

# IBM Storage Ceph Solutions Guide

Vasfi Gucer

Kyle Bader

Reginald D'Souza

Jussi Lehtinen

Jean-Charles (JC) Lopez

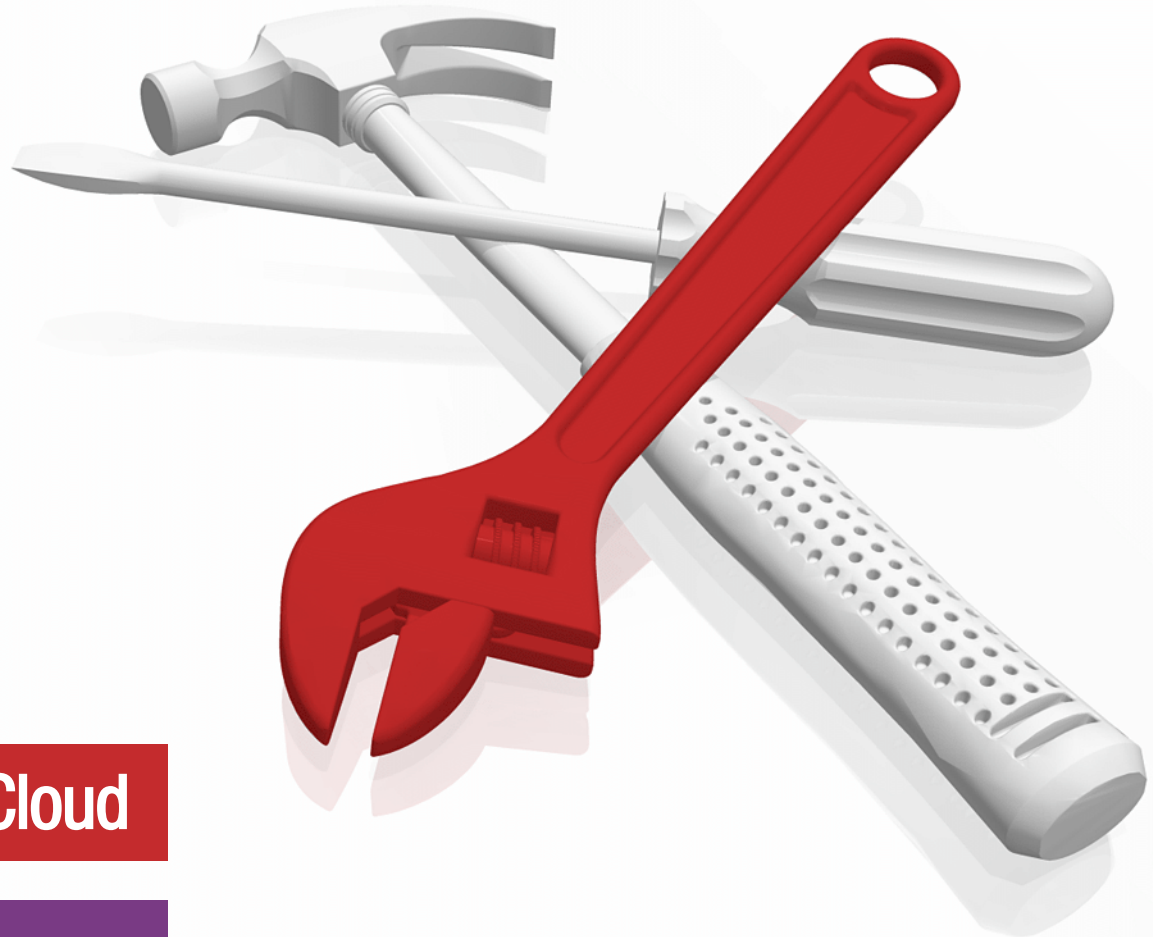
Franck Malterre

Suha Ondokuzmayis

Daniel Parkes

Poyraz Sagtekin

Tan Long Siau



 **Hybrid Cloud**

**Storage**





IBM Redbooks

**IBM Storage Ceph Solutions Guide**

May 2024

**Note:** Before using this information and the product it supports, read the information in “Notices” on page vii.

**First Edition (May 2024)**

This edition applies to IBM Storage Ceph Version 6.

**© Copyright International Business Machines Corporation 2024. All rights reserved.**

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

# Contents

<b>Notices</b> .....	vii
Trademarks .....	viii
<b>Preface</b> .....	ix
Authors .....	ix
Now you can become a published author, too! .....	xi
Comments welcome .....	xii
Stay connected to IBM Redbooks .....	xii
<b>Chapter 1. IBM Storage Ceph Dashboard</b> .....	1
1.1 Introduction .....	2
1.2 Connecting to the cluster Dashboard .....	2
1.3 Expanding the cluster .....	3
1.3.1 Adding hosts .....	3
1.3.2 Creating OSDs .....	5
1.3.3 Creating services .....	7
1.3.4 Expanding the cluster .....	8
1.3.5 Checking the configuration .....	8
1.4 Reducing the number of monitors to three .....	9
1.4.1 Configuring the web browser to access the Grafana performance tabs .....	10
1.5 Configuring RADOS Gateway .....	13
1.5.1 Creating rgw services and pools .....	13
1.5.2 Creating ingress services .....	14
1.5.3 Creating the data bucket pool by using erasure coding .....	15
1.5.4 Creating the user and its key .....	18
1.5.5 Putting data in to your bucket by using the AWS CLI .....	19
1.5.6 Checking the pool status .....	20
1.6 Creating a RADOS Block Device .....	21
1.6.1 Creating an RBD pool .....	21
1.6.2 Creating an image .....	22
1.6.3 Mapping the created disk .....	23
1.6.4 Using snapshot .....	24
1.7 Creating a Ceph File System .....	27
1.8 Monitoring .....	32
1.8.1 Main Dashboard .....	32
1.8.2 Cluster menu .....	32
1.8.3 Other Grafana performance data .....	35
<b>Chapter 2. IBM Storage Ceph for IBM watsonx.data</b> .....	37
2.1 Using IBM Storage Ceph with IBM watsonx.data .....	38
2.2 Creating users .....	38
2.2.1 Creating a user by using the Ceph Dashboard .....	38
2.2.2 Creating a user by using the CLI .....	42
2.3 Creating buckets .....	42
2.3.1 Creating a bucket with the Ceph Dashboard .....	42
2.3.2 Creating a bucket by using the CLI .....	43
2.4 Adding a bucket in IBM watsonx.data .....	44
2.4.1 Associating a catalog .....	50
2.4.2 Schema organization in a bucket .....	51

2.4.3	Creating a schema	52
2.5	Performance optimizations	54
2.5.1	Table format	54
2.5.2	Columnar formats	54
2.5.3	Partitioning	54
2.5.4	Bucketing	54
2.5.5	Table statistics	55
2.6	Querying less structured data (S3-Select)	55
2.6.1	Bucket versioning	55
2.6.2	Bucket lifecycle	55
2.6.3	Bucket notifications	56
<b>Chapter 3. A real-life Object Storage as a Service case study</b>		<b>57</b>
3.1	Company profile	58
3.2	Empowering enterprise storage: Object Storage as a Service in private cloud environments with IBM Storage Ceph	58
3.2.1	Understanding object storage: A paradigm for organizing data	58
3.2.2	Business challenges	58
3.2.3	Benefits of the approach	59
3.2.4	Architectural model	59
3.3	Sample implementation	61
3.3.1	Creating a user	61
3.3.2	Updating the quota	62
3.3.3	Usage and performance reporting	63
3.3.4	Documentation	65
<b>Chapter 4. Disaster recovery and backup and archive: IBM Storage Ceph as an Amazon S3 backup target</b>		<b>67</b>
4.1	Introduction	68
4.1.1	Benefits of IBM Storage Ceph for Veritas NetBackup	68
4.2	Implementing Veritas NetBackup with IBM Storage Ceph	69
4.2.1	Certified and supported solution	70
4.2.2	IBM Storage Insights	70
4.3	IBM Storage Ceph multi-site replication with Veritas NetBackup	70
4.3.1	Configuring the Veritas NetBackup components	71
4.4	IBM Storage Ready Nodes for Ceph	76
4.4.1	IBM Storage Ceph Editions	77
<b>Chapter 5. Amazon S3 bucket notifications for event-driven architectures</b>		<b>79</b>
5.1	Introduction	80
5.2	Cloud-native data pipelines versus legacy data pipelines	80
5.2.1	Legacy data pipelines	80
5.2.2	Cloud-native data pipelines	81
5.3	S3 bucket notifications: Components and workflows	82
5.3.1	Bucket notification workflow	82
5.3.2	Bucket notification reliability	82
5.4	Event-driven data pipeline example with the S3 bucket notification feature	83
5.4.1	Automated chest X-ray analysis for pneumonia detection	83
5.5	Configuring event notifications for a bucket in IBM Storage Ceph	85
5.5.1	Prerequisites	85
5.5.2	Configuring RGW S3 event bucket notifications	86
5.5.3	Validating the bucket notification configuration	88
5.6	Configuring S3 bucket notifications in Red Hat OpenShift with S3 RGW object storage by Fusion Data Foundation	90

5.6.1 Prerequisites . . . . .	90
5.6.2 Configuring RGW S3 event bucket notifications with FDF. . . . .	91
5.6.3 Validating the bucket notification configuration . . . . .	94
<b>Chapter 6. IBM Storage Fusion disaster recovery . . . . .</b>	<b>97</b>
6.1 Introducing Fusion Data Foundation . . . . .	98
6.1.1 Introducing Fusion disaster recovery . . . . .	98
6.1.2 Regional-DR overview . . . . .	100
6.1.3 Metro-DR overview . . . . .	101
6.1.4 Red Hat OpenShift DR with a stretched cluster . . . . .	102
6.2 IBM Storage Ceph stretch mode. . . . .	104
6.2.1 Avoiding stretched cluster issues by using Ceph stretched mode . . . . .	105
6.2.2 Stretched mode architecture layout . . . . .	106
6.2.3 Deployment example . . . . .	109
6.2.4 Configuring Ceph in stretched mode. . . . .	111
6.3 Fusion Metro-DR for RPO Zero . . . . .	115
6.3.1 Architecture: External versus internal Fusion Data Foundation deployments. . .	115
6.3.2 Metro-DR solution components. . . . .	116
6.3.3 Metro-DR architecture and layout. . . . .	116
6.3.4 Metro-DR application failover and fallback workflow. . . . .	118
6.3.5 Metro-DR deployment example . . . . .	118
<b>Chapter 7. Amazon S3 enterprise user authentication and authorization. . . . .</b>	<b>119</b>
7.1 Introducing RADOS Gateway Auth methods . . . . .	120
7.1.1 Introducing the RGW Secure Token Service . . . . .	120
7.1.2 Introducing STS with an OIDC provider . . . . .	121
7.1.3 Introducing IAM role policies. . . . .	122
7.2 Step-by-step deployment guide: S3 enterprise user authentication/authorization . . .	124
7.2.1 Installing Red Hat SSO (Keycloak) on Red Hat OpenShift . . . . .	125
7.2.2 Installing Red Hat Identity Management Server . . . . .	128
7.2.3 Hostname and DNS requirements . . . . .	128
7.2.4 Port requirements . . . . .	128
7.2.5 Installing packages . . . . .	129
7.2.6 Installing IdM. . . . .	129
7.2.7 Configuring Red Hat Single Sign-On (Keycloak) . . . . .	134
7.2.8 Client configuration . . . . .	137
7.2.9 Configuring the Ceph Realm settings . . . . .	141
7.2.10 Verifying the Keycloak configuration: JWT token retrieval and validation. . . . .	141
7.3 IBM Storage Ceph STS configuration. . . . .	144
7.3.1 Registering an OpenID Connect provider on IBM Storage Ceph . . . . .	144
7.3.2 Configuring certificates . . . . .	146
7.3.3 Configuring RGW in highly available mode with load balancing and SSL . . . . .	148
7.3.4 Registering the OIDC provider with RGW. . . . .	150
7.3.5 Role and role policy creation. . . . .	151
7.3.6 Attaching policies to roles . . . . .	155
7.3.7 Enabling STS on the RGW service. . . . .	157
7.3.8 Importing a Keycloak certificate to an RGW node . . . . .	158
7.4 Testing Assume Role With Web Identity for role-based access control . . . . .	159
7.4.1 Testing the Assume role with the Admin role . . . . .	160
7.4.2 Testing the Assume role with the Writer role . . . . .	161
7.4.3 Testing the Assume role with the Reader role . . . . .	161
<b>Abbreviations and acronyms . . . . .</b>	<b>163</b>

<b>Related publications</b> .....	165
IBM Redbooks .....	165
Other publications .....	165
Help from IBM .....	165



# Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive, MD-NC119, Armonk, NY 10504-1785, US*

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

## COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

# Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at <https://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

IBM®	Redbooks®
IBM Cloud®	Redbooks (logo)  ®

The following terms are trademarks of other companies:

ITIL is a Registered Trade Mark of AXELOS Limited.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Ceph, JBoss, OpenShift, Red Hat, are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

VMware, and the VMware logo are registered trademarks or trademarks of VMware, Inc. or its subsidiaries in the United States and/or other jurisdictions.

Other company, product, or service names may be trademarks or service marks of others.

# Preface

IBM Storage Ceph is an IBM® supported distribution of the open-source Ceph platform that provides massively scalable object, block, and file storage in a single system.

IBM Storage Ceph is designed to enable AI with enterprise resiliency, consolidate data with software simplicity, and run on multiple hardware platforms to provide flexibility and lower costs.

