COHESITY

Version 1.0

March 2021

Leverage Cohesity's Web-Scale Architecture for Splunk

Use the Cohesity Platform as a Storage Target for Your Splunk SmartStore

ABSTRACT

Cohesity's web-scale architecture provides the ideal platform to use as an S3 target for Splunk SmartStore. This guide helps you implement and configure Splunk SmartStore to use Cohesity as a web-scale storage target.

Table of Contents

Figures

Tables

Intended Audience

This paper is written for Cohesity and Splunk Administrators who plan to configure and use Cohesity as target storage for Splunk SmartStore. Cohesity recommends having familiarity with the following:

- Cohesity DataPlatform
- Splunk Administration

Terminology

The following terms will help you understand the technology, components, and solutions in Cohesity's solution for Splunk SmartStore.

Cohesity Terminology

The most important Cohesity concepts you need to understand to use it as storage for your Splunk SmartStore are Storage Domains, Views, and QoS (Quality of Service) policies.

Table 1: Cohesity Terminology

TERMINOLOGY	DESCRIPTION
Storage Domain	A Storage Domain is a named storage location on a Cohesity cluster. A Storage Domain defines the settings for storage efficiency, data resiliency, and data security. In a Storage Domain, you can create one or more Views, each of which provides NFS, SMB, and S3-compatible mount paths to access the storage.
View	Cohesity serves data to clients and hosts from logical containers called Views. Views are exposed as SMB shares, NFS exports, and/or S3 buckets. Users can modify storage efficiency, choose a QoS policy, and set access permissions at the View level.
View Template	Used to create a View using predefined templates for different use cases and scenarios.

Splunk Terminology

Splunk has specific terminology that is specific to their solution that is highlighted in Table 2 below.

Table 2: Splunk Terminology

TERMINOLOGY	DESCRIPTION
Bucket	Splunk Enterprise stores indexed data in buckets, which are directories containing both the data and index files into the data. An index typically consists of many buckets, organized by age of the data.
Index	The repository for data.
Indexer	A Splunk Enterprise instance that indexes data, transforming raw data into events and placing the results into an index. It also searches the indexed data in response to search requests.
SmartStore	An indexer feature that provides a way to use remote object stores (S3) to store indexed data. By reducing reliance on local storage, SmartStore allows you to scale compute and storage resources separately, thus improving the efficiency of resource usage.
SmartStore Cache Manager	Each indexer incorporates a cache manager that manages the SmartStore data in local storage. The cache manager attempts to maintain in local storage any data that is likely to participate in future searches. By caching the search working set, the cache manager minimizes the potential of search delays resulting from data being downloaded from the remote store.
Search	The cache manager performs these functions:

General Network and S3 Terminology

Other key networking and storage terms you should know include:

TERMINOLOGY	DESCRIPTION
VIP	A VIP is a virtual IP address that can migrate seamlessly from one physical node to another when a node fails in a Cohesity cluster. Once a node failure is resolved, the VIP automatically moves back.
DNS	Domain Name System, a hierarchical and decentralized naming system for computers, services, or other resources connected to the Internet or a private network.
FQDN	A Fully Qualified Domain Name is an absolute domain name including the hostname. A fully qualified domain name is distinguished by its lack of ambiguity, as such it is best to use the FQDN whenever possible instead of a short or relative name. An FQDN is an absolute name that specifies its location in relation to the absolute root of the domain name system.
S3 Bucket	An S3 bucket is a container for storing objects, not to be confused with Splunk buckets.
S3 Access Key ID	An S3 Access Key ID is similar to a User ID and is used in conjunction with an account's Secret Access Key to access S3 resources.
S3 Secret Access Key	An S3 Secret Access Key is similar to a Password and is used in conjunction with an account's Access Key ID to access S3 resources.

Overview

This section highlights and describes the technologies in this solution, as well as the tested and supported software versions. For the purposes of this document, Cohesity will be the target for Splunk SmartStore via the S3 protocol. The benefits and attributes of each solution is also described. Overall, the power of Cohesity SmartFiles and Splunk SmartStore is very compelling. Cohesity provides a powerful and scale out solution as a target while Splunk optimizes the reads and writes.

Cohesity Overview

Cohesity provides a web-scale, software-defined architecture that is a perfect complement to existing and new Splunk deployments.

Unlike alternative solutions that are not well-suited for managing large scale volumes and are hosted on a more expensive tier than direct-attached storage, Cohesity provides web-scale and flexible deployment on-premises or the cloud. In all scenarios, data can still be easily retrieved by Splunk via S3 in order to be searched and analyzed.

