small sample scenario platform details and features.

Table: Small Sample Scenario Platform Details: Splunk Features

Item	Detail	Rationale
Distributed deployment	No	VM HA
Search head pooling	No	N/A at this scale
Search head clustering	No	N/A at this scale

Table: Small Sample Scenario Platform Details: Nutanix

Item	Detail	Rationale
Min. # of nodes	3	Minimum number of nodes
# of clusters	1	Based on cluster size

Table: Small Sample Scenario Platform Details: Platform

Item	Detail
Hot and warm series	1000
Cold and frozen series	N/A

The following tables contain the small sample scenario hypervisor details for both options.

Table: Small Sample Scenario Hypervisor Details: AHV

Item	Detail	Rationale
Cluster size	3	Based on cluster size
Infrastructure services hosting	N/A	Built into Nutanix platform
Datastores	N/A	Uses iSCSI disks instead

Table: Small Sample Scenario Hypervisor Details: vSphere

Item	Detail	Rationale
Cluster size	3	Based on cluster size
Clusters per vCenter	1	Based on cluster size
Infrastructure services hosting	Shared cluster	Smaller deployment
Datastores	1 Nutanix NFS datastore	Nutanix handles I/O distribution and localization

Medium Sample Scenario

The next tables contain the medium sample scenario inputs.

Table: Medium Sample Scenario Indexing Inputs

Item	Detail	Rationale
Total index data	Up to 500 GB per day	Sample size
Data days in hot and warm buckets	60	Sample input
Data days in cold and frozen buckets	365	Sample input
Min. # of hot and warm indexers	4 medium	Based on daily indexing and hot and warm bucket storage
Min. # of cold and frozen nodes and indexers	10 medium	Based on daily indexing and cold and frozen bucket storage

Table: Medium Sample Scenario Search Inputs

Item	Detail	Rationale
Search users	Up to 24	Sample size
# of search heads	3	Minimum for search head clustering

The following tables contain the medium sample scenario platform details and features.

Table: Medium Sample Scenario Platform Details: Splunk Features

Item	Detail	Rationale
Distributed deployment	No	VM HA
Search head pooling	No	Not recommended
Search head clustering	Yes	Preferred approach

Table: Medium Sample Scenario Platform Details: Nutanix

Item	Detail	Rationale
Min. # of nodes	17	Minimum number of nodes
# of clusters	1	Isolated fault domains

Table: Medium Sample Scenario Platform Details: Platform

Item	Detail
Hot and warm series	3000, 8000
Cold and frozen series	6000

The following tables contain the medium sample scenario hypervisor details for both options.

Table: Medium Sample Scenario Hypervisor Details: AHV

Item	Detail	Rationale
Cluster size	17	Based on cluster size
Infrastructure services hosting	N/A	Built into Nutanix platform
Datastores	N/A	Uses iSCSI disks instead

Table: Medium Sample Scenario Hypervisor Details: vSphere

Item	Detail	Rationale
Cluster size	17	Based on cluster size
Clusters per vCenter	1	Based on cluster size
Infrastructure services hosting	Shared cluster	Smaller deployment
Datastores	1 Nutanix NFS datastore	Nutanix handles I/O distribution and localization

Large Sample Scenario

The following tables contain the large sample scenario inputs.

Table: Large Sample Scenario Indexing Inputs

Item	Detail	Rationale
Total index data	Up to 1 TB per day	Sample size
Data days in hot and warm buckets	60	Sample input
Data days in cold and frozen buckets	365	Sample input

Item	Detail	Rationale
Min. # of hot and warm indexers	8 medium	Based on daily indexing and hot and warm bucket storage
Min. # of cold and frozen nodes and indexers	19 medium	Based on daily indexing and cold and frozen bucket storage

Table: Large Sample Scenario Search Inputs

Item	Detail	Rationale
Search users	Up to 48	Sample size
# of search heads	4	Minimum recommended

The next tables contain the large sample scenario platform details and features.

Table: Large Sample Scenario Platform Details: Splunk Features

Item	Detail	Rationale
Distributed deployment	No	VM HA
Search head pooling	No	Not recommended
Search head clustering	Yes	Preferred approach

Table: Large Sample Scenario Platform Details: Nutanix

Item	Detail	Rationale
Min. # of nodes	31	Minimum number of nodes
# of clusters	1	Based on cluster size

Table: Large Sample Scenario Platform Details: Platform

Item	Detail
Hot and warm series	3000, 8000
Cold and frozen series	6000

The following tables contain the large sample scenario hypervisor details for both options.

Table: Large Sample Scenario Hypervisor Details: AHV

Item	Detail	Rationale
Cluster size	31	Based on cluster size
Infrastructure services hosting	N/A	Built into Nutanix platform
Datastores	N/A	Uses iSCSI disks instead

Table: Large Sample Scenario Hypervisor Details: vSphere

Item	Detail	Rationale
Cluster size	31	Based on cluster size
Clusters per vCenter	2	Based on deployment size
Infrastructure services hosting	Dedicated cluster	Smaller deployment
Datastores	1 Nutanix NFS datastore	Nutanix handles I/O distribution and localization

Extra-Large Sample Scenario

The next tables contain the extra-large sample scenario inputs.

Table: Extra-Large Sample Scenario Indexing Inputs

Item	Detail	Rationale
Total index data	Up to 2 TB per day	Sample size
Data days in hot and warm buckets	90	Sample input
Data days in cold and frozen buckets	365	Sample input
Min. # of hot and warm indexers	22 medium	Based on daily indexing and hot and warm bucket storage
Min. # of cold and frozen nodes and indexers	34 medium	Based on daily indexing and cold and frozen bucket storage

Table: Extra-Large Sample Scenario Search Inputs

Item	Detail	Rationale
Search users	Up to 96	Sample size
# of search heads	7 medium	Minimum recommended

The following tables contain the extra-large sample scenario platform details and features.

Table: Extra-Large Sample Scenario Platform Details: Splunk Features

Item	Detail	Rationale
Distributed deployment	No	VM HA
Search head pooling	No	Not recommended
Search head clustering	Yes	Preferred approach

Table: Extra-Large Sample Scenario Platform Details: Nutanix

Item	Detail	Rationale
Min. # of nodes	63	Minimum number of nodes
# of clusters	2	Isolated fault domains

Table: Extra-Large Sample Scenario Platform Details: Platform

Item	Detail
Hot and warm series	3000, 8000
Cold and frozen series	6000

The next tables contain the extra-large sample scenario hypervisor details for both options.

Table: Extra-Large Sample Scenario Hypervisor Details: AHV

Item	Detail	Rationale
Cluster size	32	Isolated fault domains
Infrastructure services hosting	N/A	Built into Nutanix platform
Datastores	N/A	Uses iSCSI disks instead

Table: Extra-Large Sample Scenario Hypervisor Details: vSphere

Item	Detail	Rationale
Cluster size	32	Isolated fault domains
Clusters per vCenter	2	Task parallelization

Item	Detail	Rationale
Infrastructure services hosting	Dedicated cluster	Smaller deployment
Datastores	2 Nutanix NFS datastores	Nutanix handles I/O distribution and localization

Additional Sizing Considerations When Running Splunk Enterprise Security

Consult the latest Splunk ES deployment planning guide. Currently, each Splunk ES indexer and each Splunk ES search head requires at least 16 vCPU and 32 GB of RAM. Each Splunk ES indexer can support a maximum of 100 GB of daily ingest.

Data model acceleration can speed up the data modeling of extremely large datasets. This toolset creates data summaries (pivot tables) from the underlying dataset to update and complete items such as reports and dashboards more quickly. Splunk provides data model acceleration using the high-performance analytics store functionality, which builds and stores data summaries on the indexer VMs parallel to the index buckets containing the summarized events.

Consult the latest guide for configuring data models for Splunk ES. Currently, the default settings for data model acceleration require a minimum of 3.4 times the daily ingest rate of SSD space to support this feature. This SSD space is above and beyond what you need to store your hot buckets. The Splunk guide describes how to limit data model acceleration to certain indexes and shorten retention periods to reduce disk space requirements.

Note: Accelerated data model storage per year = data volume per day x 3.4

Splunk ES can also use custom data models. If you use custom models in addition to those provided by the standard Splunk Common Information Model (CIM), you may need to increase the storage requirements.

The following working example shows the various storage requirements for a comprehensive Splunk ES deployment. We based these calculations on feedback from our Splunk customers.