Engineered to be self-healing and self-managing with no single point of failure, IBM Storage Ceph includes storage analytics for critical insights into growing amounts of data. IBM Storage Ceph can be used as a simple and efficient way to build a data lakehouse for IBM watsonx.data and next-generation AI workloads.

This IBM Redpaper publication explains the implementation details and use cases of IBM Storage Ceph. For more information about the IBM Storage Ceph architecture and technology that is behind the product, see *IBM Storage Ceph Concepts and Architecture Guide*, REDP-5721.

The target audience for this publication is IBM Storage Ceph architects, IT specialists, and storage administrators.

## Authors

This paper was produced by a team of specialists from around the world.



**Vasfi Gucer** works as the Storage Team Leader on the IBM Redbooks® Team. He has more than 30 years of experience in the areas of systems management, networking hardware, and software. He writes extensively and teaches IBM classes worldwide about IBM products. For the past decade, his primary focus has been on storage, cloud computing, and cloud storage technologies. Vasfi is also an IBM Certified Senior IT Specialist, Project Management Professional (PMP), IT Infrastructure Library (ITIL) V2 Manager, and ITIL V3 Expert.



**Kyle Bader** is a Senior Technical Staff Member and Principal Portfolio Architect for Ceph based offerings at IBM. He has spent a decade designing Ceph based storage clusters and solutions. His current interests lie at the intersection of distributed storage, containers, data-intensive applications, and machine learning.



**Reginald D'Souza** is an IBM Storage Technical Sales Specialist at IBM Australia. He has 20 years of experience as an IT infrastructure specialist managing and delivering projects that span different industry verticals. He is focused on software-defined storage and modern data protection solutions by assisting clients in their hybrid cloud journey. Reginald is also an IBM L2 Certified IT Specialist.



**Jussi Lehtinen** is a Principal Storage subject matter expert for IBM Object Storage who works for IBM Infrastructure in the Nordics, the Benelux, and Eastern Europe. He has over 35 years of experience working in IT, with the last 25 years with Storage. He holds a bachelor's degree in Management and Computer Studies from Webster University in Geneva, Switzerland.



**Jean-Charles (JC) Lopez** has been in storage for 80% of his professional career, and has been working in software-defined storage since 2013. JC helps people understand how they can move to a containerized environment when it comes to data persistence and protection. His go-to solutions are Fusion, Ceph, and Red Hat OpenShift. He takes part in the Upstream Community and downstream versions of Ceph and Red Hat OpenShift.



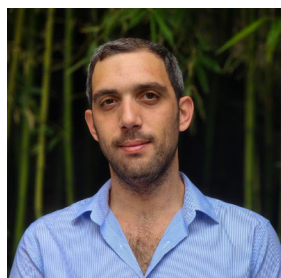
**Franck Malterre** is an information technology professional with a background of over 25 years in designing, implementing, and maintaining large x86 physical and virtualized environments. For the last 10 years, he has specialized in the IBM File and Object Storage Solution by developing IBM Cloud Object Storage, IBM Storage Scale, and IBM Storage Ceph live demonstrations and proofs of concept for IBM personnel and IBM Business Partners.



**Suha Ondokuzmayis** began his career in 2001 as an assembly language developer intern at Turkish Airlines, which was followed by working at Turkcell, Turkish Telekom, IBM, and Veritas Technologies within different departments, all with infrastructure-related responsibilities. He joined İşbank in 2013, where he has worked with IT infrastructures, IaaS and PaaS platforms, and block, file, and object storage product offerings, and backup operations. He holds a Master of Science degree in Management Information Systems from the University of Istanbul.



**Daniel Parkes** works in the IBM Storage Ceph Product Management team, where he focuses on the IBM Storage Ceph Object Storage offering.



**Poyraz Sagtekin** joined IBM in 2016 after completing his Computer Engineering degree at Middle East Technical University. Since then, he has worked in diverse areas, including IoT systems, IBM Cloud®, Kubernetes, Red Hat Enterprise Linux (RHEL), and IBM Power servers. Currently, he works as a Hybrid Cloud Services Consultant at IBM Systems Expert Labs in Türkiye. He is an IBM recognized speaker and educator.



**Tan Long Siau** works as a Software-Defined Storage Technical Specialist for IBM ASEANZK. He has over 8 years of experience in the storage industry. His main areas of interest include file and object storage, and container storage. He is a seasoned technologist with big data, AI, IoT, and high-performance computing infrastructure architectural experience.

Thanks to the following people for their contributions to this project:

Kenneth David Hartsoe, Elias Luna, Henry Vo, Wade Wallace, William West  
**IBM US**

Also, thanks to James Eckersall, James Pool, and Ian Watson from Red Hat, Inc. They started this journey by providing the initial insight and documentation for Chapter 7, “Amazon S3 enterprise user authentication and authorization” on page 119.

The team extends its gratitude to the Upstream Community, IBM, and Red Hat Ceph Documentation teams for their contributions to continuously improve Ceph documentation.

## Now you can become a published author, too!

Here’s an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an IBM Redbooks residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:  
[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

## Comments welcome

Your comments are important to us!

We want our papers to be as helpful as possible. Send us your comments about this paper or other IBM Redbooks publications in one of the following ways:

- ▶ Use the online **Contact us** review Redbooks form found at:

[ibm.com/redbooks](http://ibm.com/redbooks)

- ▶ Send your comments in an email to:

[redbooks@us.ibm.com](mailto:redbooks@us.ibm.com)

- ▶ Mail your comments to:

IBM Corporation, IBM Redbooks  
Dept. HYTD Mail Station P099  
2455 South Road  
Poughkeepsie, NY 12601-5400

## Stay connected to IBM Redbooks

- ▶ Find us on LinkedIn:

<https://www.linkedin.com/groups/2130806>

- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

<https://www.redbooks.ibm.com/subscribe>

- ▶ Stay current on recent Redbooks publications with RSS Feeds:

<https://www.redbooks.ibm.com/rss.html>



# IBM Storage Ceph Dashboard

IBM Storage Ceph comes with a GUI that is called *Dashboard*. This dashboard can simplify deployment, management, or monitoring. This chapter describes the dashboard features.

This chapter has the following sections:

- ▶ Introduction
- ▶ Connecting to the cluster Dashboard
- ▶ Expanding the cluster
- ▶ Reducing the number of monitors to three
- ▶ Configuring RADOS Gateway
- ▶ Creating a RADOS Block Device
- ▶ Creating an image
- ▶ Creating a Ceph File System
- ▶ Monitoring

## 1.1 Introduction

Before you use the IBM Ceph Dashboard, install some prerequisites that are documented in [IBM Storage Ceph documentation](#).

Start after you run the Ceph **bootstrap** command, which, among other configuration tasks, installs and starts a Ceph Monitor daemon and a Ceph Manager daemon for a new IBM Storage Ceph cluster on the local node as containers.

**Note:** You can also watch the following [video](#), which describes how to install IBM Storage Ceph by using the CLI.

The bootstrap process also provides the URL, login, and password to connect to the IBM Storage Ceph Dashboard.

Our sample cluster is made of four nodes, which is the minimum number that is supported by IBM, as shown in Figure 1-1.

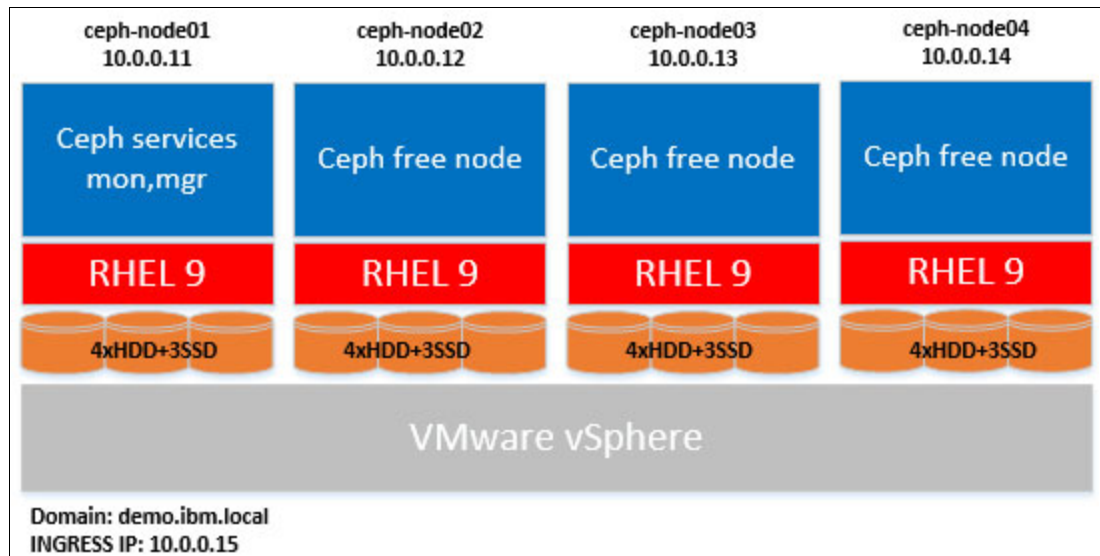


Figure 1-1 Cluster nodes

The bootstrap node is ceph-node01, and each node is connected to four HDDs and three SSDs disks that are used for the object storage daemon (OSD).

## 1.2 Connecting to the cluster Dashboard

By connecting to the URL and using the password that is provided at the end of the bootstrap process, you can log in as admin on the Dashboard after setting a new password. The Expand Cluster window opens, as shown in Figure 1-2 on page 3.



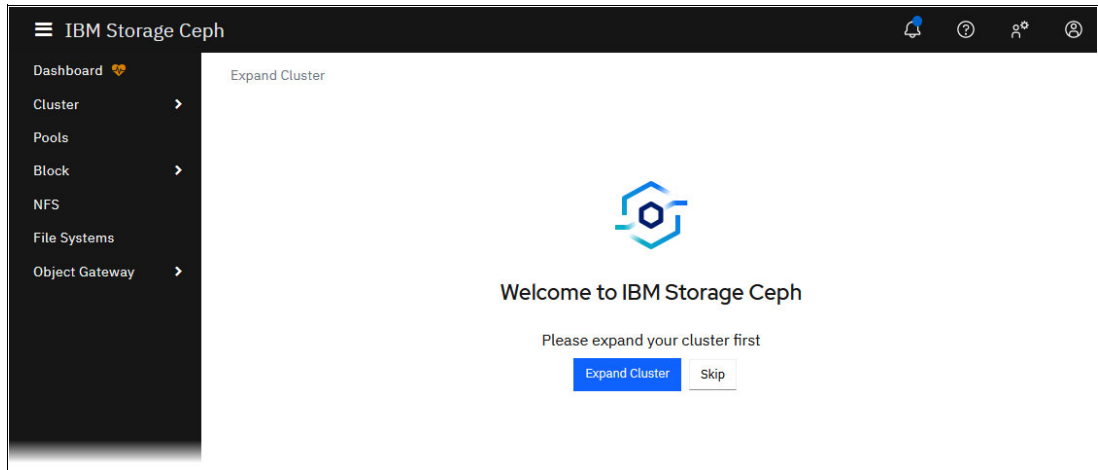


Figure 1-2 Dashboard landing page

## 1.3 Expanding the cluster

This section shows the steps to expand the cluster.

### 1.3.1 Adding hosts

To add hosts, complete the following steps:

1. To start the assistant, click **Expand Cluster**, as shown in Figure 1-3.

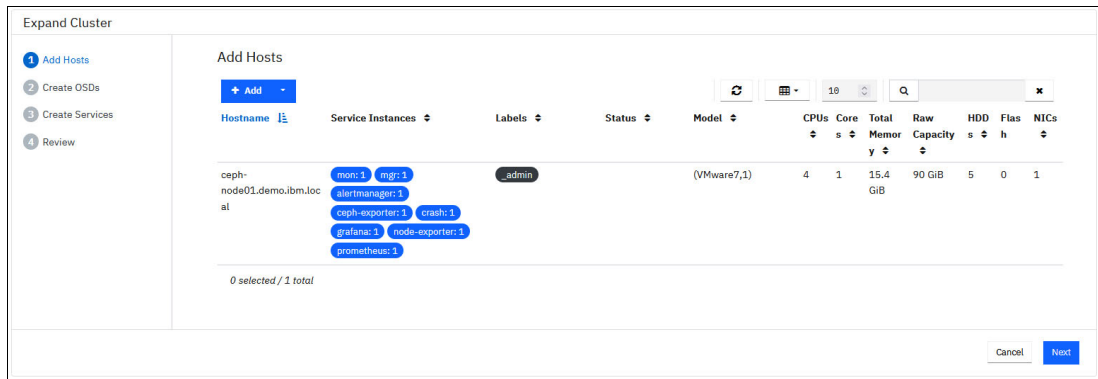


Figure 1-3 Adding hosts

2. From this window, you can either edit the configuration or add a host. The first host was installed by the bootstrap, and this host supports all the daemons of the cluster.

You can choose to add one host at a time, or add multiple hosts concurrently. In our configuration, hosts 02 - 04 receives the hostname `ceph-node[02-04].demo.ibm.local`. When adding hosts, it is convenient to select labels to apply. Labels are used to automatically deploy the corresponding daemon on the host.

Figure 1-4 shows the labels for this example.

Figure 1-4 Adding multiple hosts at once with labels

**Tip:** Clicking the question mark presents you with some examples.

Adding multiple hosts at once is faster, but you cannot specify different labels on each host. By adding one host at a time, you can select different labels, but this task is cumbersome on a large configuration. Here, every host runs the monitor and the osd daemon, so you can select these labels.