Cohesity delivers performance at scale that allows organizations to process data independent of data age or storage placement - keeping all data searchable all the time. Additionally, the non-disruptive upgrade and node failure architecture are ideally suited for Splunk by providing maximum availability around the clock. Cohesity's non-disruptive scale-out architecture makes adding additional capacity simple and seamless.

Splunk SmartStore Overview

SmartStore is an indexer capability that provides a way to use remote object stores, such as Cohesity, 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. These features allow data to reside either locally on indexers or on Cohesity. 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.

SmartStore offers several advantages to the deployment's indexing tier:

- Reduced storage cost. Your deployment can take advantage of the economy of Cohesity, instead of relying on costly local storage.
- Access to high availability and data resiliency features available through Cohesity.

- The ability to scale compute and storage resources separately, thus ensuring that you use resources efficiently.
- Simple and flexible configuration with per-index settings.
- A bootstrapping capability that allows a new cluster or standalone indexer to inherit data from an old cluster or standalone indexer.

Splunk SmartStore Bucket States

The hot buckets of SmartStore indexes reside on local storage, just as with non-SmartStore indexes. Warm buckets reside on remote storage, although copies of those buckets might also reside temporarily in local storage.

The concept of cold buckets goes away, because the need to distinguish between warm and cold buckets no longer exists. With non-SmartStore indexes, the cold bucket state exists as a way to identify older buckets that can be safely moved to some type of cheaper storage, because buckets are typically searched less frequently as they age. But with SmartStore indexes, warm buckets are already on inexpensive storage, so there is no reason to move them to another type of storage as they age.

In all respects, cold buckets in SmartStore indexes are functionally equivalent to warm buckets. The cache manager manages the migrated cold buckets in the same way that it manages warm buckets. The only difference is that the cold buckets, when needed, will be fetched into the cold path location, rather than the home path location.

Tested Components

For the solution and results described in this paper, we deployed the following versions of Cohesity and Splunk software.

Table 3: Components and Versions

COMPONENT	TESTED AND VALIDATED VERSION	SUPPORTED VERSIONS
Cohesity DataPlatform	6.3.1c	6.3.1 and above
Splunk Enterprise	8.0.2	7.3.x, 8.x.x

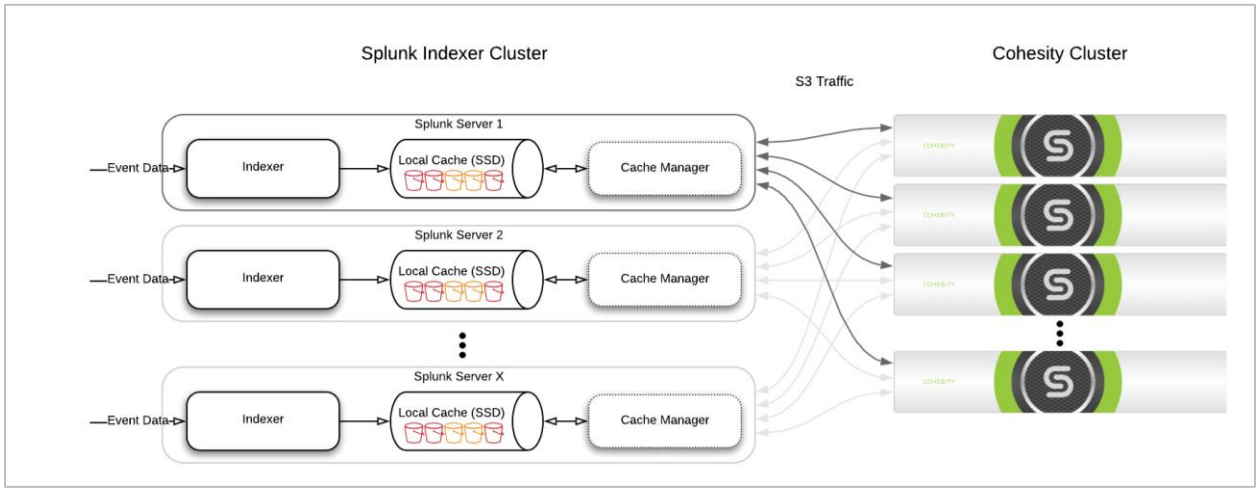
Splunk-Cohesity S3 Traffic/Data Flow

Cohesity's scale-out, shared-nothing architecture allows almost limitless capacity and throughput. As the cluster size increases, so does total available capacity and additional throughput. Each Cohesity node is able to accept and respond to S3 calls with no single points of failure. With the use of VIPs that move as needed within the cluster, downtime due to failures or maintenance/upgrades is eliminated.