- Assumptions underlying this example Splunk ES environment:
  - 1 TB of raw ingest per day, resulting in 500 GB per day on disk (in other words, 50 percent Splunk compression savings).
  - 10 indexer VMs (1 per 100 GB of ingest), each with 16 vCPU and 32 GB RAM.
  - 30 days hot, 60 days warm, 270 days cold.
  - Data model acceleration enabled with default settings.
- Required SSD space for hot data:
  - Day 1 = 1 TB
  - Day 2 = 1 TB
  - Days 3–30 = 500 GB x 28 = 14 TB
  - Data model acceleration = 3.4 x 1 TB = 3.4 TB
  - Total SSD space in the cluster required for hot data = 1 + 1 + 14 + 3.4 = ~20 TB minimum

This SSD space requirement assumes that the system retains such accelerated data models for the same amount of time as the hot query tier (for example, 30 days). Splunk's deployment guide recommends 3.4 times the daily ingest rate of SSD space for data model acceleration alone, but this allotment doesn't account for other features that could also use the high-performance analytics store.

- Required HDD space for warm data:
  - Days 31–90 = 60 x 500 GB = 30 TB
  - On larger Splunk deployments with an active warm data tier, Nutanix strongly recommends the NX-8155-G7 or NX-8150-G7 platform to ensure that enough HDD spindles are available to serve up IOPS for queries that spill out of the SSD tier.
- Required HDD space for cold data:
  - Days 91–360 = 270 x 500 GB = 135 TB
  - With Nutanix post-process compression enabled (48-hour delay), you can see additional space savings of about 20 percent on cold data, depending on your data and environment.

- Total Splunk ES datastore size = 185 TB

Nutanix Compute and Storage

Nutanix provides an ideal combination of high-performance compute with localized storage to meet any demand. True to this capability, this reference architecture contains zero reconfiguration of or customization to the Nutanix product to optimize for this use case.

The next figure shows a high-level example of the relationship between a Nutanix block, node, storage pool, and container.

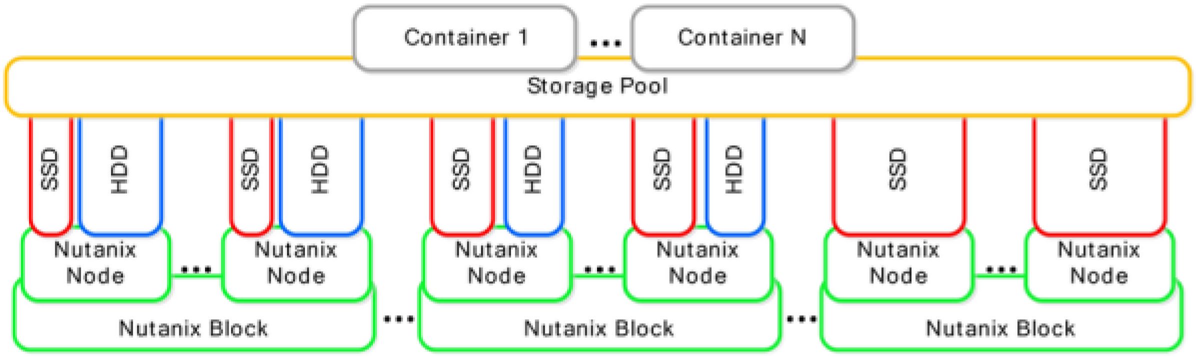


Figure 25: Nutanix Logical Layout

The following table provides the Nutanix storage pool and container configuration.

Table: Nutanix Container Configuration

Name	Role	Details
SP01	Main storage pool for all data	All disks
CTR-VM-DATA	Container for all VMs and OS	VM and index data

Network

Nutanix recommends a leaf-spine network architecture because it's designed for true linear scaling. A leaf-spine architecture consists of two network tiers: an L2 leaf and an L3 spine based on 40 GbE and nonblocking switches. This architecture maintains consistent performance without any throughput reduction because it has a single hop for communication—from any leaf (node) to the spine to any other leaf.

The following figure shows a scale-out leaf-spine network architecture that provides 20 Gbps active throughput from each node to its L2 leaf and scalable 80 Gbps active throughput from each leaf-to-spine switch, providing scale from one Nutanix block to thousands without any impact to available bandwidth.

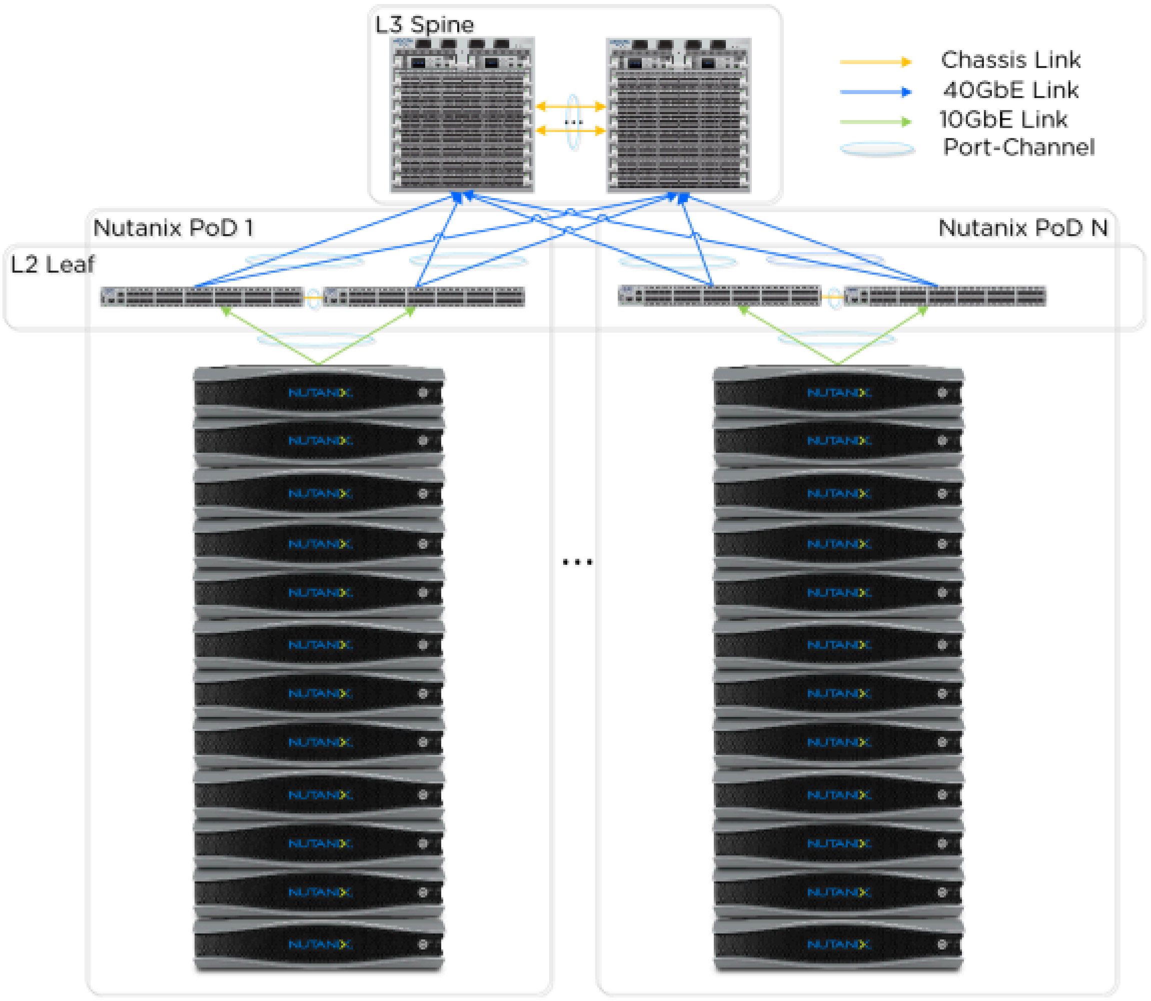


Figure 26: Leaf-Spine Network Architecture

7. Best Practices for Running Splunk on Nutanix

Network Time Protocol (NTP)

Use the Network Time Protocol (NTP) to synchronize the clocks on all nodes and application servers.

On RHEL 7 and later, chrony is the default NTP daemon. The configuration file for chrony is in /etc/chrony.conf on these systems.