To spread the different workloads across all hosts, you want the following daemon configuration, as shown in Table 1-1.

Table 1-1 Daemon dispatch

Hostname	mgr	osd	rgw
ceph-node01.demo.ibm.local	X	X	
ceph-node02.demo.ibm.local	X	X	
ceph-node03.demo.ibm.local		X	X
ceph-node04.demo.ibm.local		X	X

- To add the mgr label, edit ceph-node01. To add the osd label, edit ceph-node02. To add the rgw label, edit ceph-node03 and ceph-node04. Move the grafana daemon from ceph-node01 to ceph-node04.

Figure 1-5 on page 5 shows the hosts and their labels.

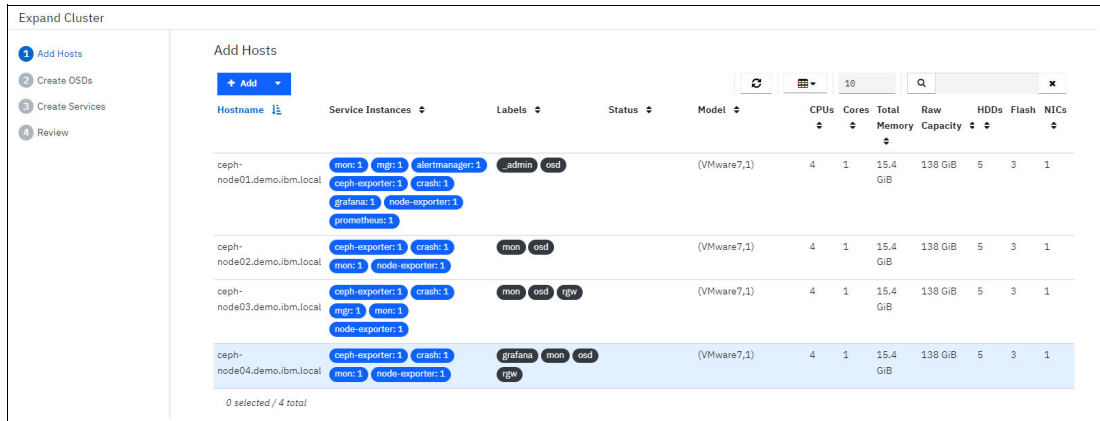


Figure 1-5 Hosts Labels

4. Deploying the daemons can take a while, but we can proceed to OSD creation by clicking **Next**.

### 1.3.2 Creating OSDs

With the Expand Cluster assistant, you can select a few, pre-defined options:

- ▶ Cost/capacity-optimized, which uses all available HDDs to store data.
- ▶ Throughput-optimized, which uses HDDs to store data and SSDs to store BlueStore write-ahead (WAL) devices and databases (DBs).
- ▶ Input/output operations per second (IOPS)-optimized, which uses nonvolatile memory express (NVMe) drives to store data.

In this example, you choose HDD drives as “Primary devices” and SSD drives as “DB” to support BlueStore and increase performance.

Complete the following steps:

1. Select the **Advanced** mode, as shown in Figure 1-6.

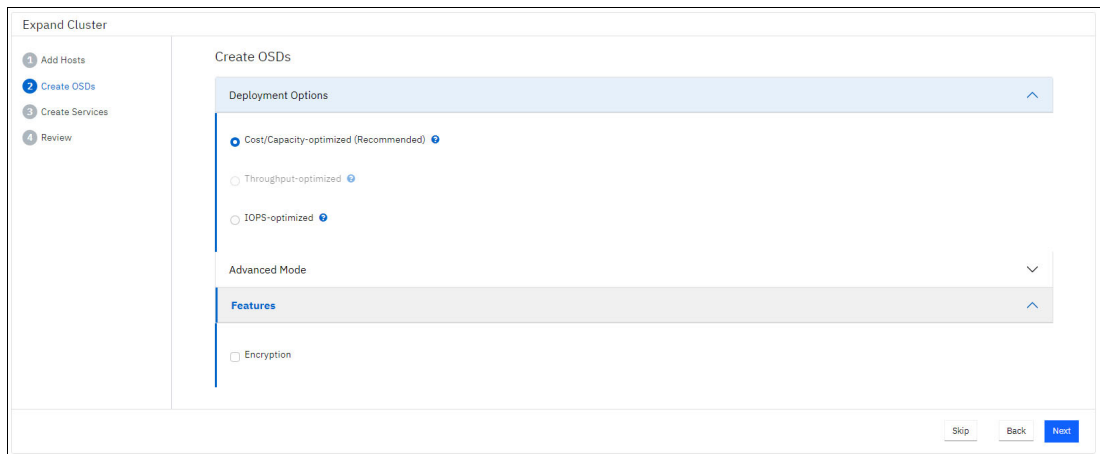


Figure 1-6 Create OSDs window

2. Click **Add Primary devices**. By using the filters at the upper right, display only the HDD type. After the HDD devices are selected, click **Add**, as shown in Figure 1-7.

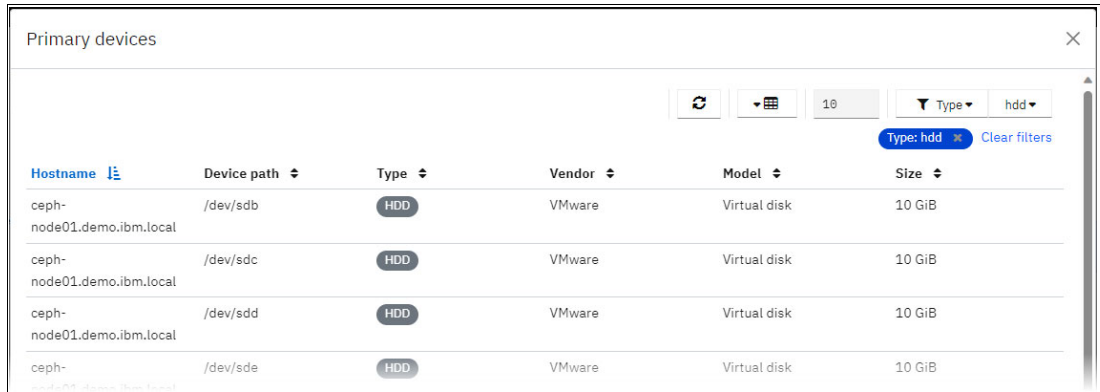


Figure 1-7 Adding Primary devices

3. When the primary devices are selected, you can add WAL devices and DB devices. In this example, you use SSD devices for DB devices only, as shown in Figure 1-8.

**Note:** Because the BlueStore journal is always placed on the fastest device, using a DB device provides the same benefit that the WAL device provides while also allowing for storing additional metadata, so no dedicated WAL is required.

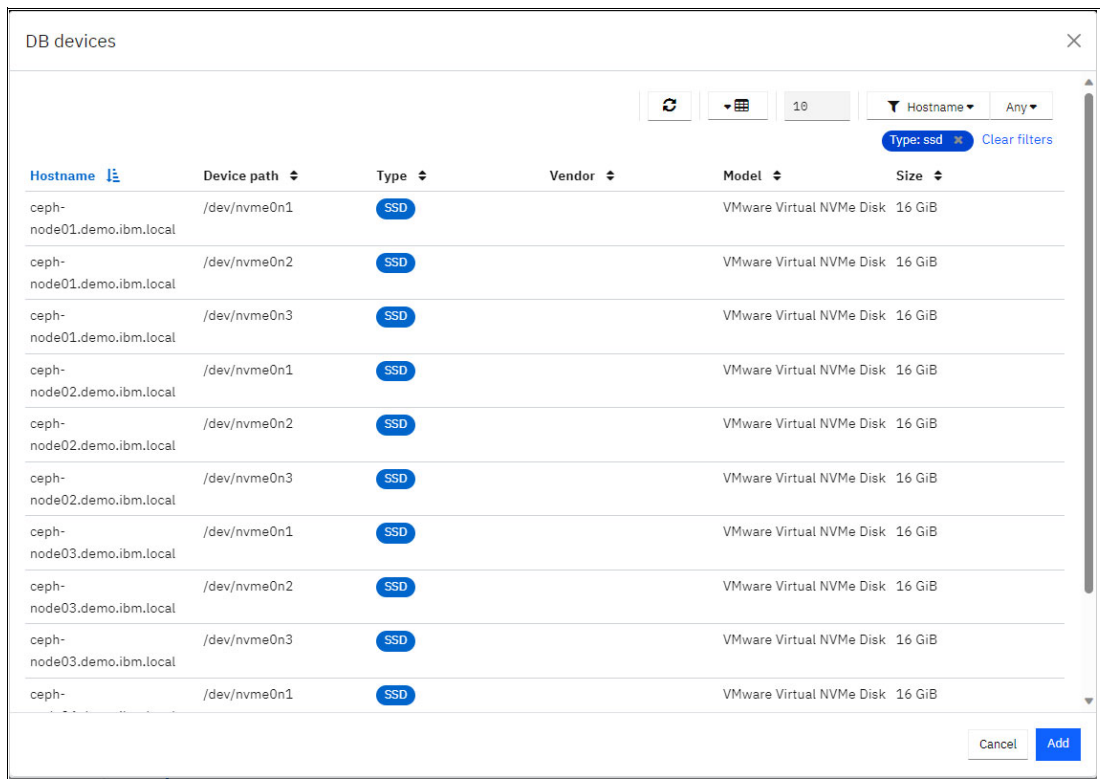


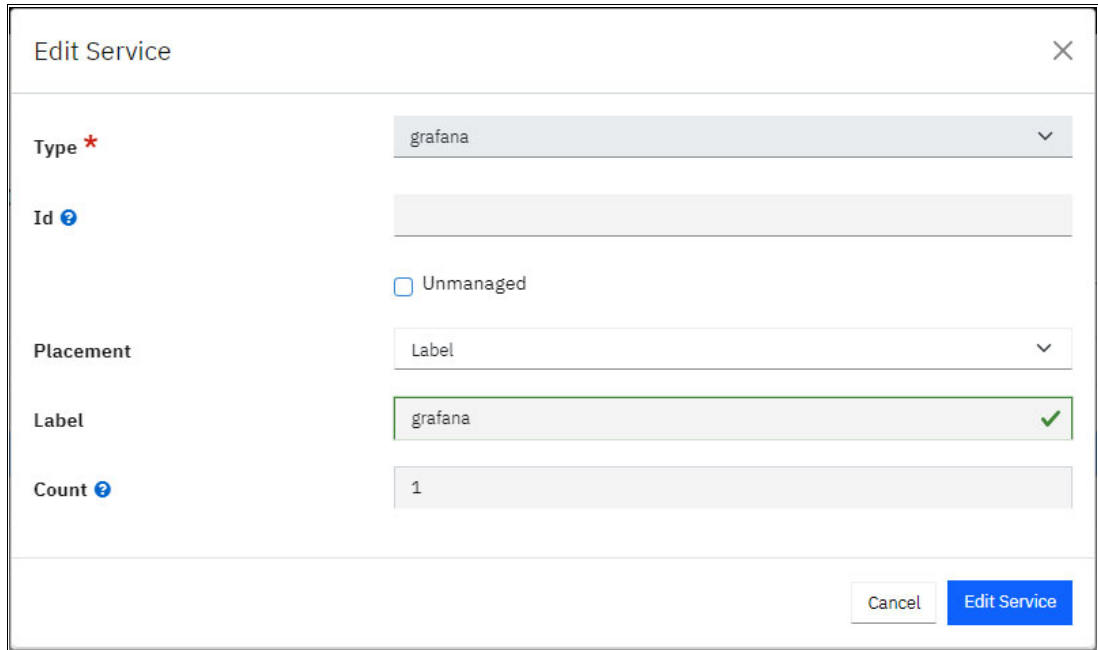
Figure 1-8 Adding DB devices

### 1.3.3 Creating services

This section describes how to create or edit services. As an example, you can edit the Grafana service to move it to ceph-node04 to free some resources on ceph-node01.

Complete the following steps:

1. Select **grafana** and select **Edit** in the drop-down list (Figure 1-9).



The screenshot shows a dialog box titled "Edit Service" with a close button (X) in the top right corner. The dialog contains several fields for configuring the service:

- Type \***: A dropdown menu with "grafana" selected.
- Id ?**: An empty text input field.
- Unmanaged**: An unchecked checkbox.
- Placement**: A dropdown menu with "Label" selected.
- Label**: A dropdown menu with "grafana" selected, indicated by a green checkmark.
- Count ?**: A text input field containing the number "1".

At the bottom right of the dialog, there are two buttons: "Cancel" and "Edit Service".

Figure 1-9 Label relocate for grafana

2. When you are done, review the configuration and start the configuration of the cluster.

### 1.3.4 Expanding the cluster

These steps start the deployment and configuration of daemons on the cluster hosts. To follow this process, by clicking the bell icon at the upper right of the window (Figure 1-10).

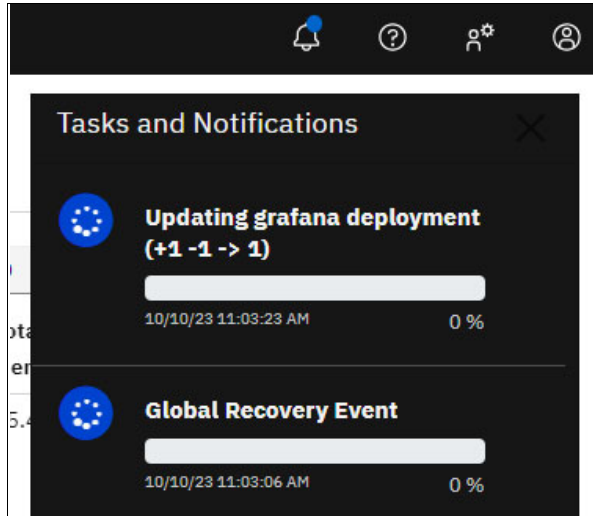


Figure 1-10 Following the deployment

### 1.3.5 Checking the configuration

Services deployment and OSDs creation take a few minutes. You can check their statuses by selecting **Cluster** → **Hosts**. The Hosts window should display the four nodes and their instanced services (Figure 1-11).