Load balancing happens automatically by using a single DNS FQDN that is set up with either a subset or all VIPs for the Cohesity cluster. When Splunk initiates S3 calls, those calls can retrieve the FQDN and all the associated IPs, and then use them. The S3 PUTs and GETs can be sent to any node in the Cohesity cluster. As a result, the load is balanced across all the Cohesity cluster's VIPs. For S3 PUTs and GETs, any Cohesity node will accept those calls, for a single S3 bucket or any number of different S3 buckets.

Figure 1 illustrates how S3 traffic from one or more Splunk servers is distributed across all Cohesity nodes.

Figure 1: Splunk-Cohesity S3 Data Flow



Implementation

The first thing you need to do is set up Cohesity as a target for Splunk SmartStore Indexes. To do so, you'll:

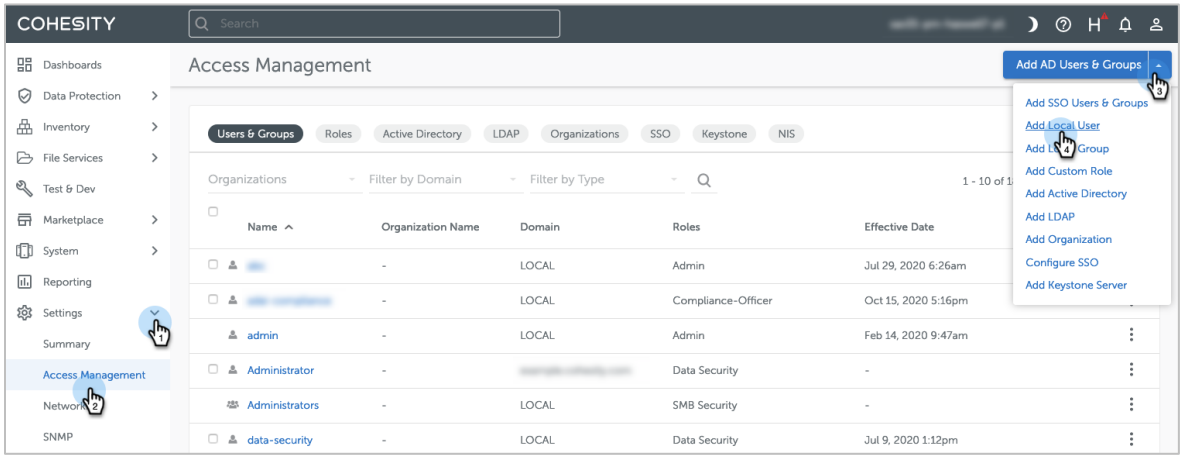
1. Create or identify a Cohesity user.
2. Create or identify a Cohesity Storage Domain.
3. Create a Cohesity S3 View.
4. Convert existing or create new Splunk Indexes to use Cohesity as a SmartStore target.

Create or Identify a Cohesity User

Each Cohesity local user has an associated S3 key pair: an Access Key ID and a Secret Access Key. You will use that key pair for the user that creates the Cohesity S3 View when you configure Splunk SmartStore. You can use the same key pair for different S3 Views to reduce the number of different key pairs you have to manage, or, depending on your business needs, create and use a different user and associated S3 key pair for each of multiple Cohesity S3 Views.

You can choose to use an existing Cohesity user or create a new one. To create a new user specifically for your Splunk SmartStore S3 buckets:

1. Log in to Cohesity as a user with the Admin role.
2. Navigate to **Settings > Access Management** and click **Add AD Users & Groups** and select **Add Local User.**