Configuring Chronyd (NTP) Service

Based on your geographic location or zone, add the necessary NTP server entries from `https://www.ntppool.org/en/` to `/etc/chrony.conf`. These entries must replace whatever other entries currently exist in the file. Thus, for the North American zone, add:

```
server 0.us.pool.ntp.org iburst
server 1.us.pool.ntp.org iburst
server 2.us.pool.ntp.org iburst
server 3.us.pool.ntp.org iburst
```

Now use the systemd interface to start the chronyd service:

```
# systemctl enable chronyd
# systemctl start chronyd
```

If necessary, allow NTP traffic ingress and egress access to and from the server:

```
# firewall-cmd --add-service=ntp --permanent
success
# firewall-cmd --reload
success
```

You can use the following commands to verify chronyd operation. For further troubleshooting, please refer to documentation available online.

Tracking

```
# chronyc tracking
Reference ID : 208.75.89.4 (time.tritn.com)
Stratum : 3
Ref time (UTC) : Wed Aug 30 12:01:15 2017
System time : 0.000030019 seconds slow of NTP time
Last offset : -0.000030078 seconds
```

```
RMS offset : 0.000167859 seconds
Frequency : 4.386 ppm fast
Residual freq : -0.000 ppm
Skew : 0.019 ppm
Root delay : 0.021557 seconds
Root dispersion : 0.001798 seconds
Update interval : 1038.1 seconds
Leap status : Normal
```

Sources

```
# chronyc sources
210 Number of sources = 4
MS Name/IP address Stratum Poll Reach LastRx Last sample
 ============================================================================
^* time.tritn.com 2 10 377 25m -853us[ -883us] +/- 11ms
^- 104.131.53.252 2 10 377 866 +1581us[+1581us] +/- 71ms
^+ time-b.timefreq.bldrdoc.g 1 10 377 276 +1705us[+1705us] +/- 20ms
^- palpatine.steven-mcdonald 2 10 377 618 +1266us[+1266us] +/- 40ms
```

Virtual Memory Settings

The standard best practice for databases and other big data applications that use the buffer cache is to keep the **vm.dirty_background_ratio** at a low value, so the kernel always flushes and keeps the total dirty buffer memory usage under the **vm.dirty_ratio** to prevent application back pressure.

If you sized the VM compute correctly, you don't need to swap. However, setting swappiness=0 can cause unexpected invocations of the OOM (out of memory) killer in certain Linux distributions. Nutanix recommends setting VM swappiness to 1.

```
vm.swappiness=1
vm.dirty_background_ratio = 1
vm.dirty_ratio = 30
vm.overcommit_memory = 0
```

Add the above changes to file **/etc/sysctl.d/vm.conf** and implement without system reboot with:

```
sysctl --system
```

Disabling Transparent Hugepages (THP)

Hugepages in Linux-based operating systems create preallocated contiguous memory space designed to assist application performance. Transparent hugepages (THP) is a Linux OS feature that conceals much of the complexity of using actual hugepages and automates the creation of contiguous memory space. Only some Linux operating systems enable it by default.

When enabled, THP can significantly degrade overall machine performance on systems that run Splunk Enterprise because of several issues:

- The implementation is too aggressive at coalescing memory pages for short-lived processes (such as many Splunk searches).
- It can prevent the jemalloc memory allocation implementation from releasing memory back to the operating system after use. The jemalloc implementation is a more scalable version of the malloc implementation used in newer distributions of Linux.
- For some workloads, THP can cause I/O regressions surrounding hugepage swapping.

Splunk has observed at least a 30 percent degradation in indexing and search performance on Linux systems where THP is active, with a similar percentage increase in latency. Where possible, disable THP on your Linux system configuration for all machines that run Splunk software, unless that machine also runs an application that requires THP.

Because the Linux OS doesn’t entirely support disabling THP and keeping it off after reboot, establish a process that’s easy to perform and repeat.

Create a systemd unit file with the following contents in **/etc/systemd/system/disable-thp.service**:

```
[Unit]
Description=Disable Transparent Huge Pages (THP)
DefaultDependencies=no
After=sysinit.target local-fs.target
Before=Splunkd.service

[Service]
Type=simple
ExecStart=/bin/sh -c 'echo never | tee /sys/kernel/mm/transparent_hugepage/
enabled > /dev/null'

[Install]
WantedBy=multi-user.target
```

Start the newly created system service:

```
sudo systemctl daemon-reload
sudo systemctl enable disable-thp
sudo systemctl start disable-thp
```

Alternatively, add **transparent_hugepage=never** to the kernel command line (that is, in grub.conf).

Disable zone_reclaim_mode on NUMA Systems

The Linux kernel can be inconsistent in enabling and disabling zone_reclaim_mode, which can lead to performance problems such as:

- Random huge CPU spikes that result in large increases in latency and throughput.
- Programs that stop responding indefinitely.
- Symptoms that suddenly appear and disappear.
- Symptoms that generally don't recur for some time after a reboot.

Ensure that you disable zone_reclaim_mode:

```
$ echo 0 > /proc/sys/vm/zone_reclaim_mode
```

Or add the line **vm.zone_reclaim_mode=0** in file **/etc/sysctl.d/vm.conf** and implement changes without reboot using the following command:

```
sysctl --system
```

Optimize SSDs

The default disk configurations on most Linux distributions aren't optimal. Follow these steps to optimize settings for your solid-state drives (SSDs).

First, ensure that the SysFS rotational flag is set to false (zero). This setting overrides any detection that the OS might attempt to ensure that it considers the drive an SSD. Repeat this step for any block devices created from SSD storage, such as mdarrays.

```
echo 0 > /sys/block/{device_name..ie sdX}/queue/rotational
```

When the target block device is an array of SSDs behind a high-end I/O controller that performs I/O optimization, select the noop scheduler:

```
echo noop > /sys/block/{device_name..ie sdX}/queue/scheduler
```

Set the nr_requests value to indicate the maximum number of read and write requests that can be queued:

```
echo 128 > sys/block/{device_name..ie sdX}/queue/nr_requests
```

You can ensure that all these changes persist after a reboot by using the following udev rules. Create a file **/etc/udev/rules.d/99-nutanix.rules** and add the required entries for disks in the range /dev/sdb to /dev/sdg. Match the range to your own OS disks.

```
ACTION=="add|change", KERNEL=="sd[b-g]", ATTR{queue/scheduler}="noop"
ACTION=="add|change", KERNEL=="sd[b-g]", ATTR{queue/nr_requests}="128"
ACTION=="add|change", KERNEL=="sd[b-g]", ATTR{queue/rotational}=”0”
```

Then, reload the new udev rules without restarting the system using the following commands:

```
sudo udevadm -d control --reload-rules
sudo udevadm trigger --type=devices --action=change
```

8. Validation and Benchmarking

We conducted the solution and testing in this document with Splunk deployed on Nutanix AHV running Nutanix AOS. We used fio (fio-3.7) benchmarks to detail storage performance on the Nutanix appliance.

Environment Overview

The target environment was a Nutanix NX-3460, which provided all Splunk hosting. We connected the Nutanix block to an Arista 7050S top-of-rack switch with 10 GbE. The following testing results capture a specific point in time in terms of Nutanix hardware, software, and hypervisor. Nutanix customers successfully run Splunk in production with versions made available since we completed the original testing.

Test Environment Configuration

Assumptions:

- fio I/O size: 2x memory

Disk configurations:

```
LVM:  6 vDisk stripe setup as follows:
# lvdisplay /dev/spdata/splkdata6
  --- Logical volume ---
  LV Path                /dev/spdata/splkdata6
  LV Name                splkdata6
  VG Name                spdata
  LV UUID                yi5uv5-OvXL-nH8a-HjxP-Qsc4-CJkD-Y7a2fe
  LV Write Access        read/write
  LV Creation host, time Indexer1, 2021-1-1 21:55:10 -0700
  LV Status              available
  # open                 1
  LV Size                1.17 TiB
  Current LE             307194
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     8192
  Block device           253:3
```

Filesystem:

Note: We recommend that you use XFS for production deployments but include filesystem creation options for ext4-only environments.

```
XFS: mkfs -t xfs /dev/mapper/spdata-splkdata6
EXT4: mkfs -t ext4 -m1 -O sparse_super,dir_index,extent,has_journal \
-T largefile /dev/mapper/spdata-splkdata6
```

The **–m1** option with mkfs reduces the superuser block reservation from 5 percent to 1 percent for root.