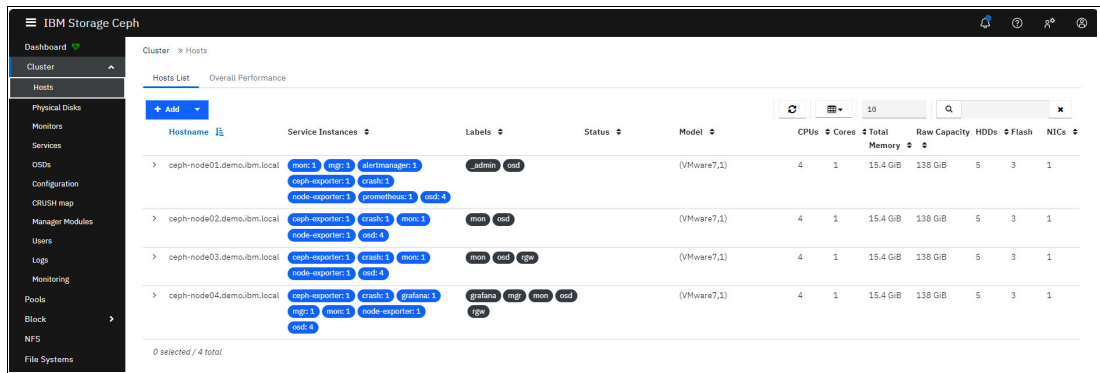


Figure 1-11 Checking the hosts services

Clicking **OSDs** shows the OSDs, which should be available for use (Figure 1-12 on page 9).

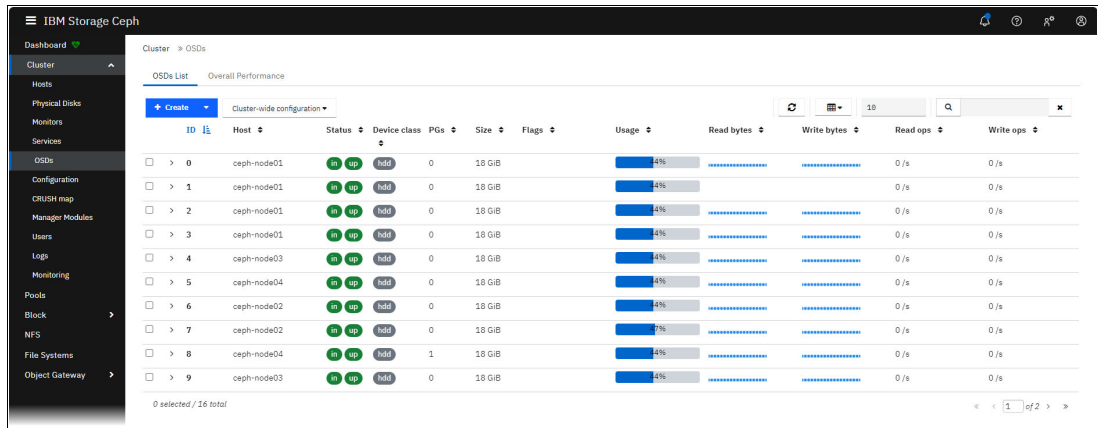


Figure 1-12 Checking the OSDs

## 1.4 Reducing the number of monitors to three

By default, a typical IBM Storage Ceph cluster has three or five monitor daemons that are deployed on different hosts. A best practice is to deploy five monitors if there are five or more nodes in the cluster. By design, the Dashboard is configured to deploy five monitors, but in our small cluster configuration of four nodes only, this configuration results in the deployment of one monitor per node. An odd number of monitors is a best practice for quorum, so we remove the monitor that is deployed on ceph-node04. This action removes some workload on ceph-node04, which is now the Grafana node.

Complete the following steps:

1. To change this setting, select **Cluster** → **Services**, click **mon**, and then click **Edit** (Figure 1-13).

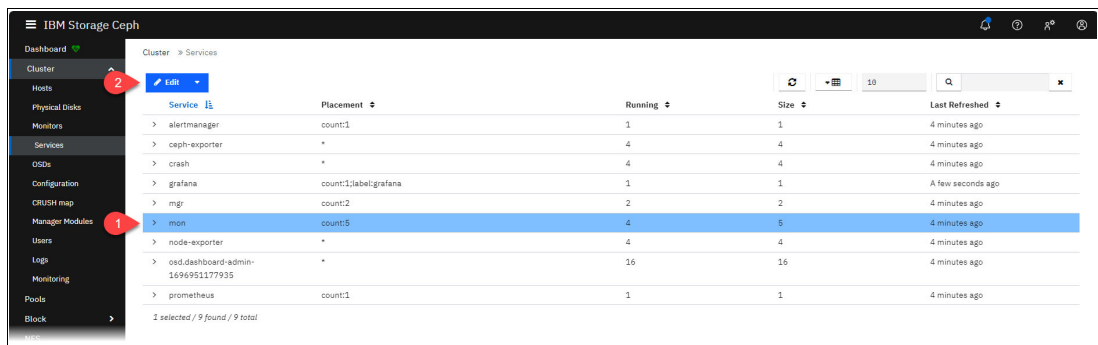


Figure 1-13 Editing the mon service

2. Edit the Hosts list to select the first three hosts and reduce the Count to 3 (Figure 1-14).

The screenshot shows the 'Edit Service' dialog box with the following configuration:

- Type \***: mon
- Id ?**: (empty)
- Unmanaged**:
- Placement**: Hosts
- Hosts**:
  - ceph-node01.demo.ibm.local
  - ceph-node02.demo.ibm.local
  - ceph-node03.demo.ibm.local
- Count ?**: 3

Buttons: Cancel, Edit Service

Figure 1-14 Selecting the mon hosts

3. If you check the Services window, you see the new placement of the mon service.

### 1.4.1 Configuring the web browser to access the Grafana performance tabs

As you move the Grafana service to `ceph-node04.demo.ibm.local`, accept its self-signed certificate to display the content of the Grafana tabs. In this example, we use Firefox, but the process should be similar on other browsers.

Complete the following steps:

1. Access the Firefox settings by clicking **Application Settings** (1), and then **Settings** (2), as shown in Figure 1-15 on page 11.



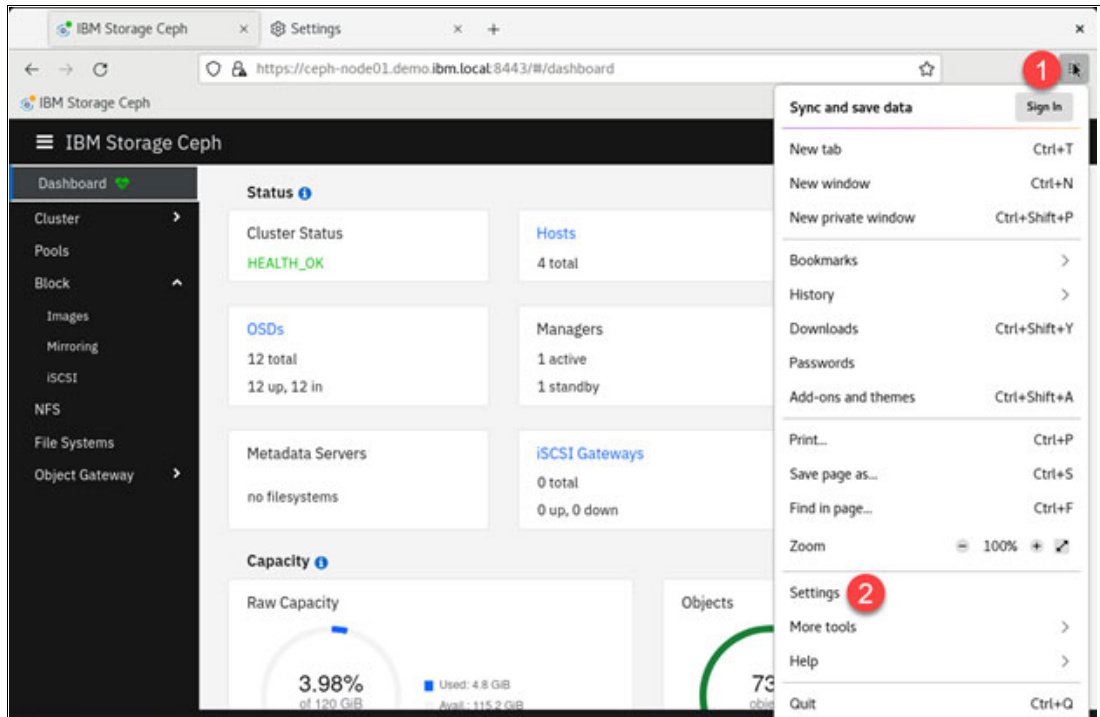


Figure 1-15 Browser certificates settings

2. Click **Privacy & Security** (1), scroll down to the Certificates pane, and then click **View Certificates...**(2) (Figure 1-16).

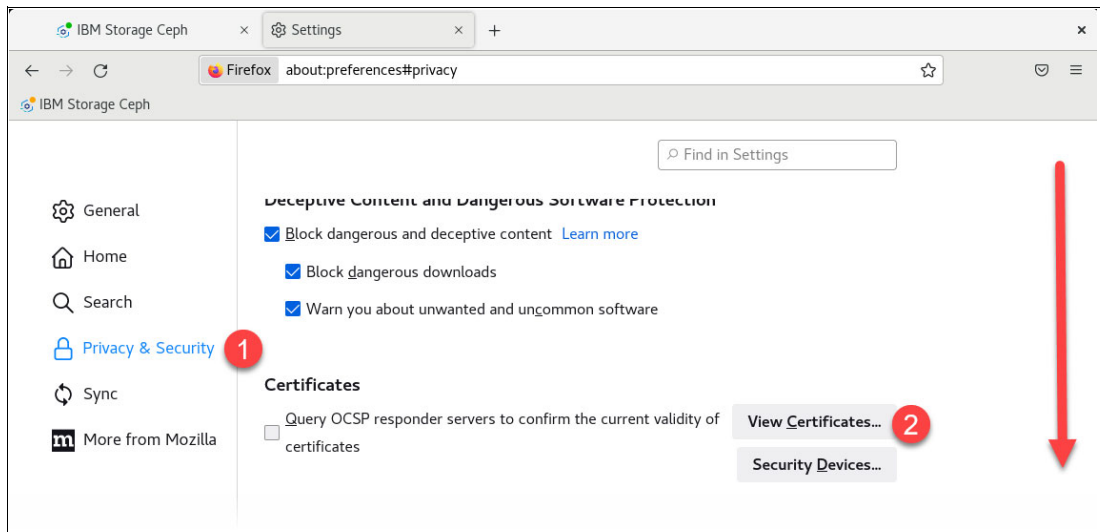


Figure 1-16 Viewing certificates

3. On the **Servers** tab of Certificate Manager, click **Add Exception...** (Figure 1-17).

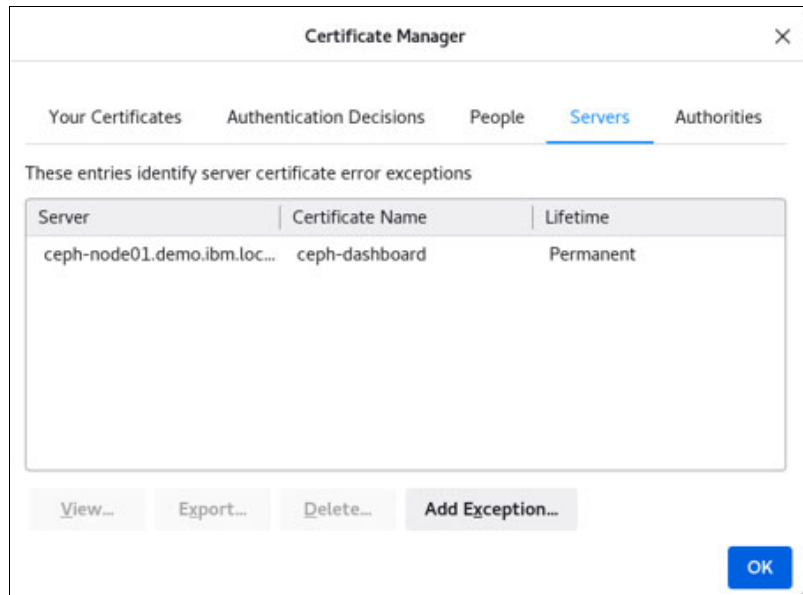


Figure 1-17 Certificates Servers tab

4. Add the ceph-node04 Grafana URL and click **Get Certificate**. Click **Confirm Security Exception** (Figure 1-18).



Figure 1-18 Server certificate URL

5. Click **OK** to validate the configuration and close the **Settings** tab.

## 1.5 Configuring RADOS Gateway

Ceph Object Storage uses the Ceph Object Gateway daemon (radosgw), which is an HTTP Server that is designed to interact with a Ceph Storage Cluster. The Ceph Object Gateway provides interfaces that are compatible with both Amazon S3 and OpenStack Swift, and it has its own user management.