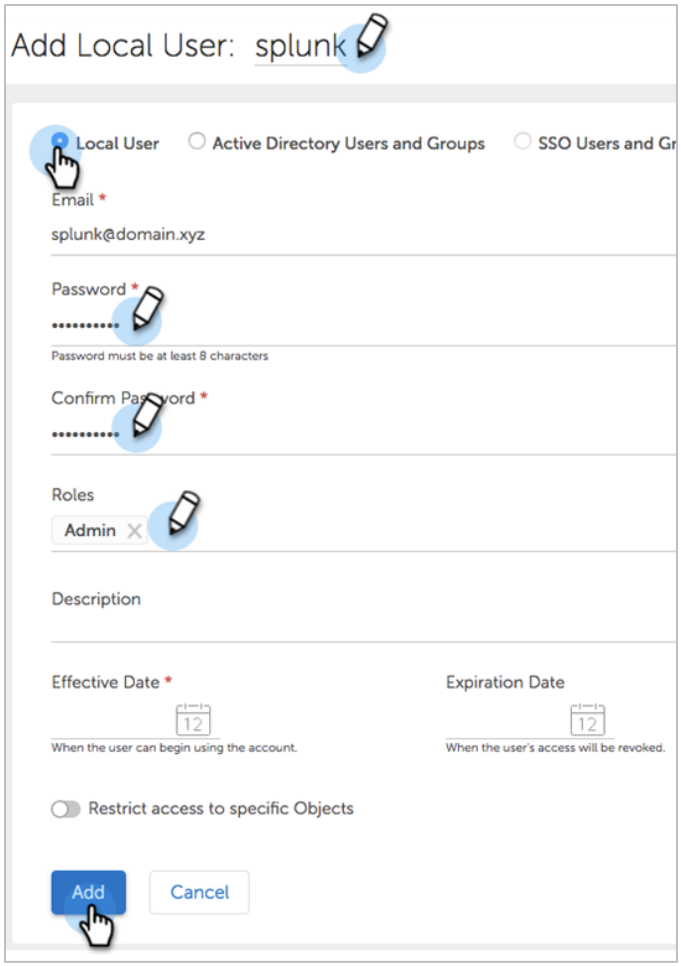


3. In the **Add Local User** form, enter a **Name** and the other details to add a new user.

   Under **Roles,** add the **Admin** role. You can change this later if required, but you need an Admin or similar role to create the Storage Domain (if applicable) and S3 View you'll need. After you create the S3 View, the Cohesity user no longer requires admin privileges to create additional Views within the Cohesity cluster.

   When complete, click **Add**.

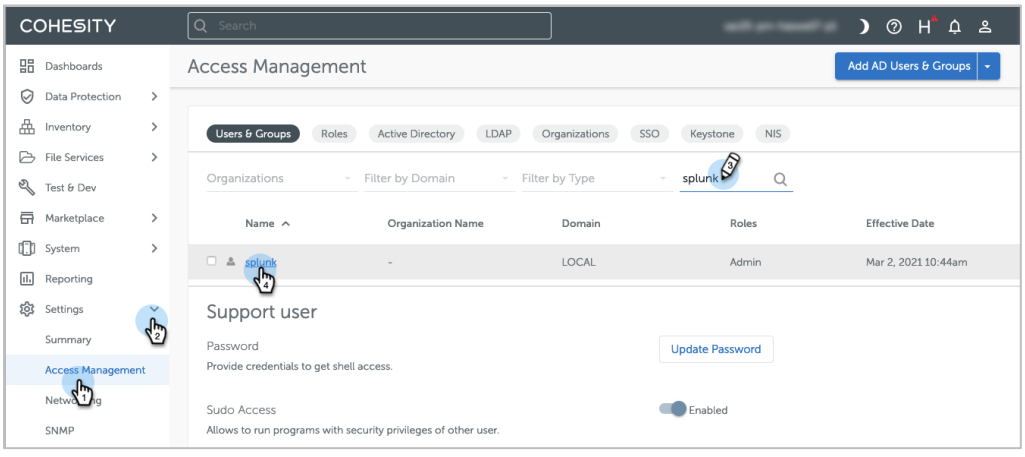
> **NOTE**: The username and passwords are not the same or related to the S3 Access Keys.



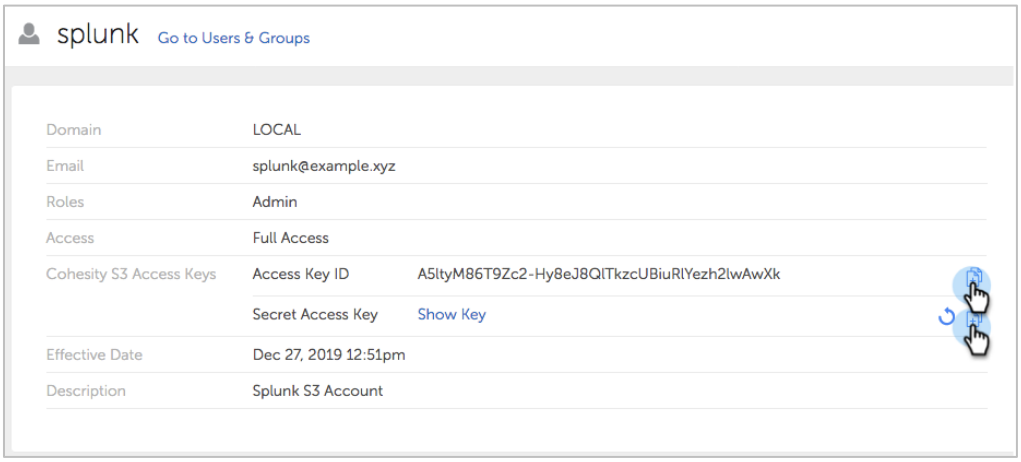
4. Retrieve the new Splunk-specific user account's **Access Key ID** and **Secret Access Key**. Because only that user can access their own S3 Access Keys, you'll need to log out and back in with the new user account. Once you have the user's **Access Key ID** and **Secret Access Key**, you'll use them to [configure the Splunk SmartStore].

   a. Log in to the Cohesity cluster with the user account you just created.

   b. Navigate to **Settings** > **Access Management**, enter a search term for the newly created account, and click the account name.



c. To copy the **Access Key ID** and **Secret Access Key**, click the **Copy** ( ) button for each. To view the **Secret Access Key**, click **Show Key**.