The **sparse_super** option indicates that backup copies of the superblock and block group descriptors are present only in a few block groups, not all of them.

The **dir_index** option uses hashed b-trees to speed up name lookups in large directories.

The **extent** option allows you to use an extent tree to map logical block numbers for a particular inode to physical blocks on the storage device. Extent tree mapping is a more efficient data structure than the traditional indirect block scheme. Using an extent tree decreases metadata block overhead, improves file system performance, and reduces the need to run any form of file system checker.

The **has_journal** option creates a journal to ensure file system consistency even across unexpected shutdowns.

The **largefile** option ensures the use of one inode per 1 MB.

Mount options:

```
XFS: inode64
```

When you specify inode64, it indicates that XFS is allowed to create inodes at any location in the filesystem

```
EXT4: relatime or noatime, nodiratime
```

The default atime behaviour is relatime, which has almost no overhead compared to noatime but still maintains reasonable atime values. All Linux filesystems use relatime as the default. Using noatime implies nodiratime but we include it here for completeness.

Hardware:

- Storage and compute: 1 Nutanix NX-3460 (2 x Intel® Xeon® CPU E5-2680 v4 @ 2.40 GHz, 28 cores)
- Network: Arista 7050Q (L3 spine) and 7050S (L2 leaf) series switches

Nutanix:

- AOS version: 5.15 LTS
- AHV: 20170830.301

Splunk VM configuration:

- OS: CentOS Linux release 8.1.1911 (Core)
- CPU and memory: 12 vCPU and 16 GB
- Disk:
  - 1 x 40 GB (OS)
  - 6 x 200 GB (data)
- Application version: 8.2.1 (build 545206cc9f70)

fio-3.7:

- Version: fio-3.7
- Data size: 2x memory per thread

Benchmarks

fio Benchmark

fio simulates a desired Splunk I/O workload by writing a job file that describes the specific setup. A job file may contain any number of threads or files, but typically it has a global section that defines shared parameters and one or more job sections describing the jobs involved. When run, fio parses this file and sets everything up as described.

Implementation

- Select a machine to be the captain for the fio testing. The test doesn’t run on this node; it's only a coordinator.
- Distribute or build fio on all non-captain machines running the test and check the version.

- If Splunk is installed on the test machines, stop Splunk services before you run the fio test.
- Start fio on all the non-captain machines using the server switch:

  ```
  fio –server &
  ```
- On the captain machine, create a text file containing a list of all hosts to communicate with. For example: host.list.
- Add each test machine by IP or hostname, one line per host.
- On the captain machine, create a job file to define the test parameters. For example: random_rw.fio.
- On the captain machine, update the job file with the switches that best define the storage parameters you want to use for the test. When testing storage for Splunk Enterprise with fio, the following settings are commonly in a job file:

  ```
  [global]
   ioengine=libaio
   iodepth=16
   bssplit=4k/60:8k/:32k/
   direct=1
   size=<2x RAM size>
   stonewall
   group_reporting directory=/mnt/<volume_name>
  [random-read-write-1]
   rw=randrw numjobs=1
  [random-read-write-many]
   rw=randrw
   numjobs=<CPU core count from OS>
  [sequential-read-write-1]
   rw=readwrite numjobs=1
  [sequential-read-write-many]
   rw=readwrite numjobs=<CPU core count from OS>
  ```

For more information or to download fio, see fio's documentation.

Interpreting the Results

fio Baseline Metrics

Table: fio Baseline Metric Values

Metric	Value	Rationale
Sequential Output—Block	>100 MBps	Affects data input rate

Metric	Value	Rationale
Sequential Output—Rewrite	30–50 MBps	Affects index performance
Sequential Input—Block	>100 MBps	Affects index performance
Random Seeks	>1,200	Affects search performance

Results

fio ran over a range of block sizes relevant to the Splunk workload (bssplit=4k/60:8k/:32k/ - 60% 4k and 20% each 8 and 32k). The results demonstrate the suitability of Nutanix for both index and search workloads on the same host and demonstrate its elegant scalability with the elastic addition of more nodes.

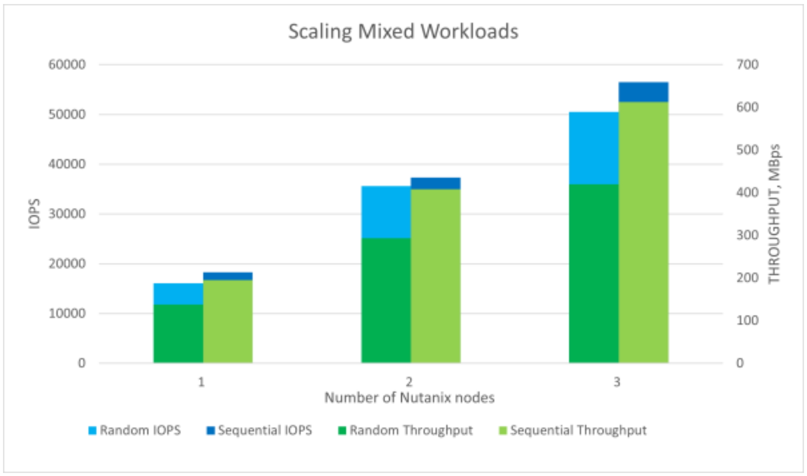


Figure 27: IOPS and Throughput Results for Scaling Mixed Workloads

Most searches are CPU bound. However, the super-sparse, “needle in a haystack” searches so prevalent in Splunk Enterprise Security apps quickly become I/O bound. The sequential reads test results (shown in the following figure) demonstrate the suitability of Nutanix for a wide range of forensic-style workloads, where you must search back over increasing time spans and touch an ever-expanding amount of cold tier data.

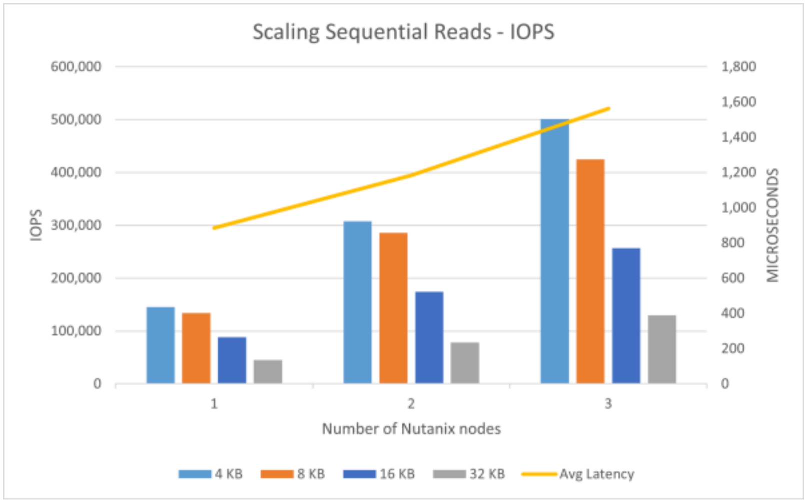


Figure 28: Test Results for Scaling Sequential Reads on Nutanix: IOPS

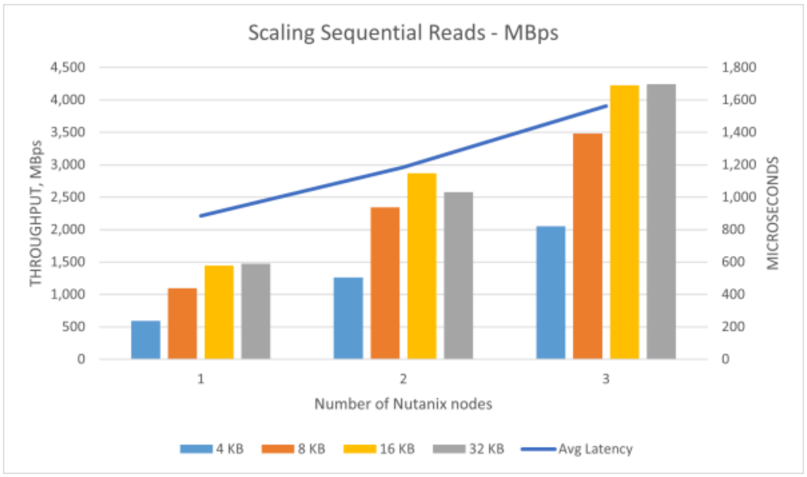


Figure 29: Test Results for Scaling Sequential Reads on Nutanix: Throughput

The sequential writes test results demonstrate that Nutanix can cope with demanding, ingest-heavy workloads that are always I/O bound. To scale, simply add more nodes to the cluster. The graphs show an almost linear scaling profile in terms of both IOPS and throughput.