The radosgw daemon is a client to Ceph Storage that provides object access to other client applications. Client applications use standard APIs to communicate with the RADOS Gateway (RGW), and the RGW uses librados module calls to communicate with the Ceph cluster.

The RGW is a separate service that externally connects to a Ceph cluster and provides object storage access to its clients. As a best practice in a production environment, run more than one instance of the RGW, which is masked by a load balancer.

### 1.5.1 Creating rgw services and pools

To create rgw services and pools, complete the following steps:

1. To create the rgw service, select **Cluster** → **Services**, and then click **Create**.

Complete the Create Service dialog box (Figure 1-19) and click **Create Service**. You previously added the rgw label to ceph-node03 and ceph-node04, so you can use this label to deploy the service on these hosts.

The screenshot shows a 'Create Service' dialog box with the following fields and options:

- Type \***: rgw (with a green checkmark icon)
- Id \***: default (with a green checkmark icon)
- Unmanaged
- Placement**: Label (with a green checkmark icon)
- Label**: rgw (with a green checkmark icon)
- Count**: (empty field with a help icon and a dropdown arrow)
- Port**: (empty field with a dropdown arrow)
- SSL

At the bottom of the dialog, there are two buttons: 'Cancel' and 'Create Service' (which is highlighted in blue).

Figure 1-19 Creating the rgw service

2. Verify the deployment of the service on the two selected hosts (Figure 1-20).

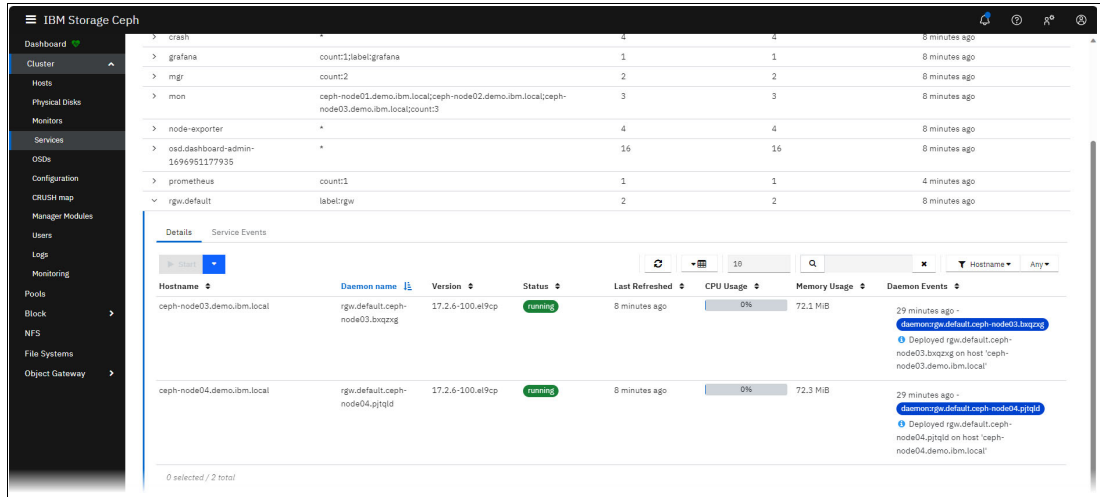


Figure 1-20 RGW services details

3. Select **Cluster** → **Pools**. You now see the three pools that were required by the rgw service to meet its storage needs (Figure 1-21).

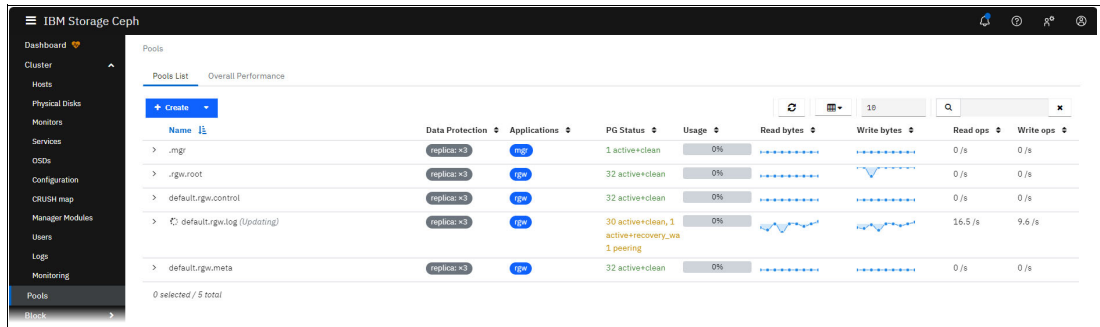


Figure 1-21 RGW service pools

## 1.5.2 Creating ingress services

To avoid a single point of failure in the Ceph RGW deployment, you need a highly available (HA) and scalable S3 and RGW endpoint that can tolerate the failure of one or more RGW services. RGW is a RESTful HTTP endpoint that can be load balanced for HA and performance with many different solutions. Since the release of Ceph 5.1, a simple way to configure HA and load balancing is to use the ingress service that is provided by cephadm, which provides an HA and load-balancing stack based on keepalived and haproxy.

Select **Cluster** → **Services**, and then **Create**. Complete the Create Service dialog box, as shown in Figure 1-22 on page 15.

The screenshot shows a 'Create Service' dialog box with the following configuration:

- Type \***: ingress
- Backend Service \***: rgw.default
- Id ? \***: rgw.default
- Unmanaged
- Placement**: Label
- Label**: rgw ✓
- Count ?**: ✓
- Virtual IP ? \***: 10.0.0.15 ✓
- Frontend Port ? \***: 8080 ✓
- Monitor Port ? \***: 8000 ✓
- CIDR Networks ?**: (empty)
- SSL

Buttons: Cancel, Create Service

Figure 1-22 Creating an ingress service

The service type is ingress. The ingress service uses the rgw default back-end service, and is deployed on the hosts with the rgw label with a virtual IP address, a front-end port, and a monitor port.

### 1.5.3 Creating the data bucket pool by using erasure coding

Now, you can create a bucket pool to store data. Automatically created RGW pools are all marked replica: x3, meaning that each object is replicated three times across nodes and OSDs for data protection. To save storage space, use an erasure coding algorithm to protect data. Erasure coding uses data and parity blocks that are encoded in a way that allows missing data to be rebuilt from parity.

Complete the following steps:

1. From the left navigation panel, select **Pools** → **Create**, and then click **Create**.
2. Specify the name `default.rgw.buckets.data`, select **erasure** as the pool type, and select the `rgw` label in Applications to deploy the pool on `rgw` hosts. Click **+** to create the Erasure Code profile (Figure 1-23).

The screenshot shows the 'Create Pool' configuration interface. It contains the following fields and options:

- Name \***: `default.rgw.buckets.data` (with a green checkmark)
- Pool type \***: `erasure` (with a green checkmark)
- PG Autoscale**: `on` (dropdown menu)
- Flags**:  EC Overwrites
- Applications**: `rgw` (with a blue edit icon and a close 'x' icon)
- CRUSH** section:
  - Erasure code profile**: `ec-4x2` (dropdown menu) with a '+' icon and a trash icon. A red arrow points to the '+' icon.
  - Crush ruleset**: `A new crush ruleset will be implicitly created.`

Figure 1-23 Creating an `rgw` data bucket

3. Create a  $k=4$ ,  $m=2$  profile, which allows the loss of two OSDs ( $m=2$ ) by distributing an object on 6 ( $k+m=6$ ) OSDs, as shown in Figure 1-24 on page 17. Click **Create EC Profile**, and then click **Create Pool**.

Create EC Profile
✕

---

**Name**  ✓

**Plugin** \* ⓘ  ▼

**Data chunks (k)** \* ⓘ  ⬆️ ✓

**Coding chunks (m)** \* ⓘ  ⬆️ ✓

**Crush failure domain** ⓘ  ▼

**Technique** ⓘ  ▼

**Packetsize** ⓘ  ⬆️ ✓

**Crush root** ⓘ  ▼

**Crush device class** ⓘ  ▼  
Available OSDs: 16

**Directory** ⓘ  ✓

Figure 1-24 Create EC Profile

**Note:** These settings are for demonstration purposes only. For a supported production environment, seven (4+2+1) Ceph nodes are required for such Erasure Coding settings.

## 1.5.4 Creating the user and its key

The cluster is ready to create buckets, users, and their keys.

Complete the following steps:

1. Select **Object Gateway** → **Users**, and then click **Create** (Figure 1-25).

The screenshot shows a 'Create User' dialog box with the following fields and options:

- User ID \***: demouser (with a green checkmark)
- Show Tenant
- Full name \***: demouser (with a green checkmark)
- Email address**: (with a green checkmark)
- Max. buckets**: Custom (with a green checkmark)
- 1000 (with a green checkmark)
- Suspended
- S3 key**:
  - Auto-generate key
- User quota**:
  - Enabled
- Bucket quota**:
  - Enabled

At the bottom, there are two buttons: 'Cancel' and 'Create User'.

Figure 1-25 Creating an RGW user

Here, you must specify only the user ID and the full name. In this example, use demouser in both cases. The user S3 key is created automatically.

After creating a user, you can create a bucket and assign it to the user.

2. Select **Object Gateway** → **Buckets**, and then click **Create**. Complete the dialog box (Figure 1-26 on page 19). The placement target corresponds to the EC pool that you created, so the bucket is protected by using erasure coding.

**Note:** Locking allows you to protect the bucket and its content from deletion for the specified period.



**Create Bucket**

**Name \*** mybucket ✓

**Owner \*** demouser

**Placement target \*** default-placement (pool: default.rgw.buckets.data)

**Locking**

Enabled ?

**Security**

Encryption ?

Cancel Create Bucket

Figure 1-26 Create Bucket

## 1.5.5 Putting data in to your bucket by using the AWS CLI

As an example, we use the AWS CLI that we previously installed on a Red Hat Enterprise Linux (RHEL) workstation. This CLI tool allows you to manipulate S3 buckets. You must configure it with the demouser's key to use it. These keys can be retrieved from the Ceph Dashboard.

Complete the following steps:

1. To get the demouser access key and secret key, select **Object Gateway** → **Users** → **demouser** → **keys** → **show**.

IBM Storage Ceph

Dashboard

Cluster

Pools

Block

NFS

File Systems

Object Gateway

Daemons

Users

Buckets

Selected Object Gateway

Object Gateway » User

+ Create

Username

dashboards

demouser

Details Keys

Show

Username Type

demouser S3

1 selected / 1 total

Max. buckets Capacity Object Limit

Limit % %

1000 No Limit No Limit

1000 No Limit No Limit

Show S3 Key

Username demouser

Access key

Secret key

Cancel

Figure 1-27 Locating the user S3 keys

- Using the RHEL workstation, open a terminal and enter `aws configure`, and then copy and paste the keys from the Dashboard (Figure 1-28).

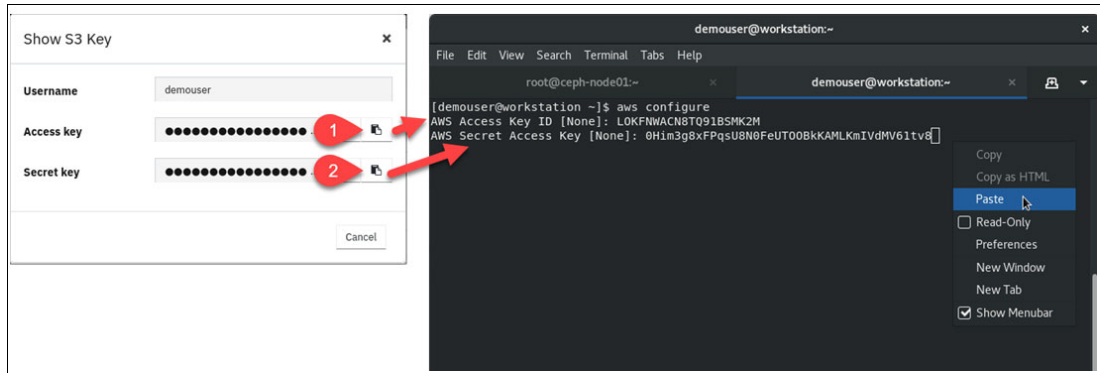


Figure 1-28 AWS CLI configuration

- Type Enter twice to keep the Default region name and Default output format as None. AWS CLI is now configured to use your bucket. Copy a file to it by entering the following command:

```
[demouser@workstation ~]$ aws --endpoint http://10.0.0.15:8080 s3 cp <some file> s3://mybucket
```

- To list the content of the bucket, use the `ls` command:

```
[demouser@workstation ~]$ aws --endpoint http://10.0.0.15:8080 s3 ls s3://mybucket
```

## 1.5.6 Checking the pool status

Going back to the Ceph Dashboard, you can also visualize the result by using Grafana. For example, you can select **Pools**, and then click **Overall Performance** (Figure 1-29).

