COHESITY

Cohesity Gaia Self-Managed Deployment Guide

Deploy Cohesity Gaia Self-Managed on your own infrastructure — Secure, Compliant, and Fully in your Control

Version 1.1

February 2026

Abstract

Cohesity Gaia is a cutting-edge generative AI application developed by Cohesity, designed to empower end users with the ability to safely, securely, and responsibly access and analyze data within Cohesity Data Cloud.

Cohesity Gaia Self-Managed is purpose-built for organizations that need the full power of Cohesity Gaia right within their own data centers. With this fully self-contained deployment, your sensitive data remains securely on-premises, helping you meet the privacy, governance, and regulatory requirements. This guide outlines the necessary steps to deploy and configure the environment, ensuring a comprehensive assessment of the solution's capabilities.

Table of Contents

Introduction

In today's rapidly evolving IT landscape, organizations increasingly seek greater control over their infrastructure to meet regulatory, security, and data sovereignty requirements. For industries with stringent mandates—such as government, defense, and healthcare—operating in disconnected or "dark site" environments is often essential. These environments demand full control over data and systems, without reliance on external networks or third-party management.

Cohesity Helios Self-Managed addresses this need by offering a **single pane of glass control plane** that can be deployed entirely on-premises. This customer-managed version of Helios empowers organizations to oversee all aspects of their data management infrastructure locally ensuring compliance, minimizing risk, and maintaining operational continuity in air-gapped or restricted environments. By transitioning from the legacy Cluster UI to the modern, unified Helios interface, customers gain centralized management, automation, and observability—delivered securely and entirely under their control.

A major step forward in this evolution is the introduction of Cohesity **Gaia in Helios Self-Managed**. Previously available only in Helios SaaS, **Cohesity Gaia** is a powerful generative AI application that uses Retrieval-Augmented Generation (RAG) to enable secure, natural language querying of backup and unstructured data stored within the Cohesity Data Cloud. Gaia helps organizations derive actionable insights and make faster, smarter decisions—without compromising data sovereignty or compliance.

Gaia, now part of Helios Self-Managed, introduces AI capabilities for the most secure and isolated environments—while maintaining strict governance and control requirements.

Core Features of Gaia as part of Helios Self-Managed:

- **Role-Based Access Control (RBAC):** Fine-grained permissions ensure users only access data they're authorized to access.
- **Inclusions and Exclusions:** Users can specify which file types and directories to include or exclude. This enhances data security by preventing exposure of sensitive content and improves access control.
- **Continuous Indexing**: Data ingested into the platform is automatically and continuously indexed, enabling real-time discovery and AI-powered insight extraction.
- **Language Support**: Cohesity Gaia Self-Managed supports English and Dutch, with additional languages planned for future releases. Please refer to product docs for more details.
- **Natural Language Querying with RAG**: Users can interact conversationally with backup data, gaining contextual answers from structured and unstructured sources. You can index the data directly in its original language. Once the dataset is created, you can ask questions in any of the supported languages and Cohesity Gaia will respond in the same language, providing a seamless and natural interaction.

  While Cohesity Gaia handles your data and queries in multiple languages, the user interface (UI) options, error messages, and system prompts will remain in English. Cohesity Gaia Self-Managed enables organizations to use AI responsibly and securely by integrating with enterprise-ready LLMs.
- Visualize **Your Datasets:** Cohesity Gaia uses cutting-edge AI and natural language processing to create visual representations of data during indexing, offering unprecedented insights into complex datasets.

- **Citations:** Cohesity Gaia includes source references within its responses, enabling you to view and download relevant sources. By hovering over a citation, you can quickly view the corresponding text reference from the associated file. This feature facilitates easy verification of information.

With Gaia in **Helios Self-Managed**, Cohesity continues to advance its vision of modern, intelligent data management—delivered wherever the customer needs it, securely and at scale. The following documents and sources are either directly referenced or have informed key sections of this solution guide:

- Helios Self-Managed Documentation
- Cohesity Gaia Security White Paper
- Cohesity Gaia SaaS Documentation

Solution Architecture Overview

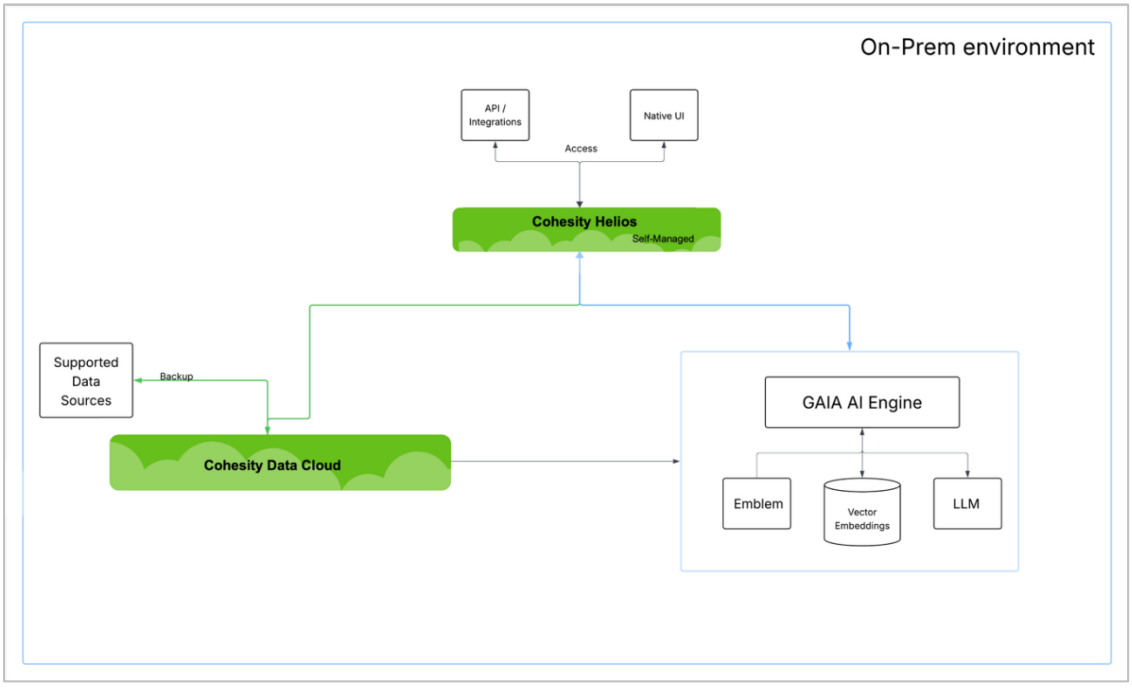
Cohesity Gaia Self-Managed is purpose-built for organizations that need the full power of Cohesity Gaia right within their own data centers. With this fully self-contained deployment, your sensitive data remains securely on-premises, helping you meet the privacy, governance, and regulatory requirements.