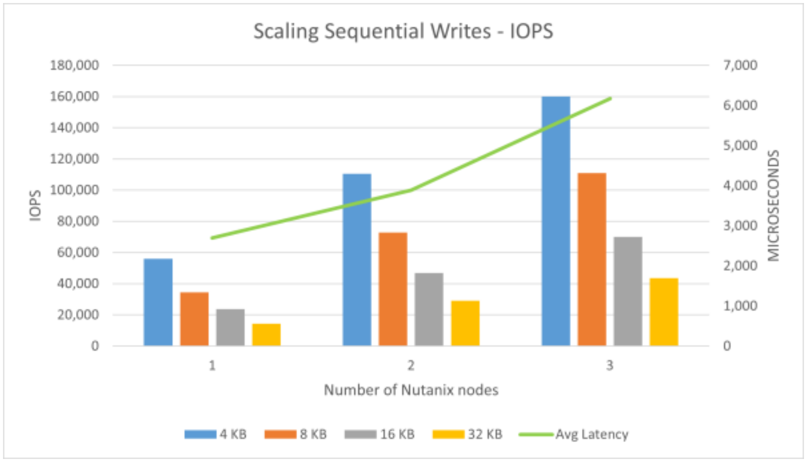


Figure 30: Test Results for Scaling Sequential Writes on Nutanix: IOPS

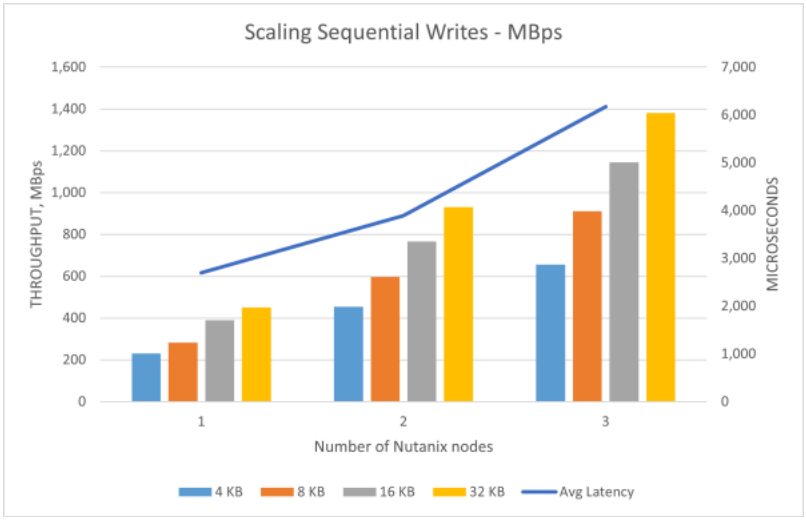


Figure 31: Test Results for Scaling Sequential Writes on Nutanix: Throughput

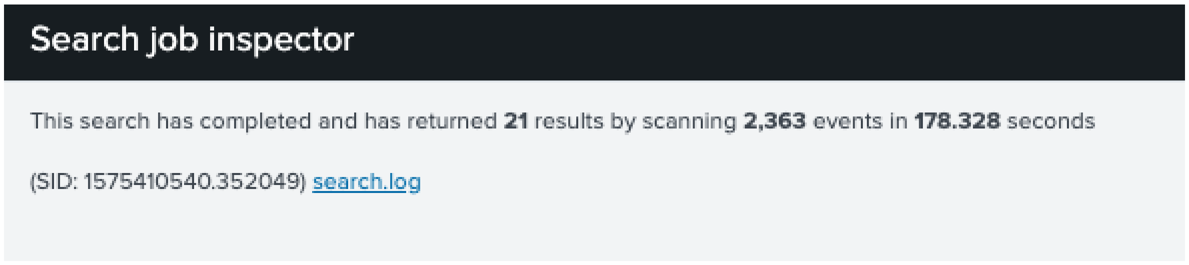
Splunk with Nutanix Objects Results

Nutanix Objects is a SmartStore-certified storage solution that delivers high performance, reliability, and massive scalability. Objects successfully completed all the required tests for both single- and multisite indexers. A Splunk multisite indexer cluster backed by Nutanix Objects in bidirectional replication can withstand site or remote store failure and return consistent search results.

For more information on setting up bidirectional replication, refer to the Objects User Guide.

During remote store performance testing, we configured the local storage in each indexer with 2 TB and the ability to store significant amounts of data locally. The workload ingested around 2 TB of data per day at a rate of 20 MBps with maxKBps = 0 in all universal forwarders, for a total index size of 2.64 TB.

We planted custom search data uniformly in the logs, which were spread across 2,004 days (6 months) as they were real historical Nutanix logs from real Nutanix clusters. We performed various successful searches, both with data in cache and after cache eviction.



Search job inspector

This search has completed and has returned **21** results by scanning **2,363** events in **178.328** seconds

(SID: 1575410540.352049) search.log

Figure 32: Search Job Inspector

We started the test with an ingestion rate of 0.5 TB per day, increased it to 1 TB per day, and finally to 2 TB per day (ingest rate of 20 MBps of data). We didn't notice any issue as we increased the ingestion rate, and all search operations were successful.