Create or Identify a Cohesity Storage Domain

The next step is to select a Cohesity Storage Domain for the S3 View you will set up as a target for the Splunk SmartStore. You can either use an existing Cohesity Storage Domain or create a new one.

NOTE: Many customers choose to use a single Storage Domain instead of creating a new one for greater global deduplication across workloads. You can even use the existing **DefaultStorageDomain**.

Create a Cohesity S3 View for Splunk

Within Cohesity, SMB shares, NFS shares/exports, and S3 buckets are accessible via Views within a given Storage Domain. To use Cohesity for Splunk, you will create a new Cohesity View for Splunk.

NOTE: For Clustered Splunk Indexers, the S3 View/bucket will be the same for all Indexers for a given index or indexes. Each index can have its own S3 View/Bucket, but across a Splunk index cluster, for a given index or set of indexes, it will be the same.

To create a new S3 View/bucket for your Splunk SmartStore:

1. Log in to the Cohesity cluster with the Splunk user you created.

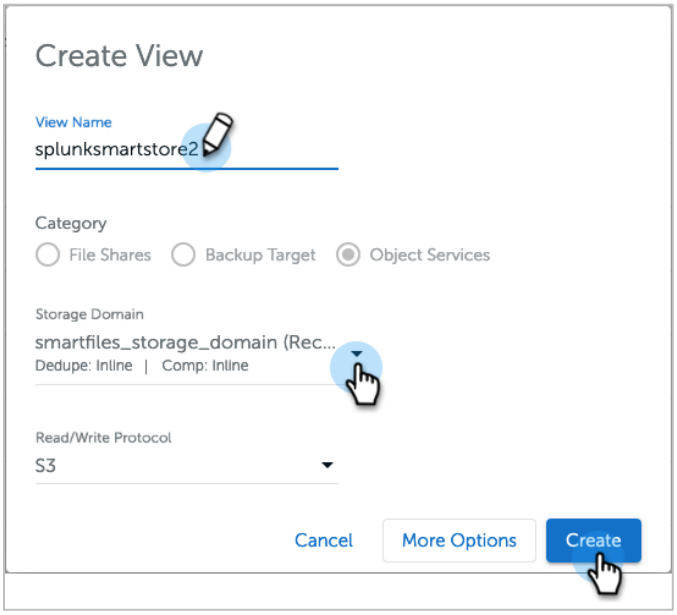
   IMPORTANT: This is critical, as the new S3 View/bucket will be owned by this user and can only be accessed with this user's S3 key pair. For example, we created a new user named 'splunk' previously, but if we now log in as admin or any other account to create the Cohesity View, it won't be accessible via the S3 key pair for the Splunk user you created on Cohesity.

2. Navigate to **File Services > View** and click the **Add** (**+**) button, then select **New View From Template > Splunk SmartStore**. In the Create View form:

   a. Enter a **View Name** for the S3 bucket.

   b. Select a **Storage Domain.**

   Once done, click **Create**.



Now your Cohesity S3 View is ready to be accessed by Splunk.

Convert Existing or Create New Splunk Indexes to use Cohesity as a SmartStore Target

Before beginning, it is essential that you become familiar with the concepts regarding the deployment and configuration of SmartStore. To learn more, see the Splunk documentation in Appendix B: Splunk SmartStore Documentation.

The SmartStore settings within `indexes.conf` enable and control SmartStore indexes. You can enable SmartStore for all of an indexer's indexes or on an index-by-index basis, allowing a mix of SmartStore and non-SmartStore indexes on the same indexer.

NOTE: When you configure these settings on an indexer cluster's peer nodes, you must deploy the settings through the configuration bundle method. As with all settings in indexes.conf, SmartStore settings must be the same across all peer nodes.

In this example, we configure SmartStore for an indexer cluster. The process to configure SmartStore for a standalone indexer is very similar.