Cohesity Gaia Self-Managed can be deployed using your existing compute, storage, and networking infrastructure. It supports seamless integration with container orchestration platforms such as Red Hat OpenShift, enabling efficient utilization of current IT resources. The following are a few key benefits:

- Leverages existing hardware and software investments, reducing the need for additional provisioning.
- Compatible with Kubernetes-based environments for streamlined deployment and management.
- Keeps data within your network boundary, eliminating the need for WAN-based data transfers.
- Enables indexing and analysis directly at the data source, improving performance and reducing latency.

Cohesity Gaia Self-Managed has 3 main components which work closely together to provide you insights into your data, by utilizing modern encryption methods, such as mutual Transport Layer Security (mTLS) and HTTPS during transit. These secure communication channels ensure that data remains confidential and protected while being transmitted between components and services within the Cohesity Gaia system.

1. **Gaia AI-Engine**: It is the main indexing, query and answer engine composing of multiple services, which performs embeddings enabling users to interact with a given dataset.
2. **Cohesity Helios Self-Managed**: Helios, deployed in self-managed mode, offers centralized fleet management, reporting, and anomaly detection capabilities all under your control.
3. **Cohesity Data Cloud**: Cohesity Data Cloud connects your sources to Back up the data and works with Helios Self-Managed and Gaia-AI-Engine to provide insights in a secure way.



How Does Cohesity Gaia Work?

Cohesity Gaia harnesses the power of RAG and revolutionizes the way users interact with their data, providing an intuitive and intelligent interface for extracting valuable insights from vast amounts of data. By combining advanced search capabilities with NLP, Cohesity Gaia enables users to ask questions and receive detailed, accurate answers, facilitating more informed decision-making and enhancing overall productivity.

The table below describes the query workflow:

Table 1: Query Workflow

Backups	Establish a connection to data within your backups
Ingest	Create datasets
	Extract text from diverse document types automatically
Index	Generate embeddings by vectorizing data, essential for anchoring answers using your enterprise data
	Store the produced vectors in a specialized database optimized for multidimensional vector search
Retrieve	Retrieve relevant data
	Provide relevant retrieved data as context to foundational models
Large Language Models	Generate answers based on the user query and context Models
Response	Provide insightful responses along with citations

Installation Requirements and Prerequisites

Ensure that the prerequisites outlined in Cohesity Gaia Self-Managed Documentation and Helios Self-Managed Documentation are satisfied before proceeding with the Cohesity Helios Self-Managed and Gaia deployment.

Table 2: Installation Requirements and Prerequisites

Requirement Type	Helios Self-Managed	Cohesity Gaia Self-Managed
Hardware	Hardware Requirements	Hardware Requirements
Software	Software Requirements	Software Requirements
Networking	IP Address & Hostname Requirements	IP Address & Hostname Requirements
Firewall Ports	Firewall Ports	Firewall Ports

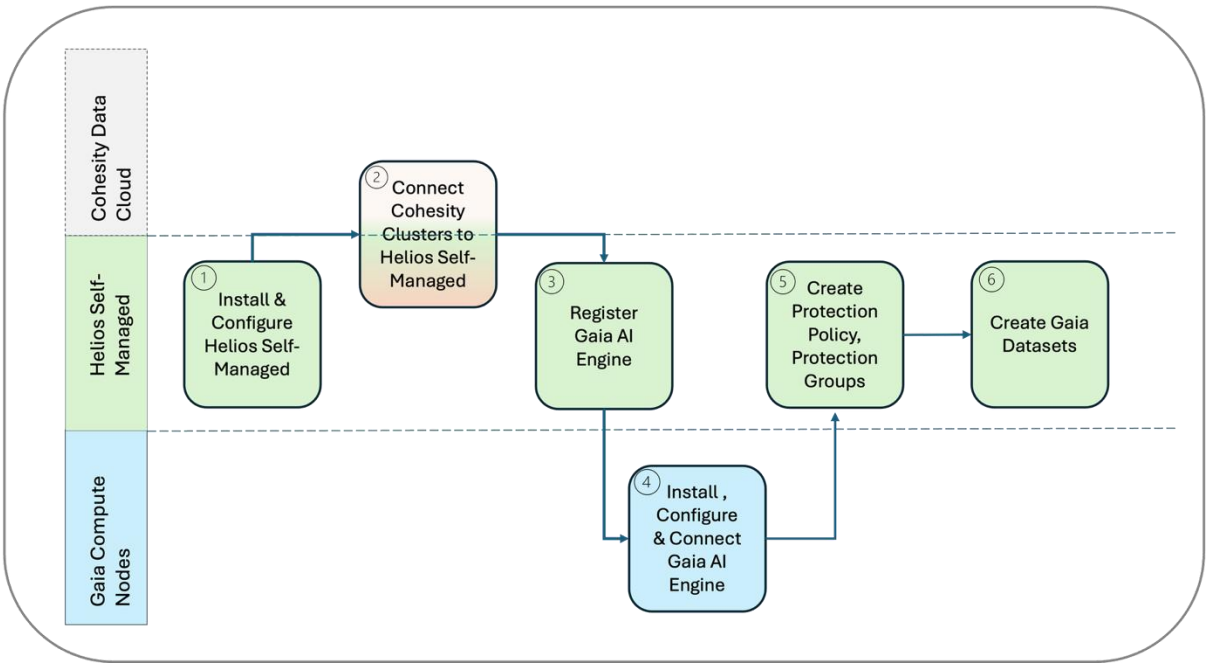
NOTE: For optimal performance, it's essential to have Helios Self-Managed nodes and Gaia AI Engine nodes in close network proximity. We recommend placing them in the same subnet to minimize hops and enhance system efficiency.

Solution Deployment

Deploying the Cohesity Gaia Self-Managed, involves several steps across all the 3 components.

In a high level, the steps are as below:

1. Install and Configure Helios Self-Managed.
2. Connect Cohesity Clusters to Helios Self-Managed.
3. Register Gaia AI Engine.
4. Install, Configure & Connect Gaia AI Engine on Gaia Compute Nodes.
5. Create Protection Policy, and Protection Groups to Protect Sources.
6. Create Gaia Datasets for the Protection Groups.



Step 1: Helios Self-Managed - Install & Configure

Setting up the Helios Appliance is the initial step for a customer-managed Helios deployment. Ensure that you complete the installation as described in the Helios Self-Managed Documentation before continuing.

Once Helios Self-Managed is installed, we will proceed to connect Cohesity Clusters in the next step.

Step 2: Connect Cohesity Clusters to Helios Self-Managed

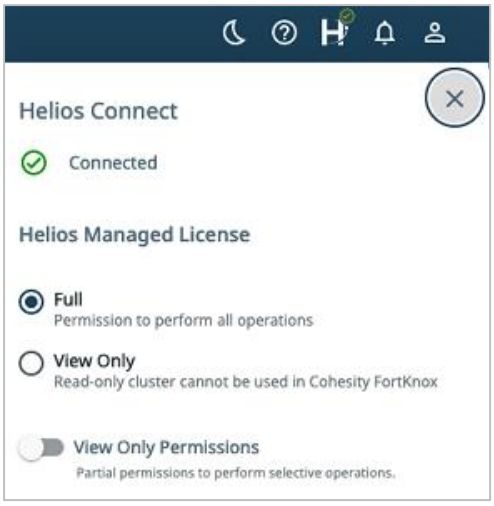
To manage your clusters in Helios you have to first follow the steps to connect your clusters to Helios.