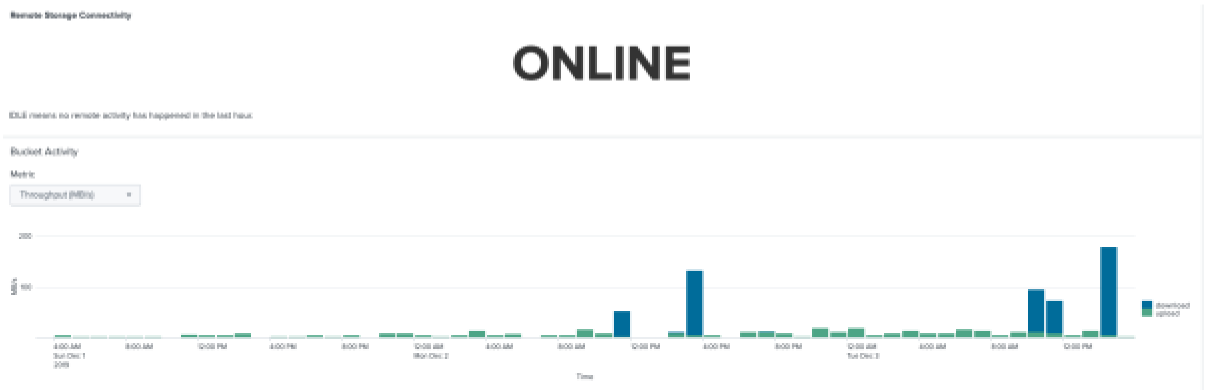


Figure 33: Single-Site Deployment Remote Storage Connectivity

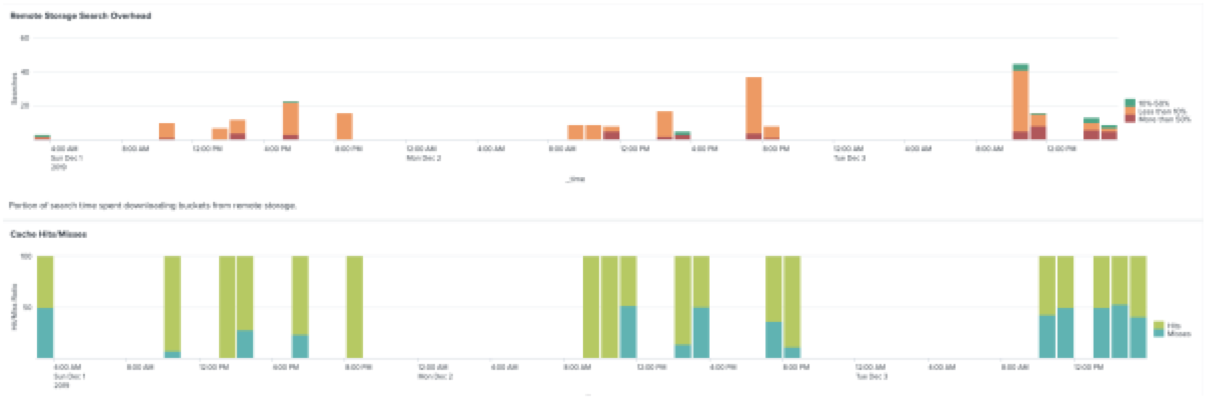
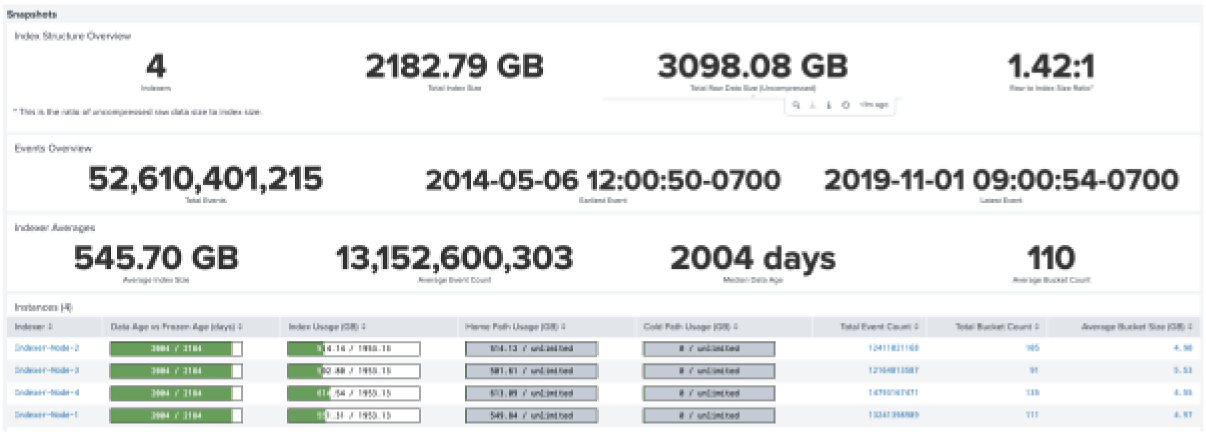


Figure 34: Single-Site Deployment Remote Storage Search Overhead and Cache Hits and Misses

The following image shows the index details for a single-site Splunk with Nutanix Objects solution.



Snapshots

Index Structure Overview

4	2182.79 GB	3098.08 GB	1.42:1
Indexers	Total Index Size	Total Raw Data Size (Uncompressed)	Raw to Index Size Ratio*

* This is the ratio of uncompressed raw data size to index size.

Events Overview

52,610,401,215	2014-05-06 12:00:50-0700	2019-11-01 09:00:54-0700
Total Events	Earliest Event	Latest Event

Indexer Averages

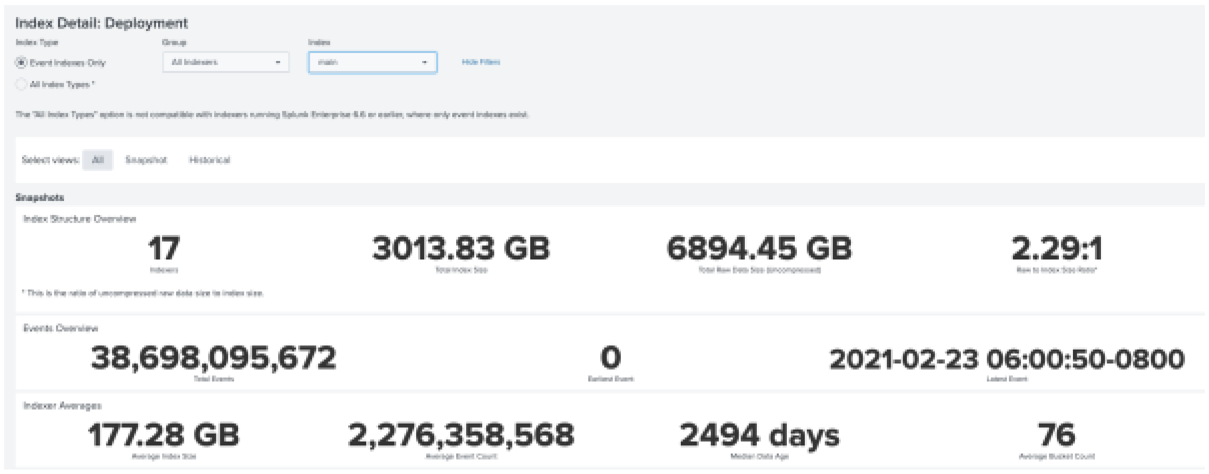
545.70 GB	13,152,600,303	2004 days	110
Average Index Size	Average Event Count	Median Data Age	Average Bucket Count

Instances (4)

Indexer	Data Age vs Frozen Age (days)	Index Usage (GB)	Home Path Usage (GB)	Cold Path Usage (GB)	Total Event Count	Total Bucket Count	Average Bucket Size (GB)
Indexer-Node-2	2004 / 2184	514.16 / 1953.15	514.13 / unlimited	0 / unlimited	12411831160	105	4.90
Indexer-Node-3	2004 / 2184	502.80 / 1953.15	501.81 / unlimited	0 / unlimited	12104813587	91	5.53
Indexer-Node-4	2004 / 2184	614.54 / 1953.15	613.89 / unlimited	0 / unlimited	14790767471	135	4.55
Indexer-Node-1	2004 / 2184	551.31 / 1953.15	549.84 / unlimited	0 / unlimited	13241398589	111	4.97

Figure 35: Single-Site Splunk with Nutanix Objects Index Details

The following image shows the index details for the multisite Splunk with Nutanix Objects solution. Both sites returned identical results.



Index Detail: Deployment

Index Type: (● Event Indexes Only) (All Index Types *)

Group: All Indexers

Index: main

Hide Filters

The "All Index Types" option is not compatible with indexers running Splunk Enterprise 6.6 or earlier, where only event indexes exist.

Select views: All | Snapshot | Historical

Snapshots

Index Structure Overview

17	3013.83 GB	6894.45 GB	2.29:1
Indexers	Total Index Size	Total Raw Data Size (uncompressed)	Raw to Index Size Ratio*

* This is the ratio of uncompressed raw data size to index size.

Events Overview

38,698,095,672	0	2021-02-23 06:00:50-0800
Total Events	Earliest Event	Latest Event

Indexer Averages

177.28 GB	2,276,358,568	2494 days	76
Average Index Size	Average Event Count	Median Data Age	Average Bucket Count

Figure 36: Multisite Splunk with Nutanix Objects Index Details

The following image shows the remote storage search overhead and cache hits and misses for the multisite Splunk with Nutanix Objects solution.

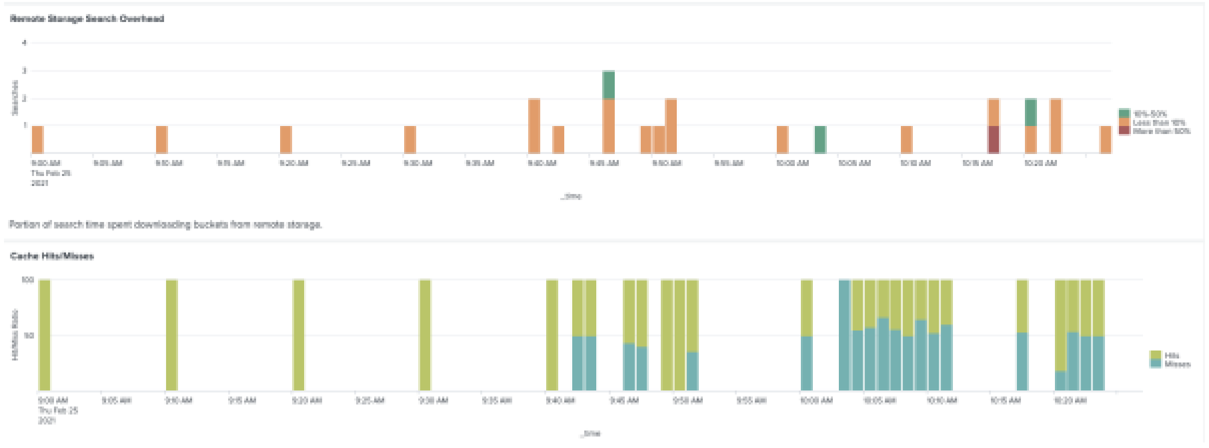


Figure 37: Multisite Deployment Remote Storage Search Overhead and Cache Hits and Misses

We also tested the multisite environment for SmartStore multisite failure and recovery scenarios with stable mode, Site 1 failure, Site 1 recovery, RS1 down, and RS1 up. All test scenarios resulted in success.