The configuration below is an example of an `indexes.conf` file located at `$SPLUNK_HOME/etc/master-apps/_cluster/local` on the master node.

```
[default]
repFactor = auto
remotePath = volume:s3/$_index_name

[volume:s3]
storageType = remote
path = s3://splunksmartstore2/
remote.s3.endpoint = https://<cohesityvip.fqdn>:3000
remote.s3.multipart_upload.part_size = 0
remote.s3.access_key = A5ltyM86T9Zc2-Hy8eP8QlTkzcUBiu8lYezh2lwAwXk
remote.s3.secret_key = Ikfq6ta-f-BITnxScvTVZ5akKDsLHlR9wUKhWLZt_P9

[index1]
homePath = $SPLUNK_DB/index1/db
thawedPath = $SPLUNK_DB/index1/thaweddb
coldPath = $SPLUNK_DB/index1/colddb
```

Table 4: SmartStore indexes.conf Settings and Description

SETTING	DESCRIPTION
`path`	This is calling out the S3 bucket name, which matches the Cohesity S3 View you created.
`remote.s3.endpoint`	This is the DNS VIP name that should already exist for the Cohesity cluster. It should resolve to a subset of or, more likely, all the individual node VIP IP addresses for the cluster. As you can see in the example, we included port 3000. This is required as Cohesity listens on port 3000 for S3 calls.
`remote.s3.multipart_upload.part_size`	For Cohesity clusters less then 6.6, set to `0` in order to disable multipart uploads. Cohesity clusters do support multipart uploads but clusters less than 6.6, see performance degradation when multipart uploads are enabled. For Cohesity clusters greater than 6.6 this setting can be left unset.
`remote.s3.access_key`	This is the S3 Access Key for the Cohesity user that created and owns the S3 View.
`remote.s3.secret_key`	This is the S3 Secret Key for the Cohesity user that created and owns the S3 View.

You can configure SmartStore globally, using the same settings for all indexes, or you can configure SmartStore on a per-index basis. If you configure SmartStore on a per-index basis, you can have a mix of SmartStore and non-SmartStore indexes on the same indexer or indexer cluster. You can also specify different remote volumes for different SmartStore indexes.

To summarize, you can choose from these storage options:

- All indexes stored remotely, on a single volume.
- All indexes stored remotely, on multiple volumes.
- Some indexes stored locally, with others stored remotely on one or more remote volumes.

Because `remotePath` is within the default stanza, all indexes will be SmartStore indexes or migrated to SmartStore. If you only want some indexes to be SmartStore-enabled or migrated, refer to Splunk's documentation in Appendix B.

Once the `indexes.conf` configuration file is ready, deploy it using the normal deployment method.

```
$SPLUNK_HOME/bin/splunk apply cluster-bundle
```

Check the status using the following command:

```
$SPLUNK_HOME/bin/splunk show cluster-bundle-status
```

Monitor SmartStore Status and Bucket Activity

To verify SmartStore connectivity to Cohesity on an individual indexer server, use the following command:

```
$SPLUNK_HOME/bin/bin/splunk cmd splunkd rfs -- ls --starts-with volume:s3
```

The resulting output should look similar to this:

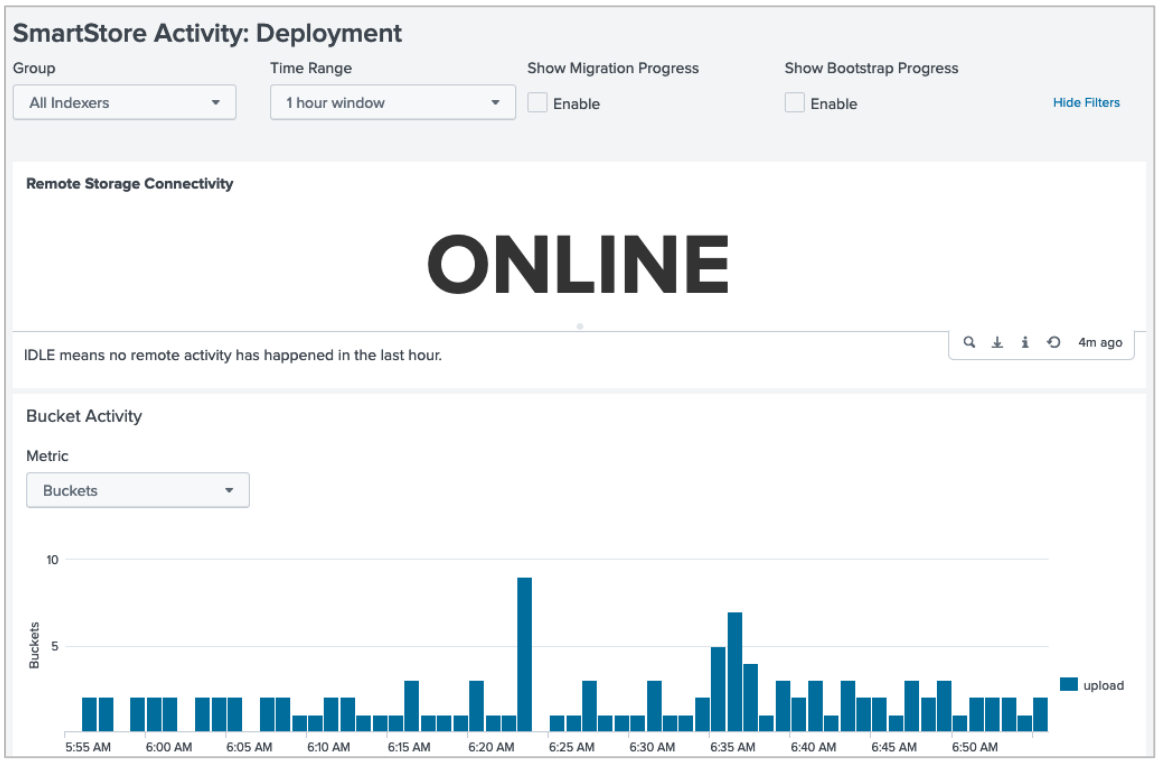
```
size,name
7,_audit/db/04/dd/0~5A667A0B-25E5-47ED-9F77-BFF70EA6C624/guidSplunk-5A667A0B-25E5-47ED-9F77-BFF70EA6C624/.rawSize
537663,_audit/db/04/dd/0~5A667A0B-25E5-47ED-9F77-BFF70EA6C624/guidSplunk-5A667A0B-25E5-47ED-9F77-BFF70EA6C624/1580322589-1580321189-9721689253426813735.tsidx
112,_audit/db/04/dd/0~5A667A0B-25E5-47ED-9F77-BFF70EA6C624/guidSplunk-5A667A0B-25E5-47ED-9F77-BFF70EA6C624/Hosts.data
111,_audit/db/04/dd/0~5A667A0B-25E5-47ED-9F77-BFF70EA6C624/guidSplunk-5A667A0B-25E5-47ED-9F77-BFF70EA6C624/SourceTypes.data
107,_audit/db/04/dd/0~5A667A0B-25E5-47ED-9F77-BFF70EA6C624/guidSplunk-5A667A0B-25


...


284,index1/db/ff/d4/105~9FA05FD7-3595-49D7-AE5B-D0C39F729DB7/guidSplunk-9FA05FD7-3595-49D7-AE5B-D0C39F729DB7/Strings.data
75,index1/db/ff/d4/105~9FA05FD7-3595-49D7-AE5B-D0C39F729DB7/guidSplunk-9FA05FD7-3595-49D7-AE5B-D0C39F729DB7/bucket_info.csv
4334594853,index1/db/ff/d4/105~9FA05FD7-3595-49D7-AE5B-D0C39F729DB7/guidSplunk-9FA05FD7-3595-49D7-AE5B-D0C39F729DB7/rawdata/journal.gz
164552,index1/db/ff/d4/105~9FA05FD7-3595-49D7-AE5B-D0C39F729DB7/guidSplunk-9FA05FD7-3595-49D7-AE5B-D0C39F729DB7/rawdata/slicemin.dat
1603234,index1/db/ff/d4/105~9FA05FD7-3595-49D7-AE5B-D0C39F729DB7/guidSplunk-9FA05FD7-3595-49D7-AE5B-D0C39F729DB7/rawdata/slicesv2.dat
97,index1/db/ff/d4/105~9FA05FD7-3595-49D7-AE5B-D0C39F729DB7/guidSplunk-9FA05FD7-3595-49D7-AE5B-D0C39F729DB7/splunk-autogen-params.dat
2682,index1/db/ff/d4/105~9FA05FD7-3595-49D7-AE5B-D0C39F729DB7/receipt.json
```

This method is ideally suited for checking SmartStore shortly after configuring Splunk SmartStore, as a large number of Splunk buckets have not yet been uploaded to Cohesity. Listing all the uploaded buckets becomes less practical once a large number of buckets have been written. For an overall view of the status and history of a Splunk cluster with respect to SmartStore, we recommend using the **Splunk Monitoring Console** on the **Splunk Master Node**. The Splunk Master Console includes SmartStore Activity details, as well as several other helpful dashboards.