You can connect clusters to Helios seamlessly using a token-based approach. This is an intuitive and secure method for claiming clusters. You can claim a Cohesity cluster through a token generated in Helios. Helios validates this token and claims the cluster if it passes all checks. The streamlined process reduces complexity and enhances user efficiency, allowing for smoother cluster management.

Connecting Cohesity Clusters to Helios Self-Managed has below steps:

1. Generate a Token in Helios Self-Managed.
2. Apply the token on the Cohesity Cluster to connect it to Helios Self-Managed.

   After successfully connecting the cluster to Helios, a green check mark appears on the Helios icon and the **Helios Connect** side panel status changes to **Connected**. This visual indication confirms that the connection between the cluster and Helios has been established and is active.



   In **Cohesity Cluster** UI, navigate to **Settings > Cluster Management**, and check whether the cluster is listed on the page.

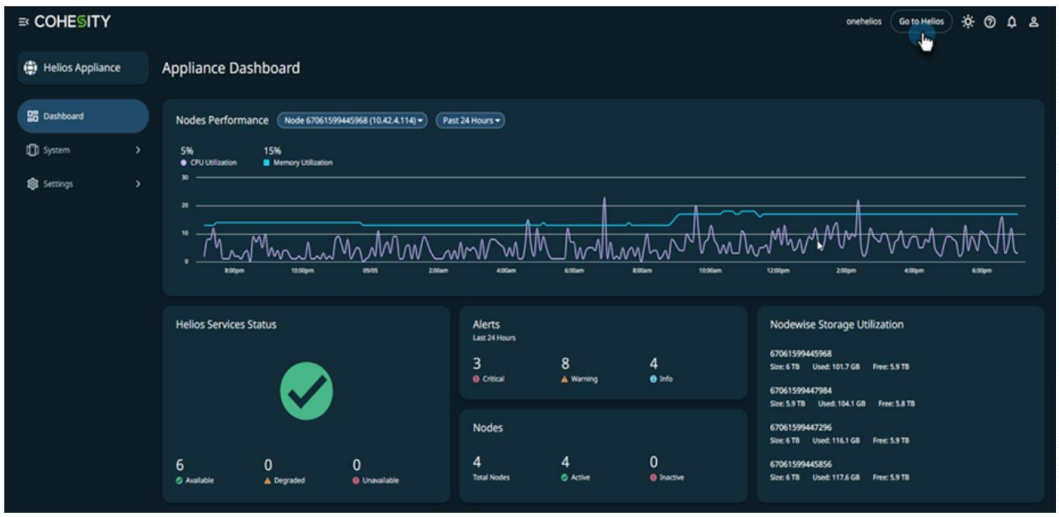
Step 3: Register Gaia AI Engine in Helios Self-Managed

After deploying and configuring your Helios Self-Managed and connecting Cohesity Clusters to Helios, the next step is to Register Gaia AI Engine.

You can sign in to Helios through the Helios Appliance or the Helios FQDN.

To log in to Helios through the Helios Appliance:

1. Log in to the Helios Appliance.
2. In the Helios Appliance user interface, click **Go to Helios** in the upper-right corner. You are redirected to the Cohesity Data Cloud Sign In page.



3. Enter your credentials and click **Sign In**.

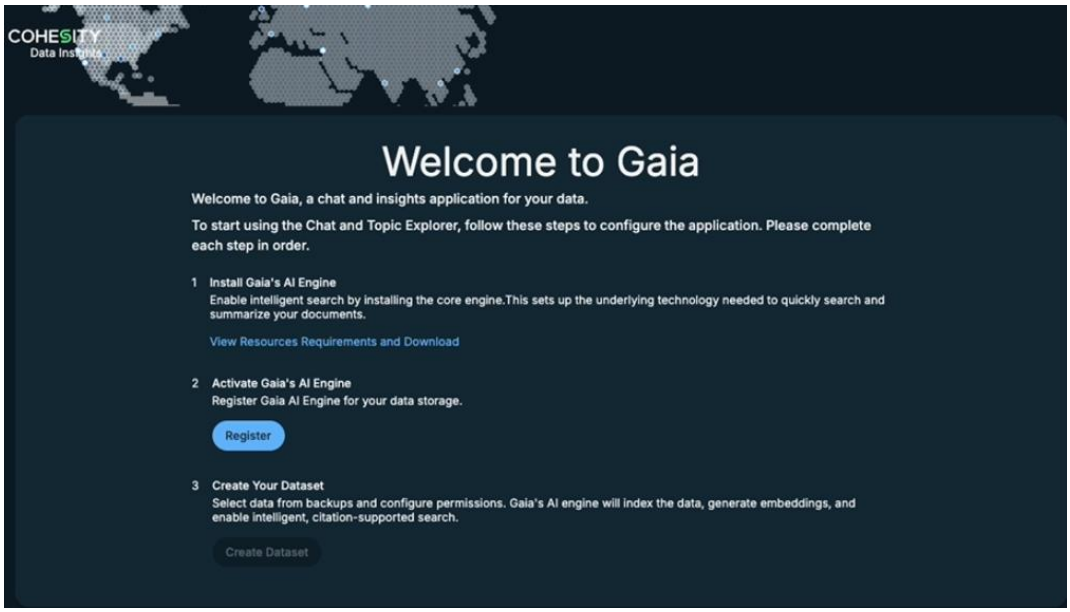
To log in to Helios through the Helios FQDN:

1. Open your web browser and enter the FQDN for Helios.
2. Enter your credentials and click **Sign In**.

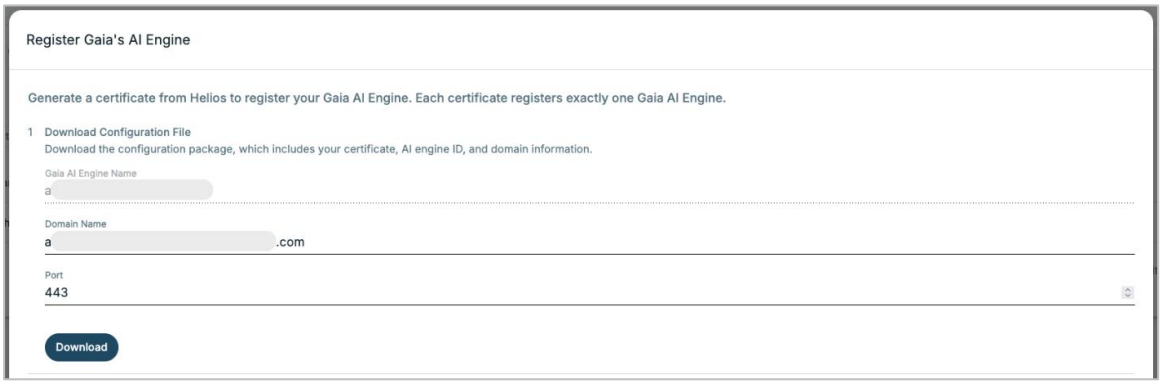
> **NOTE**: You can also sign in using Active Directory or Single Sign-On. For more information, see Helios Self-Managed documentation.