The following figure shows Indexer-Node-1 data tiering before the forced data rollover from the hot to the warm tier.

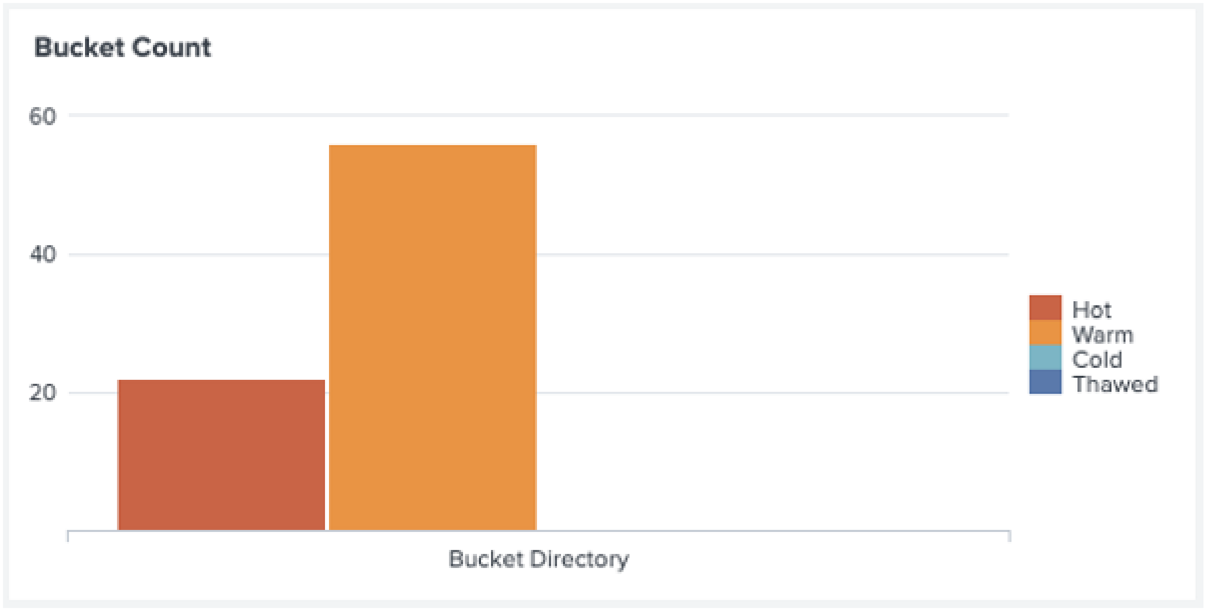


Figure 38: Indexer-Node-1 Data Tiering Pre-Rollover

The following image shows Indexer-Node-1 data tiering after the forced data rollover from the hot to the warm tier.

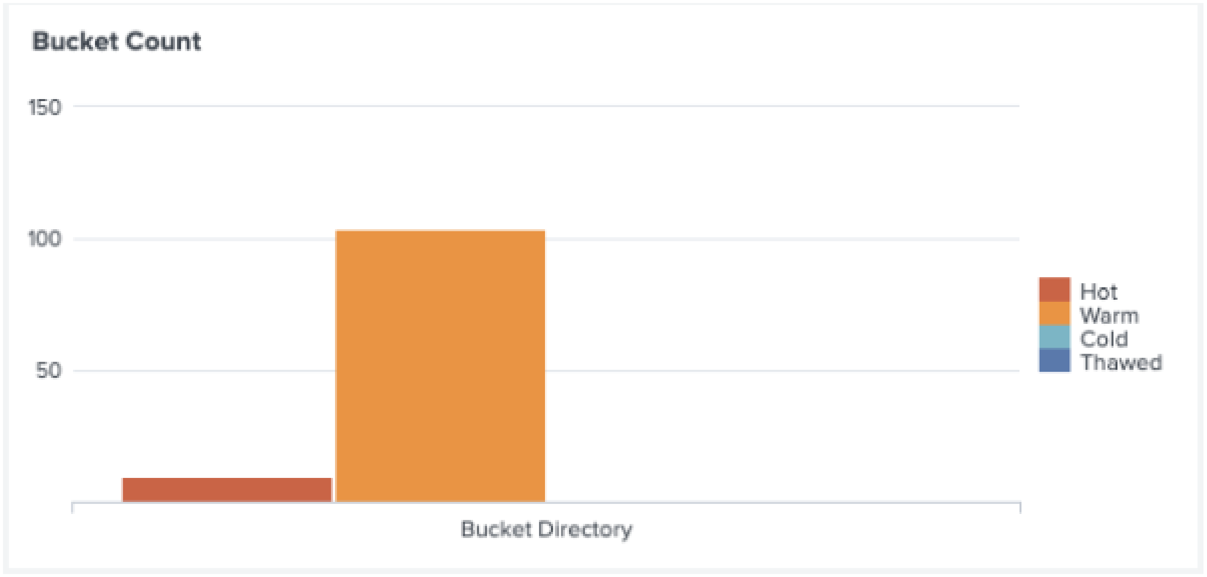
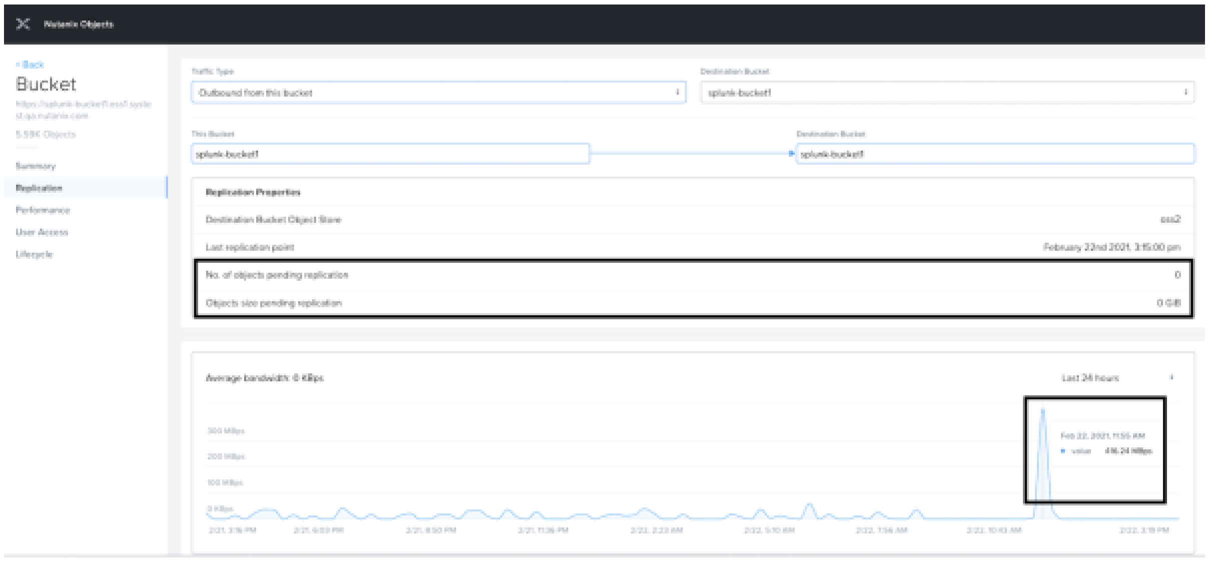


Figure 39: Indexer-Node-1 Data Tiering Post-Rollover

We tested for replication lag (pending replication) and average and peak bandwidth from the object store (RS1).

Note: Nutanix Objects supports NearSync replication between sites.