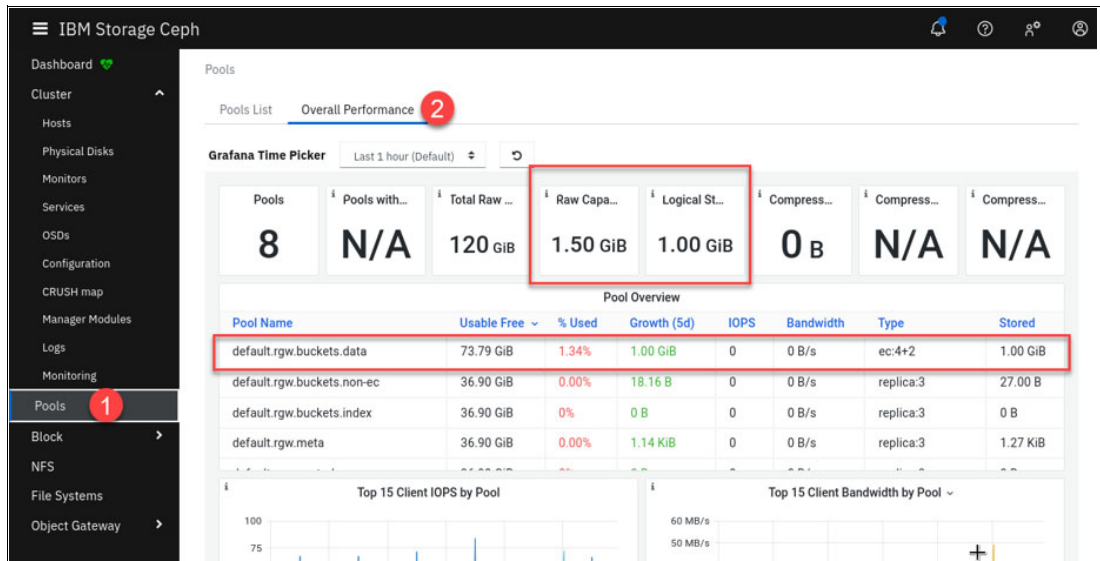


Figure 1-29 Visualizing Grafana data

You see the raw and logical space that is used on OSDs and some statistics for the `default.rgw.buckets.data` pool.

**Note:** In this capture, 73.79 GB is equivalent to 4x10 GB because of the 4x2 EC.

## 1.6 Creating a RADOS Block Device

Ceph supports block storage through the RADOS Block Device (RBD) access method for Linux distributions and Kubernetes. RBDs can be accessed either through a kernel module (Linux or Kubernetes) or through the librbd API (OpenStack or Proxmox). In Kubernetes, RBDs address the need for RWO persistent volume claims (PVCs).

### 1.6.1 Creating an RBD pool

To create a block disk, first create a pool to allow some storage by completing the following steps:

1. Select **Block** → **Images** and **Create RBD pool** (Figure 1-30).

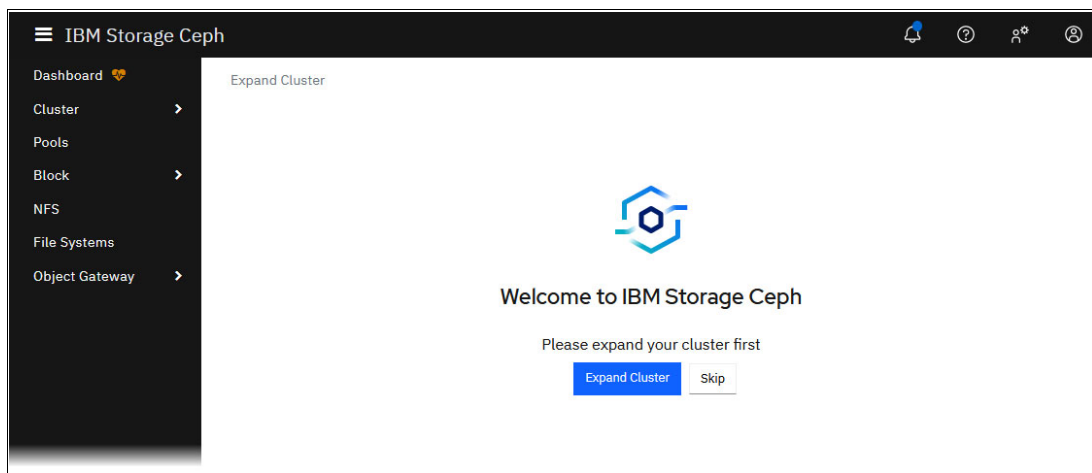


Figure 1-30 Block Image

- In this example, we name the pool `rbd_pool`, and we associate Applications with the `rbd` label (Figure 1-31).

The screenshot shows a 'Create Pool' dialog box with the following fields and values:

- Name \***: `rbd_pool` (with a green checkmark)
- Pool type \***: `replicated` (with a green checkmark)
- PG Autoscale**: `on` (dropdown menu)
- Replicated size \***: `3` (input field)
- Applications**: `rbd` (with a blue icon and a close button)
- CRUSH**
  - Crush ruleset**: `replicated_rule` (with a plus and minus icon)
- Compression**
  - Mode**: `none` (dropdown menu)
- Quotas**
  - Max bytes**: `e.g., 10GiB` (input field)
  - Max objects**: `0` (input field)
- RBD Configuration**
  - Quality of Service**: (input field)

At the bottom of the dialog, there are two buttons: 'Cancel' and 'Create Pool'.

Figure 1-31 Creating `rbd_pool`

## 1.6.2 Creating an image

Now that you have a pool for the disk, create the disk that will be made accessible. Select **Block** → **Images**, then select **Create**. Complete the dialog box to set the disk name (`disk1`), the pool to use (`rbd_pool`), and the size of the disk (Figure 1-32 on page 23).

The screenshot shows a 'Create RBD' dialog box with the following configuration:

- Name \***: disk1 (with a green checkmark)
- Pool \***: rbd\_pool (with a green checkmark)
- Use a dedicated data pool
- Size \***: 2 GiB (with a green checkmark)
- Features**:
  - Deep flatten
  - Layering
  - Exclusive lock
  - Object map (requires exclusive-lock)
  - Fast diff (interlocked with object-map)
  - Mirroring

At the bottom right, there is a blue link 'Advanced...'. At the bottom center, there are 'Cancel' and 'Create RBD' buttons.

Figure 1-32 Creating a disk

### 1.6.3 Mapping the created disk

The image is now available on the Ceph cluster, but it still must be mounted on the client file system. Because the `ceph-common` package is installed and configured on the workstation, you can use the RBD Kernel module to mount this device (Example 1-1).

Example 1-1 Mounting the image

---

```
[demouser@workstation ~]$ sudo rbd map disk1 -p rbd_pool
/dev/rbd0
[demouser@workstation ~]$ sudo mkfs.ext4 /dev/rbd0
mke2fs 1.45.6 (20-Mar-2020)
Discarding device blocks: done
Creating a file system with 524288 4k blocks and 131072 inodes
File system UUID: cc69fa65-3656-4bce-bcb3-8220b0ddbc08
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912
Allocating group tables: done
Writing inode tables: done
Creating a journal (16384 blocks): done
Writing superblocks and file system accounting information: done
[demouser@workstation ~]$ mkdir rbd_mnt
[demouser@workstation ~]$ sudo mount /dev/rbd0 rbd_mnt/
[demouser@workstation ~]$ sudo chown demouser:demouser -R rbd_mnt/
```

---

Now, you can use the `rbd_mnt` folder like any other folder in the file system, such as by moving a file into the new mounted folder. In this example, we use it to create snapshots (Example 1-2).

*Example 1-2 Moving the file into the new mounted folder*

```
[demouser@workstation ~]$ mv test.file rbd_mnt/
[demouser@workstation ~]$ ll -h rbd_mnt/test.file
-rw-r--r--. 1 demouser demouser 420M Oct 11 11:58 rbd_mnt/test.file
```

## 1.6.4 Using snapshot

When you view the **Images** tab of the Dashboard, you see the space that is used on the disk. Now, create a snapshot by completing the following steps:

1. Check the disk usage under Provision Total (Figure 1-33).

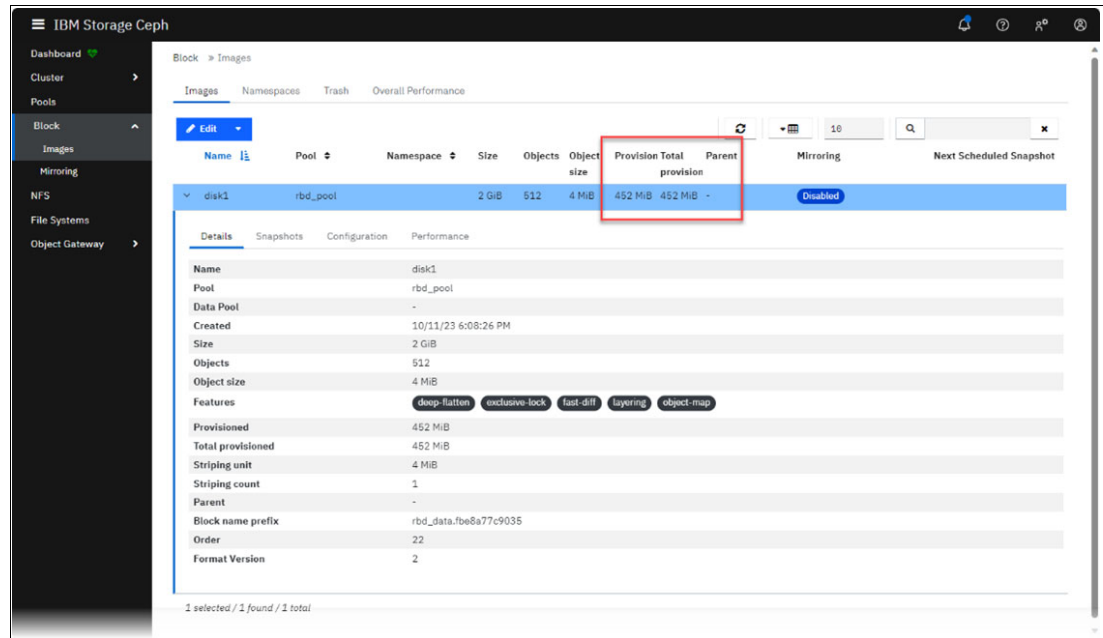


Figure 1-33 Disk usage

2. Create a snapshot by clicking the **Snapshots** tab, and then click **+ Create**. A name is automatically assigned (Figure 1-34 on page 25).

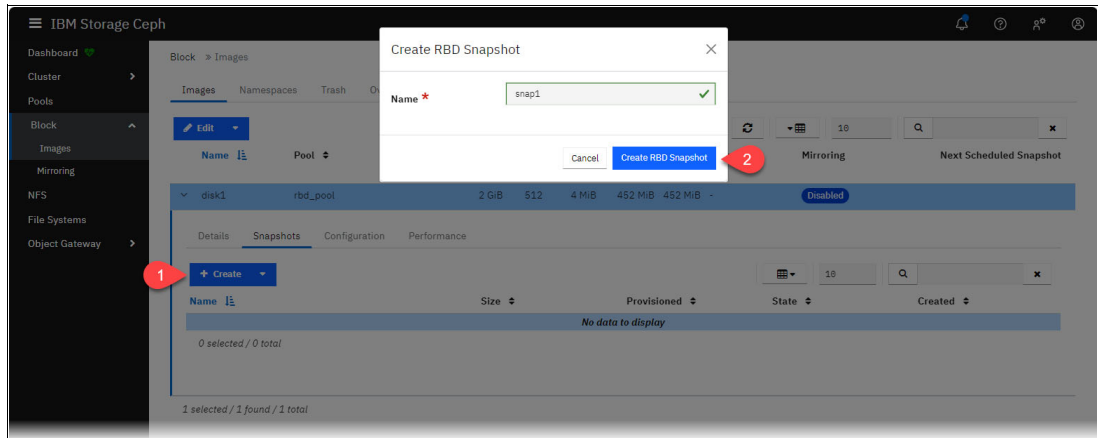


Figure 1-34 Snapshot creation

3. On the command console, delete the file (Example 1-3).

*Example 1-3 Deleting the test file*

```
[demouser@workstation ~]$ rm rbd_mnt/test.file
[demouser@workstation ~]$ ll rbd_mnt/
total 16
drwx-----. 2 demouser demouser 16384 Oct 11 11:12 lost+found
```

4. Rolling back a snapshot means overwriting the existing image. To save time, use a snapshot clone. To do so, change the snapshot from unprotected to protected by selecting **Rename** → **Unprotect** (Figure 1-35).

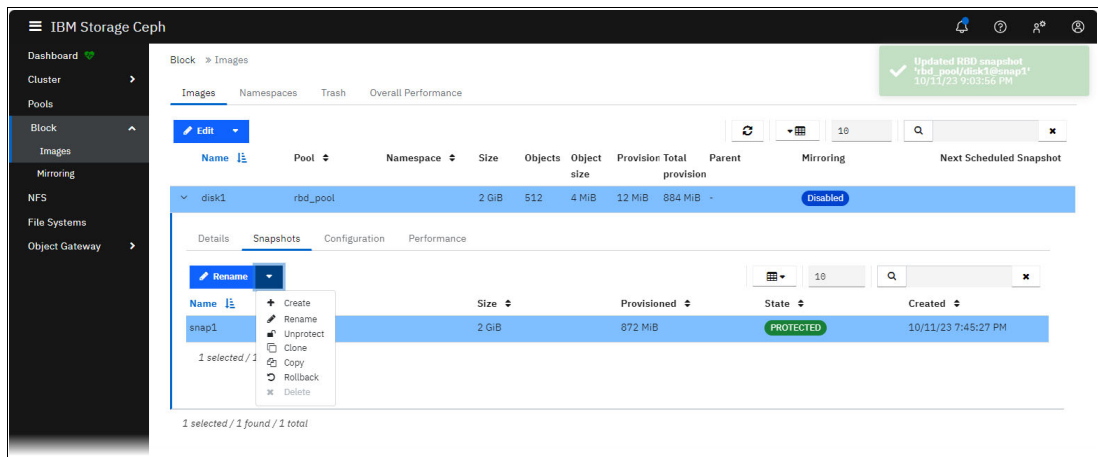


Figure 1-35 Cloning a snapshot

5. Create a clone of the snapshot. Select **Rename** → **Clone** and enter the name of the clone (Figure 1-36).

The screenshot shows a 'Clone RBD' dialog box with the following fields and options:

- Clone from:** rdb\_pool/disk1@snap1
- Name \***: disk1\_clone
- Pool \***: rdb\_pool
- Use a dedicated data pool ?
- Size \***: 2 GiB
- Features:**
  - Deep flatten
  - Layering
  - Exclusive lock
  - Object map (requires exclusive-lock)
  - Fast diff (interlocked with object-map)
  - Mirroring ?