Step 3.1. Register Gaia AI Engine in Helios Self-Managed

1. Log in to Cohesity Helios Self-Managed.
2. Click the app-selector menu and navigate to **Insights > Data Insights.** The Gaia onboarding page is displayed:



3. Click **Register** and on the **Register Gaia's AI Engine** page, enter the following details and click Download:
   - **Gaia AI Engine Name**: Enter the name of the Gaia AI Engine. This name is displayed in the **Gaia Region** drop-down list when creating a dataset.
     Ex: gaia-ai-engine
   - **Domain Name**: Enter the fully qualified domain name.
     Ex: gaia-ai-engine.cohesity.com
   - **Port Number**: Cohesity recommends that you use port number 443.



4. Download the **Configuration File** for Gaia AI Engine

   The configuration file is downloaded in the JSON format.
   - Transfer this JSON file to the gaia-ai-engine directory mentioned in the next section. This configuration file contains essential values and certificates required to establish the connection.
   - Transfer the S3 certificate, which is used to authenticate with the S3 endpoint, to the gaia-ai-engine directory.

Step 4: Install, Configure & Connect Gaia AI Engine

In this section, we will focus on the Gaia AI Engine Installation and Configuration. Before proceeding to install Gaia AI Engine, ensure the following components are set up and configured.

Step 4.1 Set Up the Red Hat OpenShift Cluster

- Deploy and configure a **Red Hat OpenShift cluster**, using the preferred installation method for on-premises hardware.

  Please refer to the Software Requirements for the Red Hat OpenShift Cluster version.

Step 4.2 Install and Configure NVIDIA GPU Drivers Support

- Deploy the **Node Feature Discovery (NFD) Operator** and create a Feature Discovery Instance. For more information, see Installing the Node Feature Discovery Operator on OpenShift.
- Deploy the **NVIDIA GPU Operator**. If the nodes are connected to the internet, the operator automatically installs NVIDIA drivers. Otherwise, manually install the drivers on each node. For more information, see Installing the NVIDIA GPU Operator.

  Please refer to the Appendix-I for a sample set of installation steps.

Step 4.3 Configure Block Storage

- Install the **LVM Storage Operator** from the OpenShift hub, or storage can be provisioned through your storage operator.
- Create an LVM storage cluster instance.

Step 4.4 Configure Networking

- Install the **MetalLB Operator** or you can provision network operator specific to your environment to expose the service.
- Reserve an IP address for use by the load balancer provider configured above for external access.
- Ensure that the Cohesity cluster can reach this IP address.
- Ensure that all Cohesity clusters connected to Helios Self-Managed that will interact with the Gaia AI Engine can reach this IP address.
- Create a **DNS A/AAAA records** that points to this IP address.
- When you register the Cohesity cluster in Helios Self-Managed, it operates within a designated subnet. If the Gaia AI Engine is deployed on a Kubernetes cluster located in a different subnet, you must ensure that there is network connectivity between the Cohesity cluster and the Kubernetes cluster hosting Gaia. This connectivity is essential to enable seamless communication between the clusters.
- Ensure that there is network connectivity between the Cohesity cluster and the Gaia AI Engine, and between the Gaia AI Engine and the S3 endpoint.
- Ensure that your network firewall allows outbound traffic from the Cohesity cluster to the Kubernetes cluster where Gaia AI Engine is deployed.

- Ensure that the DNS name assigned to the Gaia AI Engine resolves to a reachable IP address from all required networks and the associated port is allowed in all relevant firewalls.

  NOTE: Detailed sample configurations for MetalLB, including IP Address Pool and L2 Advertisement manifests, are provided in Appendix-II for your reference.

Step 4.5 Set Up Container Registry

- Choose a Docker container registry (For example: Quay/Harbor or any Self-hosted registry), for Gaia AI Engine to push and pull the images for installation.
- Ensure that the Kubernetes cluster where the Gaia AI Engine is deployed can reach the container registry, and the repository used has permissions to push images and pull images.
- Ensure that the container registry has enough space to accommodate all the required images for the Gaia AI Engine

Step 4.6 Configure S3-Compatible Storage

When indexing a dataset, Cohesity Gaia generates embeddings and text. In a Self-Managed deployment, you must configure S3-compatible storage to store this data.

To set up the storage, gather the following details:

- S3 service hostname and port number
- Bucket name
- Access key ID
- Secret access key
- Determine and download the CA certificate for connecting to this S3 endpoint.

  You can use the following command to identify the certificate used by the S3 service:

  ```
  openssl s_client -showcerts -connect <hostname>:<port>
  ```

  Replace <*hostname*> and <*port*> with the appropriate values for your S3 service.

  From the output, copy the block(s) starting with -----BEGIN CERTIFICATE----- and ending with -----END CERTIFICATE----- into your configuration file, saving with the extension .pem

  Example:

  -----BEGIN CERTIFICATE----
  MIIDdTCCAl2gAwIBAgIUL6hN/mRkZ7Xp5M1W2z8z4F5G7cwwDQYJKoZIhvcNAQEL
  kK9p5m2G9J9f8vR8L3fXyL7D4qP5R8t/kK9p5m2G9J9f8vR8L3fXyL7D4qP5R8t/
  kK9p5m2G9J9f8vR8L3fXyL7D4qP5R8t/kK9p5m2G9J9f8vR8L3fXyL7D4qP5R8t/

  -----END CERTIFICATE-----

  File: cohesity-s3-ssl.pem

- Verify S3 connectivity from the Gaia AI Engine to the S3 endpoint.

Cohesity recommends creating an S3 View using Cohesity SmartFiles to store the embeddings and text. When creating the S3 View, select **Backup Target SSD** as the QoS policy and enable the **Pin Views to SSD** option. For more about these options, see Performance.

For more information about creating an S3 View, see Create an S3 View.

Step 4.7 Install Gaia Service from Helios Self-Managed

Helios Self-Managed provides the **Gaia** service. Until Gaia is installed, you cannot access it. If you try to launch Gaia from **Insights > Gaia**, you will be redirected to an **Upgrade Now** page.

To install the Gaia Service, refer to the Manage Services from the Helios Self-Managed Guide.

Step 4.8 Install and Register the Gaia AI Engine

To install and configure the Gaia AI Engine,

1. Download the Gaia AI Engine tar file from Cohesity Downloads on the machine used as a Jump host where the partition has enough space. Ensure that machine has both internet access and connectivity to the OpenShift cluster. For more information, see Jump Host.
2. Verify if the Software dependencies are installed on the Jump Host.
3. Perform the login to the openshift cluster from the machine, this is needed as the installer uses this session to perform the installation actions.

   ```
   oc login https://api.<>:6443/ -u kubeadmin -p <>
   ```

4. Extract the tar archive and navigate to the gaia-ai-engine directory. This directory contains the necessary installation scripts and configuration files required to deploy the Gaia AI Engine.