Replication Properties

Destination Bucket Object Store	oss2
Last replication point	February 22nd 2021, 3:15:00 pm
No. of objects pending replication	0
Objects size pending replication	0 GiB

Average bandwidth: 0 KBps

Figure 40: Nutanix Objects Replication Lag and Bandwidth Results

9. From Bare Metal to Nutanix: A Brief Customer Case Study

The following graphs demonstrate the potential benefits of migrating a Splunk deployment running on bare-metal servers to Nutanix. This customer in the financial technology space ran Splunk for log analysis of machine-generated data and periodically used the Splunk Enterprise Search functionality. They tested a four-node Splunk (6.3.1) indexer cluster running in two different configurations:

- Bare metal: 4 HP Proliant, dual E5-2665
  - Local DAS (24 x 15 KB SAS disks)
  - 16 cores and 32 GB RAM per host
- Nutanix: 1 NX-3460-G4 chassis (four nodes) dual E5-2680v4
  - 2 SSD and 4 HDD per node
  - 12 vCPU and 18 GB RAM per VM

Note: Splunk search metrics summary: Four times faster with 1:1 vCPU-to-CPU, two to three times faster with 2:1 vCPU-to-CPU oversubscription.

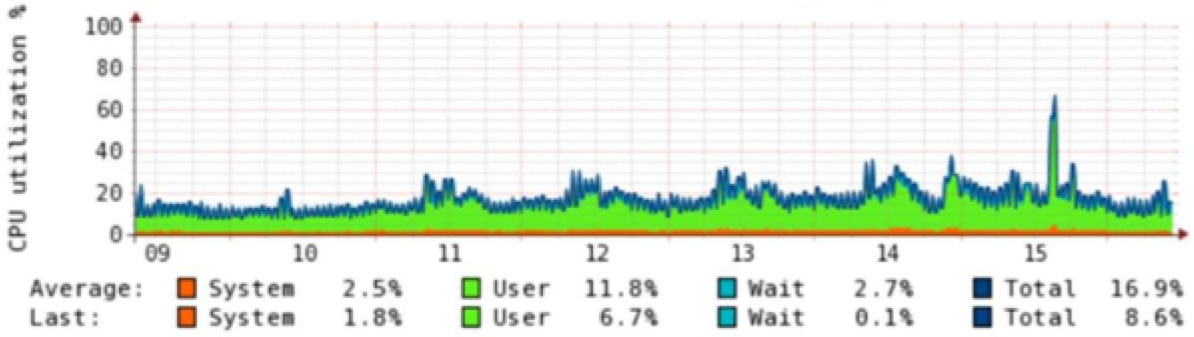


Figure 41: CPU Utilization on Bare-Metal Deployment

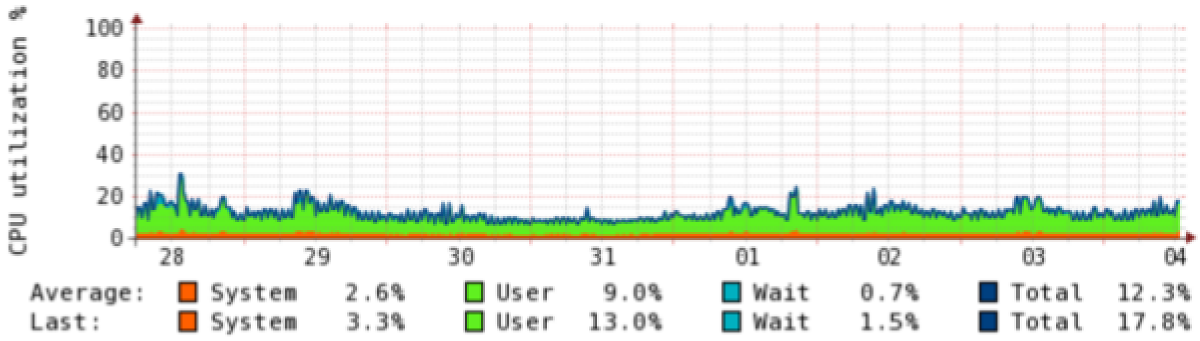


Figure 42: CPU Utilization After Migrating Indexers to Nutanix

The two graphs above show the CPU utilization before and after migration to the Nutanix platform. The CPU utilization is lower and considerably smoother on Nutanix, and the average wait I/O metric is dramatically reduced—by almost four times. The customer interpreted this improvement to mean that a Splunk process could do more actual work when scheduled on a processor.

The next two graphs show the increase in IOPS after migration to Nutanix. The Nutanix tiering algorithms ensure that the Splunk search and indexing processes continually access the SSD hot tier to guarantee the most responsive I/O profile. The difference in results before and after migration is quite stark, with the Splunk processes on Nutanix achieving upward of 16,000 IOPS. Again, this means that running Splunk on Nutanix enables approximately four times the IOPS that running on bare metal does.

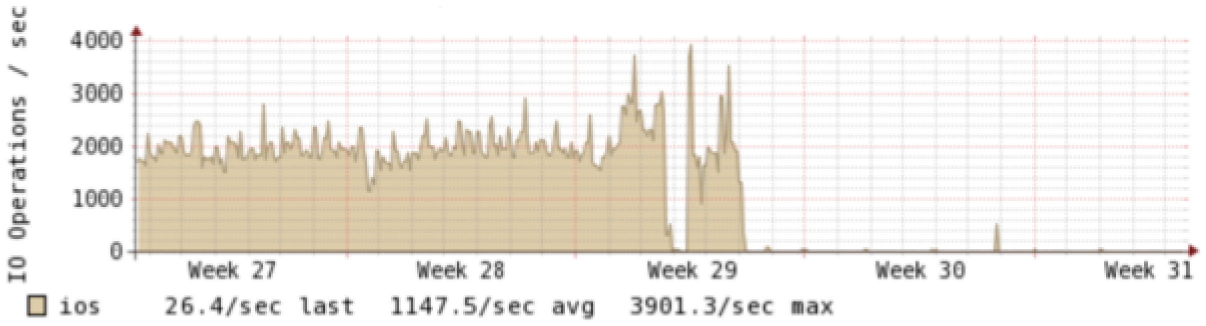


Figure 43: IOPS for Indexer Cluster Running on Bare Metal

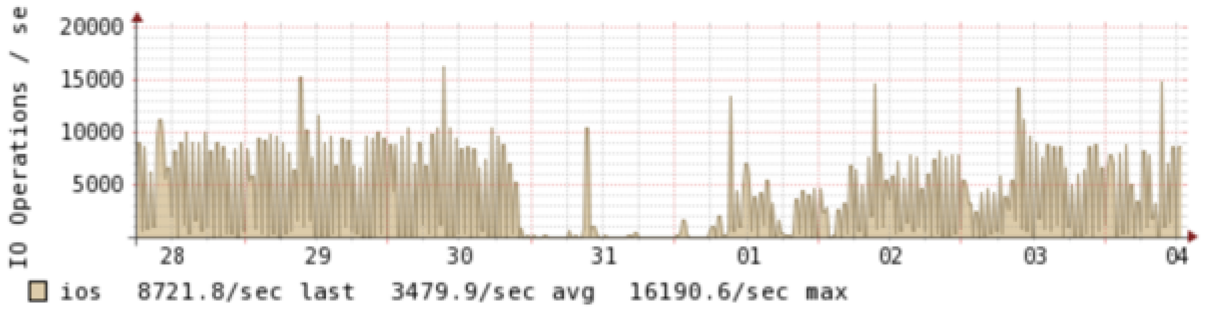


Figure 44: IOPS for Indexer Cluster After Migration to Nutanix

10. Conclusion

With Nutanix, customers can start their Splunk deployments small and then scale out the infrastructure as needed to meet data ingest and retention requirements. The Nutanix cloud platform OS ensures that the system remains available, providing consistent ingest, indexing, and search performance. Administrators can focus on Splunk and their applications, not on the infrastructure, transparently adding compute and storage resources to the cluster as the environment grows.

The Splunk on Nutanix solution provides a single high-density platform for Splunk, VM hosting, and application delivery. Whatever your company’s requirements, there are a range of Nutanix models to choose from.

For Nutanix or Splunk on Nutanix questions, please use our Nutanix Next Community.

11. Appendix

References

1. Splunk Enterprise Distributed Deployment Manual
2. Splunk Enterprise, Multisite indexer cluster deployment overview
3. Splunk Enterprise Security, Deployment planning
4. Nutanix, Controller VM Failure documentation
5. Nutanix, Host Failure documentation
6. Splunk Enterprise, What happens when the manager node goes down
7. Configure data models for Splunk Enterprise Security
8. Splunk Enterprise, The basics of indexer cluster architecture
9. SmartStore architecture overview
10. SmartStore Deep Dive
11. Splunk Architecture: Data Flow, Components and Topologies
12. How to Deploy Splunk SmartStore for Improved Data Storage

About Nutanix

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

List of Figures