To check the SmartStore activity using the console:

1. Log in to the Splunk Master Node for the cluster.
2. Go to **Settings > Monitoring Console.**

3. Navigate to **Indexing > SmartStore > SmartStore Activity: Deployment.**

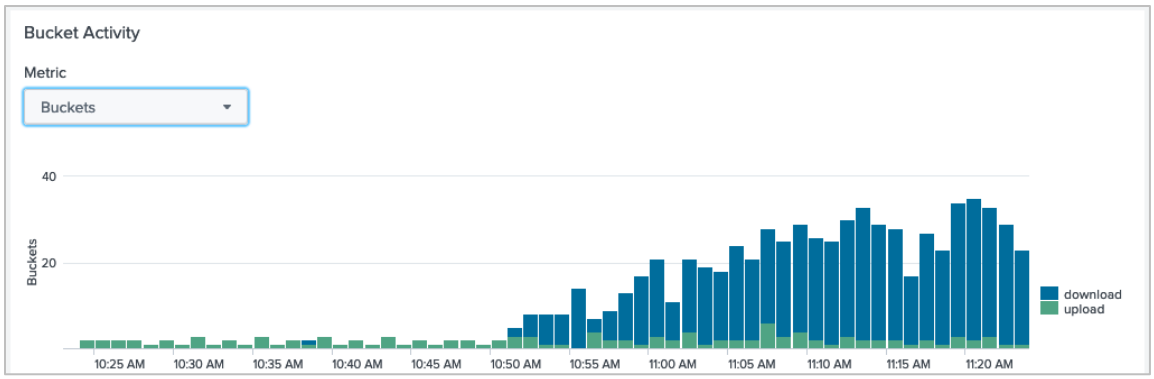


If the configuration is successful, the **Remote Storage Connectivity** status shows ONLINE and the buckets are uploaded. In this case, no current searches require Splunk buckets from Cohesity to be downloaded, as all searches are hitting the indexer's local cache.

If one or more searches are happening and the indexed data to be searched is no longer in the indexer's cache, the cache manager retrieves the data from Cohesity in order to search the data. This data remains in the indexer's cache until the cache manager evicts it.

Figure x below shows an example of downloads from the Cohesity cluster to the indexer's cache needed for the current searches. You can also see the uploads from newly indexed data that is being concurrently copied/written to the Cohesity cluster.

Figure 2: Download and Upload Activity between Splunk Indexer Cache Manager and Cohesity



Appendix A: Splunk SmartStore Considerations and Notes

- The responsibility for high availability and disaster recovery of SmartStore warm buckets shifts from the Splunk cluster to Cohesity. This shift offers the important advantage that warm bucket data is fully recoverable even if the cluster loses a set of peer nodes that equals or exceeds the replication factor in number.
  - When a bucket in a SmartStore index rolls to warm and moves to the Cohesity cluster, the Cohesity cluster takes over responsibility for maintaining high availability of that bucket.
- Indexer cache should be on SSD or similar storage and sized so that the most frequent searches hit the cache.
- Indexer cluster's replication factor and search factor should be equal, for example, 3/3.
- Data retention policy for SmartStore indexes is configured with settings similar to those for non-SmartStore indexes. See Splunk's documentation for specifics.
  - On indexer clusters, data retention for SmartStore indexes is managed cluster-wide. Once a bucket in the index meets the criteria for freezing, the cluster removes the bucket entirely from the system, both from remote storage and from any local caches where copies of it exist.
- Typically, there is no need to change the SmartStore maximum download and upload rates/threads, but with very large Splunk and Cohesity deployments, it might be advantageous to increase these.
  - The `max_concurrent_downloads` setting in `server.conf` specifies the maximum number of buckets that can be downloaded simultaneously from remote storage. Its default is 8.
  - The `max_concurrent_uploads` setting in `server.conf` specifies the maximum number of buckets that can be uploaded simultaneously to remote storage. Its default is 8.

Appendix B: Splunk SmartStore Documentation

Learn more about Splunk SmartStore and its various configurations from the Splunk documentation:

- About Splunk SmartStore
- SmartStore architectural overview
- SmartStore settings (indexes.conf)
- Spec and example of indexes.conf (SmartStore and non-SmartStore specific settings)
- Configure data retention for SmartStore indexes
- The SmartStore cache manager
- Troubleshoot SmartStore

Your Feedback

Was this document helpful? Send us your feedback!

About the Authors

Justin Willoughby is 20-year IT veteran, currently working for Cohesity as a Solutions Engineer. In this role, Justin architects, builds, tests, and validates business-critical applications, databases, and virtualization solutions with Cohesity's DataProtect platform.

Document Version History

VERSION	DATE	DOCUMENT HISTORY
1.0	Mar 2021	Original document

ABOUT COHESITY

Cohesity radically simplifies data management. We make it easy to protect, manage, and derive value from data -- across the data center, edge, and cloud. We offer a full suite of services consolidated on one multicloud data platform: backup and recovery, disaster recovery, file and object services, dev/test, and data compliance, security, and analytics -- reducing complexity and eliminating mass data fragmentation. Cohesity can be delivered as a service, self-managed, or provided by a Cohesity-powered partner.

Visit our website and blog, follow us on Twitter and LinkedIn and like us on Facebook.

2000041-001-EN