   ```
   tar -xvf <>.tar.gz
   ```

5. To begin the installation, use one of the following commands:
   - Single-node installation:

     ```
     ./gaia-ai-engine.sh --install --deployment-mode singlenode
     ```

   - Cluster installation (default option):

     ```
     ./gaia-ai-engine.sh --install --deployment-mode cluster
     ```

     ```
     To install with additional parameters specified:

     ./gaia-ai-engine.sh --install --pull-secret regcred --docker-registry
     <> --s3-host <> --s3-port <> --s3-access-key-id <> --s3-secret-key <>
     --s3-bucket-name <> --config-file-path <> --deployment-mode <>
     ```

6. Confirm acceptance of the End User License Agreement (EULA) when prompted.
7. Acknowledge that all prerequisite steps have been completed.

   ```
   2025-09-28 14:17:17 👉 NOTE: Make sure you have done the following:
   2025-09-28 14:17:17 👉 NOTE:  * Installed dependencies (kubectl, helm, docker, jq, yq, curl, oc) with correct versions.
   2025-09-28 14:17:17 👉 NOTE:  * Configured the kube config to point to the cluster where you want to install Gaia AI Engine.
   2025-09-28 14:17:17 👉 NOTE:  * Logged into a container registry that will host the images.
   2025-09-28 14:17:17 👉 NOTE:  * Have image pull secret credentials for the container registry.
   2025-09-28 14:17:17 👉 NOTE:  * Placed the gaia-ai-engine-config.json in the current directory.
   2025-09-28 14:17:17 👉 NOTE:  * Obtained an S3 host, port and credentials.
   ```

```
2025-09-28 14:17:17 👉 NOTE:  * Decided on a storage class name to be used for Gaia AI Engine service.
2025-09-28 14:17:17 👉 NOTE:  * Installed the Nvidia driver on the cluster.
2025-09-28 14:17:17 👉 NOTE:  * Installed the Nvidia operator for the GPU to become allocable in the cluster.
2025-09-28 14:17:17 👉 NOTE:  * Decided on the number of GPU nodes to be used for Gaia AI Engine service.
2025-09-28 14:17:17 ❓ CONFIRMATION: Make sure you have done all the above. Do you want to proceed? (y/n): y
```

8. The Installer checks for all the dependencies, and alerts if some dependencies are not met. Install the dependency and start the process again.

```
2025-09-30 01:11:56  🚦 Performing Pre-flight Checks...
2025-09-30 01:11:56 ℹ INFO: Checking for required command line tools...
2025-09-30 01:11:56 ✓ SUCCESS: Found command: kubectl
2025-09-30 01:11:56 ✓ SUCCESS: Found command: helm
2025-09-30 01:11:56 ✓ SUCCESS: Found command: docker
2025-09-30 01:11:56 ✓ SUCCESS: Found command: jq
2025-09-30 01:11:56 ✓ SUCCESS: Found command: yq
2025-09-30 01:11:56 ✓ SUCCESS: Found command: curl
2025-09-30 01:11:56 ✓ SUCCESS: All required commands are installed.
2025-09-30 01:11:56 ℹ INFO: Checking versions of required tools
2025-09-30 01:11:56 ℹ INFO: kubectl version: v1.28.0
2025-09-30 01:11:56 ✓ SUCCESS: kubectl version v1.28.0 is sufficient.
2025-09-30 01:11:56 ℹ INFO: helm version: v3.8.0+gd141386
2025-09-30 01:11:56 ✓ SUCCESS: helm version v3.8.0+gd141386 is sufficient.
2025-09-30 01:11:56 ℹ INFO: docker version: 28.1.1
2025-09-30 01:11:56 ✓ SUCCESS: docker version 28.1.1 is noted.
2025-09-30 01:11:57 ℹ INFO: Detected OpenShift cluster.
2025-09-30 01:11:57 ✓ SUCCESS: Found oc in path
2025-09-30 01:11:57 ✓ SUCCESS: oc version 4.18.0 is sufficient.
```

9. Press ‘y’ to confirm if the Kubernetes cluster is Red Hat OpenShift cluster. The Installer will fail if the below condition is not met:
   - Kubectl unable to reach any Kubernetes cluster. Check your kubeconfig to ensure it points to the correct valid cluster.

     To login to the Kubernetes cluster, see step 3 above in section 4.7. Once successfully logged in, ensure you confirm the cluster context.

```
2025-09-28 14:40:40 ℹ INFO: checking kubectl connection with cluster
2025-09-28 14:40:40 ✓ SUCCESS: kubectl can reach the Kubernetes cluster. Using context: 'myproject/api-hpe-********-com:6443/kube:admin'.
2025-09-28 14:40:40 ❓ CONFIRMATION: Do you wish to continue with cluster context: myproject/api-********-com:6443/kube:admin ? (y/n): y
2025-09-28 14:41:34 ℹ INFO: Kubernetes cluster version: v1.32.8
2025-09-28 14:41:34 ✓ SUCCESS: Kubernetes cluster version v1.32.8 is sufficient.
```

10. Type ‘n’ for Configuring Self-Hosted LLM endpoint as this is not yet supported.

```
2025-09-30 01:12:00 ❓ CONFIRMATION: Do you want to configure self-hosted LLM endpoint? (y/n): n
2025-09-30 01:12:05 ℹ INFO: Skipping BYOLLM configuration.
```

11. The Installer checks for the GPUs installed. **If there are no GPUs installed, or detected, please type ‘n’**, and proceed to verify if the GPU Support is setup correctly as specified in Step 4.3. GPUs are essential for Gaia AI Engine.

```
2025-09-30 01:12:05 ℹ INFO: Checking for GPU resources (nvidia.com/gpu).
2025-09-30 01:12:06 ⚠ WARNING: Found 0 GPUs. Need at least 2 GPUs (LLM+encoder).
2025-09-30 1:12:06 ❓ CONFIRMATION: Do you want to continue? (y/n):
```

12. Enter the absolute path to the Gaia AI Engine JSON configuration file that you previously copied to the machine (Gaia AI Engine Config can be downloaded from Helios Self-Managed).