Buttons at the bottom include 'Cancel' and 'Clone RBD'. An 'Advanced...' link is also present.

Figure 1-36 Cloning the disk1 snapshot

Figure 1-37 shows the Snapshot list.

The screenshot shows a 'Snapshot list' table with the following data:

Name	Pool	Namespace	Size	Objects	Object size	Provision Total	Parent	Mirroring	Next Scheduled Snapshot
> disk1	rdb_pool		2 GiB	512	4 MiB	12 MiB 884 MiB	-	Disabled	
> disk1_clone	rdb_pool		2 GiB	512	4 MiB	876 MiB 876 MiB	rdb_pool/disk1@sn	Disabled	

Footer: 1 selected / 2 found / 2 total

Figure 1-37 Snapshot list



6. To restore the deleted file, create a directory to mount the clone, and then map the cloned disk before mounting it in the new directory. You can now copy or move the deleted file that is in the `rnd_mnt` directory (Example 1-4).

*Example 1-4 Cloning the file*

---

```
[demouser@workstation ~]$ mkdir rbd_clone
[demouser@workstation ~]$ sudo rbd map disk1_clone -p rbd_pool
/dev/rbd1
[demouser@workstation ~]$ sudo mount /dev/rbd1 rbd_clone/
[demouser@workstation ~]$ ls rbd_clone/
lost+found  test.file
```

---

## 1.7 Creating a Ceph File System

One of the features available in a Ceph cluster is the Ceph File System (CephFS). CephFS is a POSIX-compliant file system that stores its data and metadata by using Ceph pools.

Because POSIX uses inodes to uniquely reference its content, CephFS offers a dedicated component that allows CephFS client machines to find the actual RADOS objects for an inode. This component is known as the *MetaData Server (MDS)*.

Before a CephFS client can physically access the content of a file or a directory (an inode), it contacts one of the active MDSs in the Ceph cluster to obtain the RADOS object name for the inode. When the MDS provides the CephFS client with the information, the client can contact the Object Storage Daemon that protects the placement group (PG) containing the data. The CRUSH algorithm provides the object name to OSD mapping as for any Ceph access method.

The MDS also tracks the metadata for the directory or the file, such as ACLs.

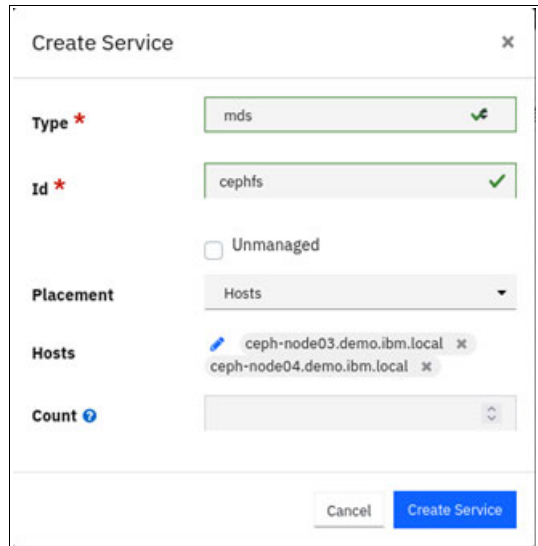
To configure a CephFS, deploy one or more MDSs (at least two to provide an HA architecture). When the MDS components are running, you can create your shared file system.

This section describes how to configure an MDS and create a file system through the Ceph dashboard. The configuration creates two Ceph pools:

- ▶ `cephfs_data` to contain the file system data.
- ▶ `cephfs_metadata` to contain the file system metadata.

Complete the following steps:

1. To create the MDS service, select **Cluster** → **Services**, and click **+ Create**. Complete the dialog box, as shown in Figure 1-38.



The screenshot shows a 'Create Service' dialog box with the following fields and values:

- Type \***: mds (with a green checkmark)
- Id \***: cephfs (with a green checkmark)
- Unmanaged**:
- Placement**: Hosts (dropdown menu)
- Hosts**: ceph-node03.demo.ibm.local, ceph-node04.demo.ibm.local (with close buttons)
- Count**: (empty numeric input field)

Buttons at the bottom: Cancel, Create Service

Figure 1-38 Creating an MDS service

2. To create the CephFS pools, select **Pools**, and then click **+ Create**. Complete the dialog box, as shown in Figure 1-39 on page 29.

### Create Pool

**Name \***  ✓

**Pool type \***  ✓

**PG Autoscale**

**Replicated size \***

**Applications** cephfs ✕

---

#### CRUSH

**Crush ruleset**  ⓘ + 🗑️

---

#### Compression

**Mode**

---

#### Quotas

**Max bytes ⓘ**

**Max objects ⓘ**

Figure 1-39 CephFS pools

3. Repeat step 2 on page 28 to create the cephfs\_metadata pool with the following values:
  - Name: cephfs\_metadata
  - Pool type: replicated
  - Applications: cephfs

When you are finished, verify your configuration (Figure 1-40).

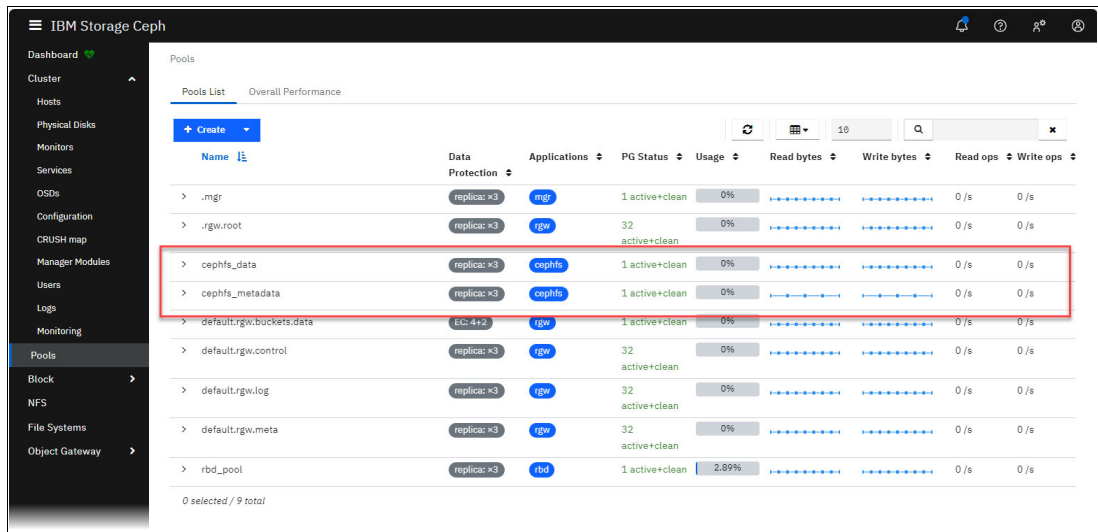


Figure 1-40 CephFS pools

4. Create a file system by opening a terminal session with root@ceph-node01 and run the commands that are shown in Example 1-5.

*Example 1-5 Creating a file system*

```
[root@ceph-node01 ~]# ceph fs new mycephfs cephfs_metadata cephfs_data
new fs with metadata pool 9 and datapool 8
[root@ceph-node01 ~]# ceph fs ls
name: mycephfs, metadata pool: cephfs_metadata, data pools: [cephfs_data ]
```

5. Create a user with read/write access to the file system (Example 1-6).

*Example 1-6 Creating a user*

```
[root@ceph-node01 ~]# ceph fs authorize mycephfs client.demouser / rw
[client.demouser]
key = AQDr/CZ12+FU0xAA56mXD8tYUbnkdeP1Kj2w==
```

6. When the key is generated, use it to mount the file system on the client by using **fstab**, for example. For this demonstration, we use **ceph-fuse**, a user-space client for CephFS file systems.

To use **ceph-fuse** on the workstation, export the key to a key ring file, and copy it to the workstation (Example 1-7).

*Example 1-7 Exporting the key to a key ring file and copying it to the workstation*

```
[root@ceph-node01 ~]# ceph auth get client.demouser -o
/etc/ceph/ceph.client.demouser.keyring
exported key ring for client.demouser
[root@ceph-node01 ~]# scp /etc/ceph/ceph.client.demouser.keyring
workstation:/etc/ceph/
```

7. Mount the file system on the workstation. In a terminal console, opened as demouser@workstation, you mount the file system (Example 1-8 on page 31).

*Example 1-8 Mounting the file system on the workstation*

```
[demouser@workstation ~]$ mkdir cephfs_mnt  
[demouser@workstation ~]$ sudo ceph-fuse --id demouser cephfs_mnt/ --client-fs  
mycephfs  
2023-10-11T14:58:30.249-0500 7f9f76cb7300 -1 init, newargv = 0x55ffab40d670  
newargc=15
```

```
ceph-fuse[19798]: starting ceph client  
ceph-fuse[19798]: starting fuse  
[demouser@workstation ~]$ sudo chown demouser:demouser -R cephfs_mnt  
[demouser@workstation ~]$ ls cephfs_mnt/
```

8. Now, you can use the file system. From the Dashboard, you can see the File Systems properties (Figure 1-41).

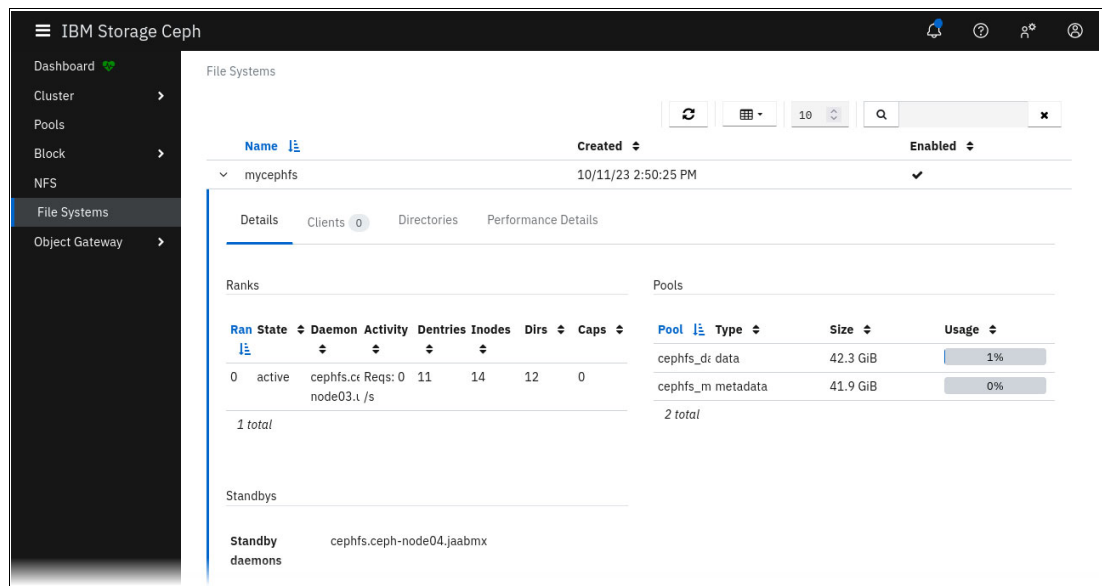


Figure 1-41 File Systems properties

## 1.8 Monitoring

The Ceph Dashboard provides monitoring information to the storage administrator.

### 1.8.1 Main Dashboard

The Dashboard main page provides a clear and concise overview of the cluster's health. The Inventory section lists all cluster components, their health status (indicated by a tick or cross), and a direct link to the related page (in blue text) (Figure 1-42).

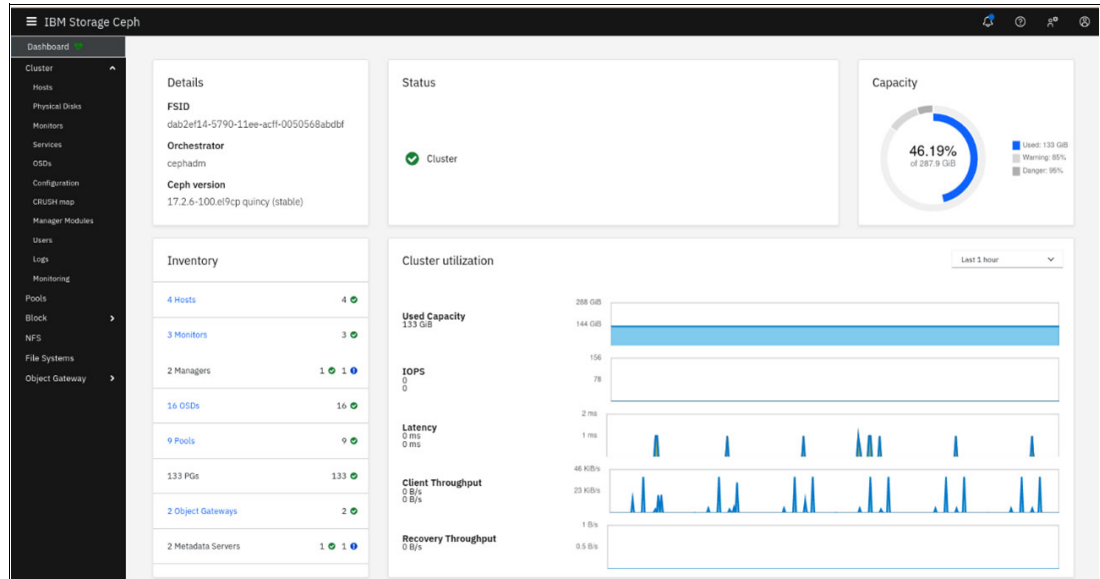


Figure 1-42 Dashboard monitoring

The bell icon in the upper right provides access to the last active notifications. Solved notifications are automatically deleted from this list. Some failure notifications might even provide the storage administrator with solutions to apply.