    If the file path is incorrect, the script will notify you and you must restart the installation.

```
2025-09-30 01:15:56 🔑 Enter path of the json config obtained from the Registration wizard
Please provide the file path (e.g. absolute/path/to/gaia-ai-engine.json): /home/cohesity/Downloads/gaia-ai-engine/gaia-ai-engine.json
2025-09-26 02:04:34 ✓ SUCCESS: Configuration file found at: /home/cohesity/Downloads/gaia-ai-engine/gaia-ai-engine.json
2025-09-26 02:04:34 ✓ SUCCESS: Reading config from /home/cohesity/Downloads/gaia-ai-engine/gaia-ai-engine.json
```

13. Enter the S3 endpoint hostname, port, S3 access key, S3 secret key and bucket name. Enable SSL and provide the path to the certificate file you copied earlier, in Step 4.6

```
2025-09-26 02:04:35 📁 Configure S3 Endpoint and Credentials
Please provide an S3 hostname (e.g. my.s3.endpoint.com): sac-pm.cohesity.com
2025-09-26 02:04:44 ✓ SUCCESS: Got S3 hostname: sac-pm.cohesity.com
Please provide the port for S3 endpoint (e.g. 443): 3000
2025-09-26 02:04:48 ✓ SUCCESS: Got S3 endpoint port: 3000
Please provide the S3 access key ID (e.g. my-access-key-id): iTqiDIjv984c-yr47wHb2JaXyBhB_pFPAM
Please provide the S3 secret access key (e.g. my-access-secret-key): ICGLPRT15- gChsBpQGLJ2y4
Please provide the S3 bucket name (e.g. my-s3-bucket): s3archive
2025-09-26 02:05:14 ✓ SUCCESS: Got S3 bucket name: s3archive
2025-09-26 02:05:14 ❓ CONFIRMATION: Do you want to use SSL for S3 connection? (y/n): y
Please provide the path to s3 endpoint ca certificate (e.g. absolute/path/to/my/s3/ca.pem): /home/cohesity/Downloads/gaia-ai-engine/haswel.pem
2025-09-26 02:05:52 ✓ SUCCESS: Certificate file found at: /home/cohesity/Downloads/gaia-ai-engine/haswel.pem
2025-09-26 02:05:53 ℹ INFO: Testing basic connectivity to S3 endpoint: https:// sac-pm.cohesity.com:3000
2025-09-26 02:05:53 ✓ SUCCESS: Basic connectivity check to sac-pm.cohesity.com:3000 successful (HTTP 000000)
```

14. Choose the storage provider class to provision persistent volumes in the Kubernetes cluster where Gaia AI Engine is deployed, as specified in Step 4.3.

```
2025-09-30 01:27:37 🔎 Choose a storage class for persistent volumes:
1. lvm
Enter your choice (1-1): 1
2025-09-30 01:27:41 ℹ INFO: Selected storage class: lvm
2025-09-30  1:27:41 ℹ INFO: Selected service type: LoadBalancer
```

15. Enter the container registry address to be used for image storage.

    If your registry requires authentication, create an image pull secret by providing the necessary credentials.

    If no credentials are required, you may leave the username and password fields empty.

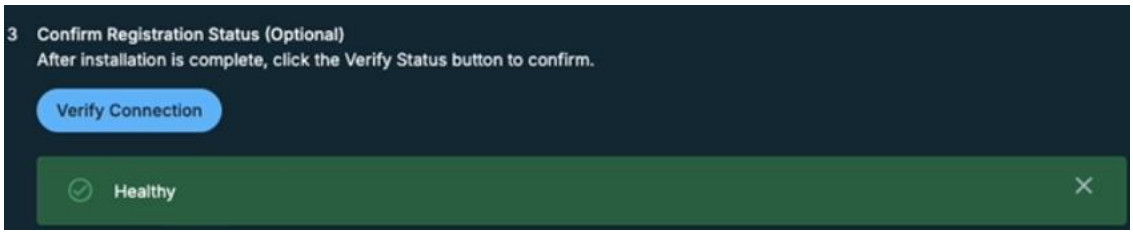
    Specify image pull secret name as: regcred

```
Enter container registry for hosting images (e.g., my.docker.io): docker.io
2025-09-30 01:30:43 ℹ INFO: Using Docker registry: docker.io
2025-09-30 01:30:43 ⚙️ Image Pull Secret for Container Registry
2025-09-30 01:30:43 ❓ CONFIRMATION: Does this registry require credentials? (y/n): y
2025-09-30 01:30:47 👉 NOTE: Share the name of image pull secret in the Kubernetes cluster. Please ensure that the secret exists in the namespace where you are installing the service.
Enter the image pull secret name for docker.io (e.g., my-registry-secret): regcred
2025-09-30 01:30:54 ℹ INFO: Creating image pull secret ‘regcred’ in namespace 'gaia-ai-engine'.
Enter username: jayd
Enter password:
secret/regcred created
2025-09-30 01:31:02 ✓ SUCCESS: Image pull secret 'regcred' created in namespace 'gaia-ai-engine'.
```

    The installation process begins. Note the following:

    - The script extracts the required images and pushes them to the specified container registry.
    - The script first deploys all required Custom Resource Definitions (CRDs) and dependencies, followed by the main application.
    - The Installer will push the images to the container registry, and uses it to deploy the necessary pods for Gaia AI Engine.
    - Once all the images are pushed, the Gaia AI Engine will start the installation and complete.

16. After the installation is complete, go back to the **Cohesity Helios UI**, and click **Verify Connection.** If the installation is successful, the status changes to **Healthy**.



You can view the connection status and Gaia AI Engine details in the **Settings > AI Engine Management** page.

If the installation is unsuccessful, the status changes to **Failed**. If the installation fails, contact Cohesity Support.

17. The next step is to create Protection Groups to Backup the Sources, to gain insights.

Step 4.9 Manage AI Engine

Click on the **Settings > AI Engine Management** page, to display the connection status of the Gaia AI Engine. From this page, you can also edit the AI Engine details or delete the AI Engine.

The following details are displayed:

- AI Engine name
- Connection status
- Last updated date and time

Step 5: Register Sources, Creation Protection Groups

With the Gaia AI Engine successfully installed, we proceed to register the sources, create protection policies and protection groups on the Cohesity Cluster.

For the list of Supported Sources and File Types, please refer the Product documentation.

We will take a look at Protecting NAS Source to gain insights.

- Register NAS Source
- Create a Protection Policy
- Create a Protection Group

For more up-to-date information on this refer to Gaia - Plan and Prepare page in the Helios Self-Managed documentation.

Step 6: Create Gaia Datasets

A dataset is a logical collection of data. Datasets play a fundamental role as a building block in the functioning of Cohesity Gaia. Datasets provide the necessary information for the system to learn and respond to questions. For example, a dataset in Cohesity Gaia could be the contents stored in the OneDrive of an individual or a specific department, such as Finance. It encompasses the organized and structured collection of data relevant to that person or department, and it could include various types of files, documents, or information that are logically grouped for easy access and management.

The initial setup of Cohesity Gaia involves the following steps:

- **Add data sources**—Choose the data sources that have been previously protected on Cohesity. To empower the Gaia - AI Assistant in assisting with user queries, Cohesity indexes the selected objects. Indexing involves organizing specific objects so that Cohesity Gaia can efficiently retrieve relevant information and provide accurate responses to user inquiries.
- **Choose authorized users**—Select and designate authorized users to ensure that the dataset remains accessible only to individuals with the appropriate permissions.

To configure Cohesity Gaia:

1. Log in to Helios Self-Managed.

   NOTE: Only a user with the Manage Gaia privilege can configure Cohesity Gaia. Administrative tasks can only be performed by individuals possessing this privilege.

2. Click the **Insights** pillar.
3. Click **Data Insights**.
4. Click **Create Dataset** and create the dataset.
5. Converse With Cohesity Gaia.

Please refer to Gaia - Plan and Prepare page in the Helios Self-Managed documentation, to Create Roles, and to Create Datasets.

Cohesity Gaia supports **continuous indexing**. This feature keeps your dataset updated to reflect the latest snapshot and enables you to interact with the most up-to-date data. *Please contact Cohesity support to enable this feature.*

In addition to conversing, Cohesity Gaia uses cutting-edge AI and natural language processing to create visual representations of data during indexing, offering unprecedented insights into complex datasets. The data exploration feature automatically analyzes text chunks within a dataset and generates a visually engaging word cloud. This functionality allows you to explore the dataset and uncover new insights.

Once the dataset is indexed, the dynamic word cloud represents the frequency of key terms and offers an intuitive way to identify the primary focus areas within the dataset. If a dataset is indexed using the **Continuous Indexing** option, topics are refreshed every seven days to reflect the latest data.

You can enable topic exploration to:

- Organize data into easily navigable themes
- Get an overview of what constitutes the dataset
- Get a list of relevant questions to ask

Before creating the dataset, you need to enable topic exploration. This allows Cohesity Gaia to analyze the dataset and generate a word cloud effectively.

To enable topic exploration, please refer to Visualize Your Datasets section.

Supported Languages

Cohesity Gaia now offers comprehensive multilingual support. You can now create datasets in the following languages:

- English
- Dutch

with additional languages planned for future releases. Please refer to product docs for more details.

You can index the data directly in its original language. Once the dataset is created, you can ask questions in any of these supported languages and Cohesity Gaia will respond in the same language, providing a seamless and natural interaction.

NOTE: While Cohesity Gaia handles your data and queries in multiple languages, the user interface (UI) options, error messages, and system prompts will remain in English.

Large Language Models

Large Language Models (LLMs) are advanced AI models designed to understand and generate human-like text on a large scale. LLMs are trained on massive amounts of data and can perform various tasks, such as text completion, summarization, and question answering.

Cohesity Gaia Self-Managed is integrated with enterprise grade LLM(s) and enables organizations to use AI responsibly and securely.

Converse With Cohesity Gaia

Once you have completed the initial setup and your datasets are created and indexed, you can select a dataset and start interacting with your backup data. Cohesity Gaia analyzes the data within your selected dataset and provides answers to your questions. You can ask follow-up questions and continue the conversation based on the responses.

To start a conversation with the **Gaia - AI Assistant**:

1. Log into **Cohesity Data Cloud**.
2. Click the **Insights** pillar.
3. Click **Data Insights**.
4. On the **Gaia - AI Assistant** page, choose the dataset and start interacting with your backup data.

Appendix I – Installing Nvidia GPU Operator

Installing the NVIDIA GPU Operator on OpenShift is a two-step process that requires the Node Feature Discovery (NFD) Operator to be installed first.

Step 1: Install the Node Feature Discovery (NFD) Operator

The GPU Operator relies on NFD to label your nodes (e.g., identifying which ones actually have a physical GPU).

1. In the OpenShift console, go to **Operators** > **OperatorHub**.
2. Search for "Node Feature Discovery".
3. Select the official operator by Red Hat and click **Install**.
4. Once installed, go to **Installed Operators** > **Node Feature Discovery**.
5. Click the **NodeFeatureDiscovery** tab and select **Create NodeFeatureDiscovery**.
6. Use the default name (nfd-master) and click **Create**.

Step 2: Install the NVIDIA GPU Operator

1. Navigate back to **Operators** > **OperatorHub**.
2. Search for "NVIDIA GPU Operator".
3. Click **Install**.
   - Namespace: It is highly recommended to use the suggested nvidia-gpu-operator namespace.
   - Update Channel: Select the channel that includes your required version (25.3.4).
4. Click **Install** and wait for the status to reach Succeeded.

Step 3: Create the ClusterPolicy

The Operator is just the "manager"; the ClusterPolicy is what actually deploys the drivers and software to your nodes.

1. Go to **Operators** > **Installed Operators**.
2. Click on the NVIDIA GPU Operator.
3. Select the ClusterPolicy tab and click **Create ClusterPolicy**.
4. You will see a large YAML/Form. For standard installation, the defaults are usually sufficient.

   TIP: If you are using a specific version (like 25.3.4), ensure the image tags in the driver section match your requirements.
5. Click **Create**.

Appendix II – Installing MetalLB

MetalLB provides load-balancer functionality for bare-metal Kubernetes/OpenShift clusters. It allows services of type LoadBalancer to get external IPs in environments without cloud load balancers. Below is a simple MetalLB instruction demonstration, please review the steps and configure according to the enterprise requirements.

Step 1: Install the MetalLB Operator

1. Log in to the OpenShift Web Console with cluster-admin privileges.
2. Navigate to **Operators** > **OperatorHub**.
3. Search for "MetalLB" in the filter box.
4. Select the MetalLB Operator (provided by Red Hat).
5. Click **Install**. On the installation page, keep the default settings (Namespace: openshift-operators or metallb-systemdepending on your OCP version).
6. Click Install again and wait for the status to show "Succeeded".

Step 2: Create the MetalLB Instance

Installing the Operator doesn't start MetalLB yet; you must create an instance of it.

1. Navigate to **Operators** > **Installed Operators**.
2. Click on the **MetalLB Operator**.
3. Select the **MetalLB** tab and click **Create MetalLB**.
4. Leave the default name as metallb and click **Create**.

Step 3: Configure Security Context Constraints (SCC) on OpenShift

MetalLB requires some pods to run with privileges for L2 networking.

```
oc adm policy add-scc-to-user privileged -z speaker -n metallb-system

oc adm policy add-scc-to-user privileged -z controller -n metallb-system
```

Step 4: Configure IPAddressPool and L2 Advertisement

MetalLB is now installed but idle. To start assigning IPs, you must define an **IPAddressPool** and an **L2Advertisement** (for Layer 2 mode).

```
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: gaia-lb-ip-pool
  namespace: metallb-system
spec:
  addresses:
    - 10.15.3.100-10.15.3.100
---
apiVersion: metallb.io/v1beta1
kind: L2Advertisement
metadata:
  name: gaia-lb-l2-advertisement
  namespace: metallb-system
spec:
  ipAddressPools:
    - gaia-lb-ip-pool
```

Apply the configuration.

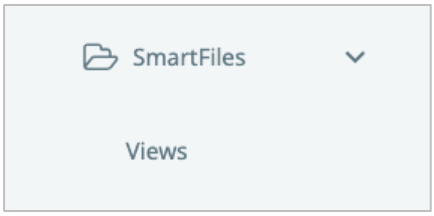
```
kubectl apply -f gaia-lb.yaml
```

Appendix III – Create Cohesity Smartfiles S3 View

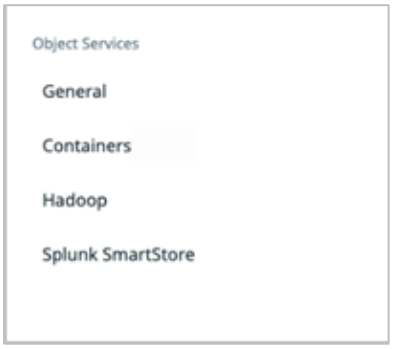
You can create a Cohesity Smartfiles S3 View, to store the embeddings and text generated by Cohesity dataset creation.

To create a S3 View:

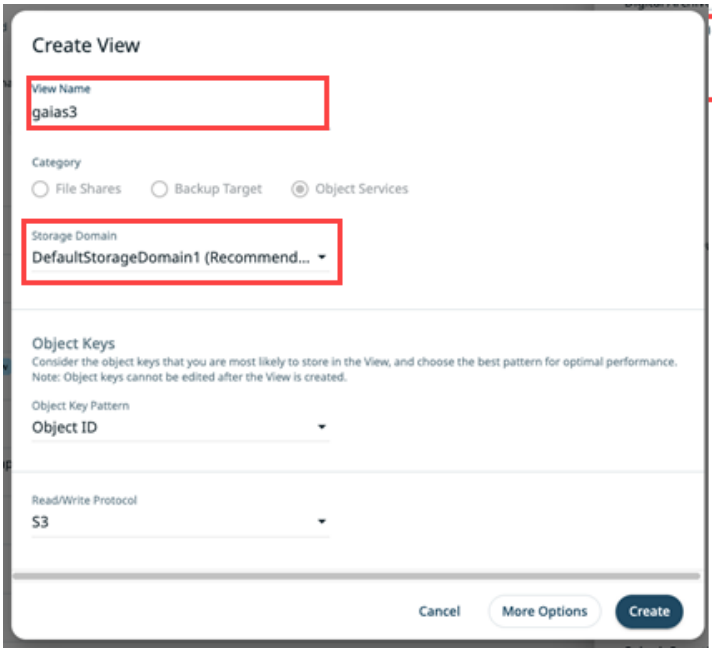
1. Login to Cohesity Cluster UI.
3. Navigate to **SmartFiles > Views** and do one of the following:



4. On the **Views** page, click **Create View**. On the **View Templates** side sheet, navigate to the Predefined tab and select **General** from Object Services.



5. Enter a name for the view and select the Storage Domain. Ensure that the Storage Domain has **Encryption Enabled.**



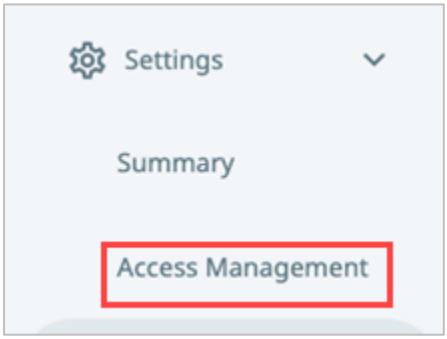
For more information about the available options, see About Cohesity View Options.

6. Add Subnets to Allowlist for SMB/NFS.
7. Click **Create** to create the view.

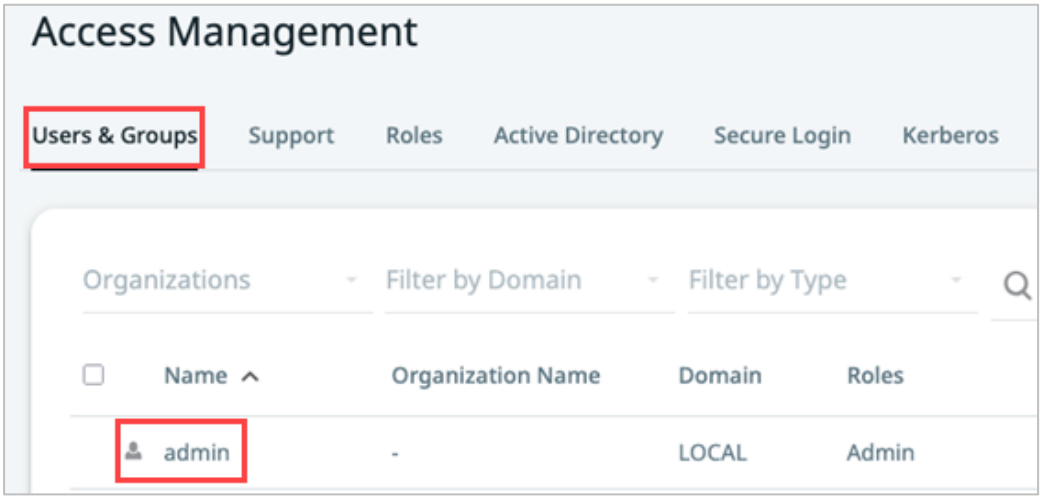
   Cohesity Smartfiles needs TCP port 3000 to enable S3 protocol access to Views hosted on the Cohesity cluster.

Access Key and Secret Key

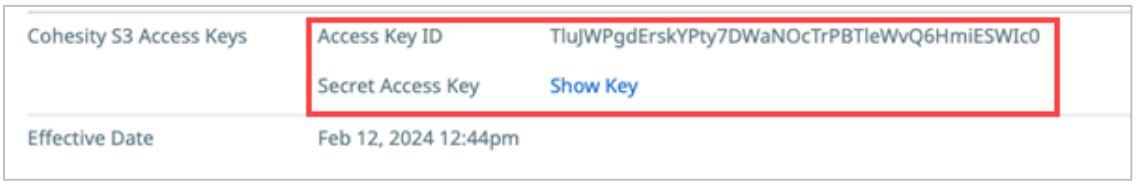
1. Navigate to **Settings** > **Access Management**.



2. Click on Users & Groups.
3. Select the user who is the bucket owner and click on the user.



4. Copy the Cohesity S3 Access Keys.



Your Feedback

Was this document helpful? Send us your feedback!

Authors

Jedidiah Sonavane, AI Solutions Architect, AI COE

He focuses on AI solutions here at Cohesity. He works on proofs of concept, enterprise data protection, insights, solution validation, solution design, testing, qualification, and ensuring software usability. He collaborates closely with teams to tailor solutions that meet customer needs while adhering to industry standards and best practices.

Other essential contributors include:

- Akshay Kumar, Director, Data Protect & AI COE
- Ashok Kumar Alluri, Senior Staff Software Engineer, Engineering

Document Version History

Version	Date	History
1.1	Feb 2026	Content Updates
1.0	Oct 2025	Original document

About Cohesity

Cohesity is a leader in AI-powered data security and management. Aided by an extensive ecosystem of partners, Cohesity makes it easier to protect, manage, and get value from data – across the data center, edge, and cloud. Cohesity helps organizations defend against cybersecurity threats with comprehensive data security and management capabilities, including immutable backup snapshots, AI-based threat detection, monitoring for malicious behavior, and rapid recovery at scale. Cohesity solutions are delivered as a service, self-managed, or provided by a Cohesity-powered partner. Cohesity is headquartered in San Jose, CA, and is trusted by the world's largest enterprises, including six of the Fortune 10 and 42 of the Fortune 100.

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

.