### 1.8.2 Cluster menu

This menu provides access to the cluster configuration components. Hosts and OSDs also have an **Overall Performance** tab, which displays the Grafana data.

Figure 1-43 on page 33 shows the Hosts performance data.

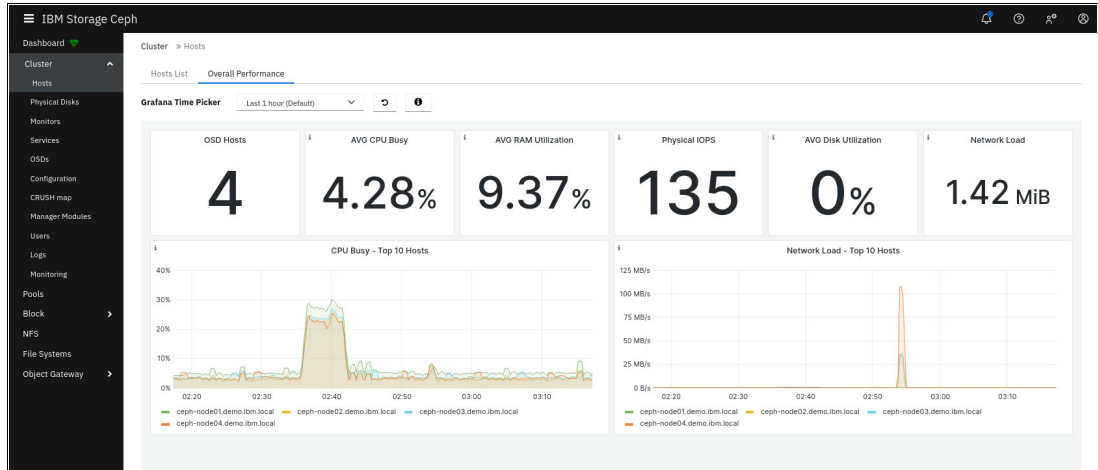


Figure 1-43 Hosts overall monitoring

Figure 1-44 shows the OSDs performance data.

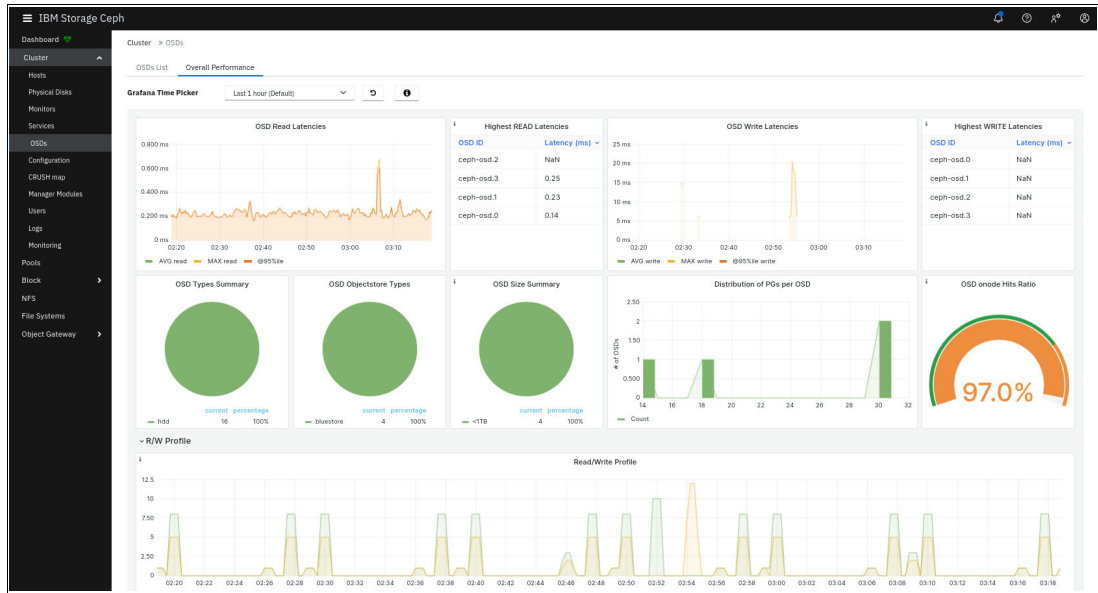


Figure 1-44 OSDs overall monitoring

Each of these views can be focused on one host or one OSD. The example in Figure 1-45 shows ceph-node01 performance details.

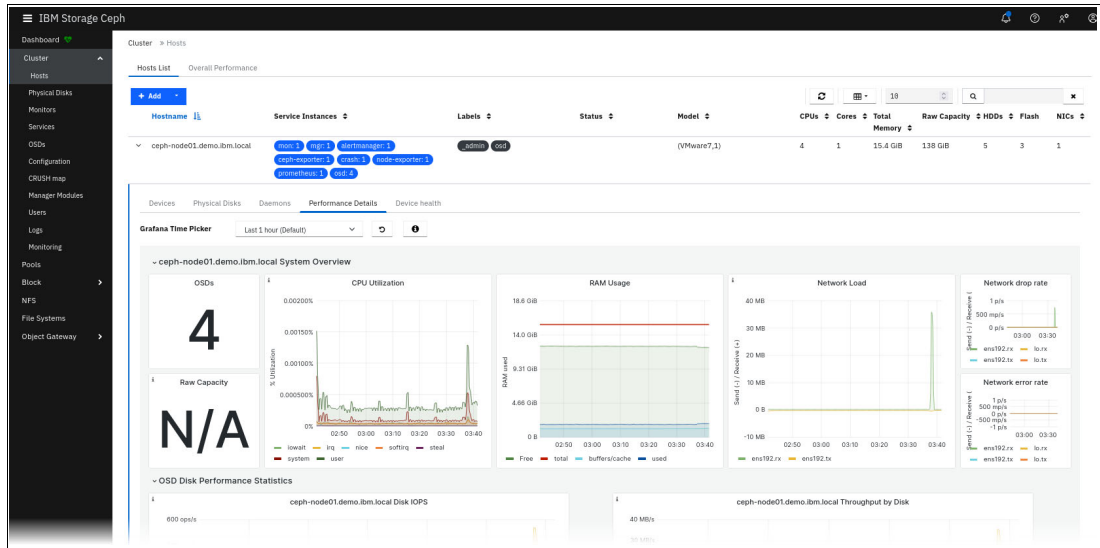


Figure 1-45 Host detail monitoring

The **Cluster** menu also provides access to the logs, which is helpful for sorting events when troubleshooting a problem (Figure 1-46).

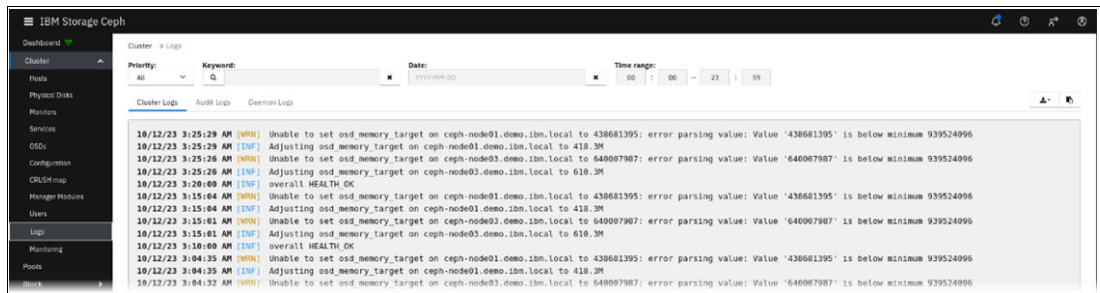


Figure 1-46 Logs monitoring

The **Monitoring** menu can list active and previous alerts with detailed information about them, but the most useful tab might be the **Silences** tab. By using the **Silences** tab, you can silence specific alerts when doing maintenance on the cluster to avoid flooding the logs (Figure 1-47).

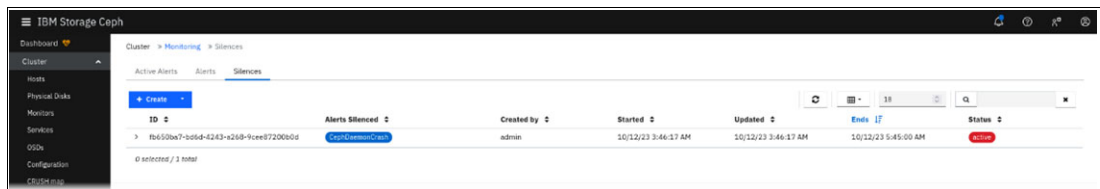


Figure 1-47 Monitoring silences



### 1.8.3 Other Grafana performance data

Grafana also displays performance data for pools, block images, file systems, and object gateways. Figure 1-48 shows an example on a CephFS pool.

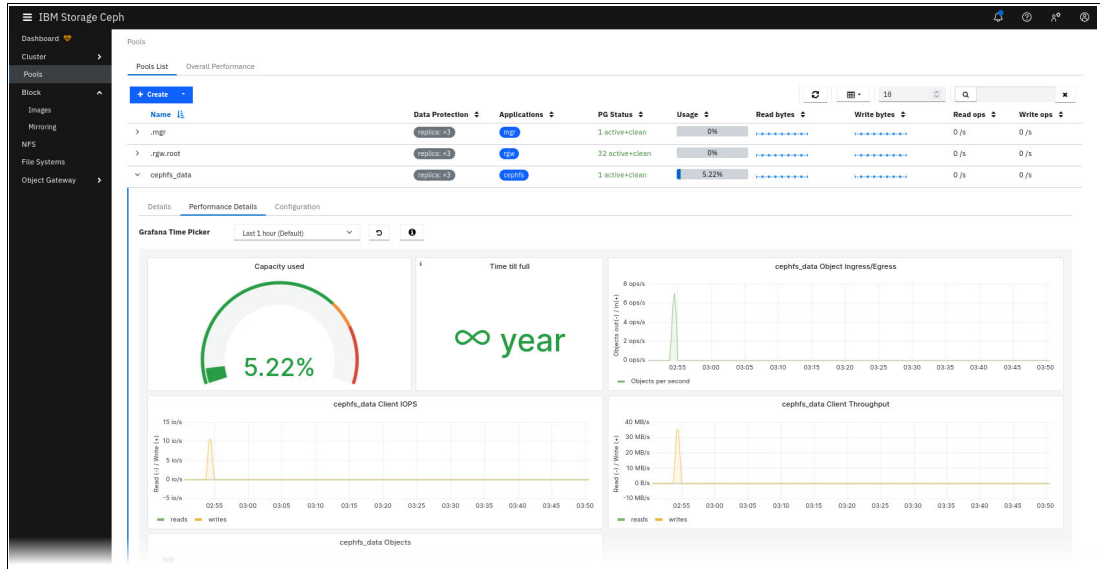


Figure 1-48 Pool monitoring





# IBM Storage Ceph for IBM watsonx.data

IBM watsonx.data empowers enterprises to scale their analytics and AI capabilities with a purpose-built data store, which leverages an open lakehouse architecture. Through its robust querying, governance, and open data formats, IBM watsonx.data facilitates seamless data access and sharing. With IBM watsonx.data, you can swiftly connect to data, extract actionable insights, and optimize data warehouse expenses.

IBM Storage Ceph can be used as an efficient way to build a data lakehouse for IBM watsonx.data and next-generation AI workloads.

This chapter covers the configuration steps for integrating IBM Storage Ceph and IBM watsonx.data.

This chapter has the following sections:

- ▶ Using IBM Storage Ceph with IBM watsonx.data
- ▶ Creating users
- ▶ Creating buckets
- ▶ Performance optimizations
- ▶ Querying less structured data (S3-Select)

**Note:** For more information about IBM watsonx.data with IBM Fusion HCI, see *Accelerating IBM watsonx.data with IBM Fusion HCI*, REDP-5720.

## 2.1 Using IBM Storage Ceph with IBM watsonx.data

IBM watsonx.data and IBM Storage Ceph enable unified access to your data across databases (DBs) and data lakes, enabling you to optimize each configuration within a single IBM Storage Ceph cluster. You can share large volumes of data through open table formats like Apache Iceberg, which is designed for high-performance analytics and large-scale data processing, while simultaneously storing vast amounts of data for various large dataset analyses.

IBM Storage Ceph also supports multiple vendor open formats for analytic datasets and enables different engines to access and share data by using tools like Parquet, Avro, Apache Optimized Row Columnar (ORC), and more. This chapter covers the configuration steps for integrating IBM Storage Ceph and IBM watsonx.data.

## 2.2 Creating users

The first step is to create an IBM watsonx user by using the Ceph Dashboard or the CLI.

### 2.2.1 Creating a user by using the Ceph Dashboard

To create a user by using the Ceph Dashboard, complete the following steps:

1. Using the menu in the left pane, browse to the Object Gateway section and click the **Users** tab to open the pane for user management. Click **Create** to open the user creation page (Figure 2-1).

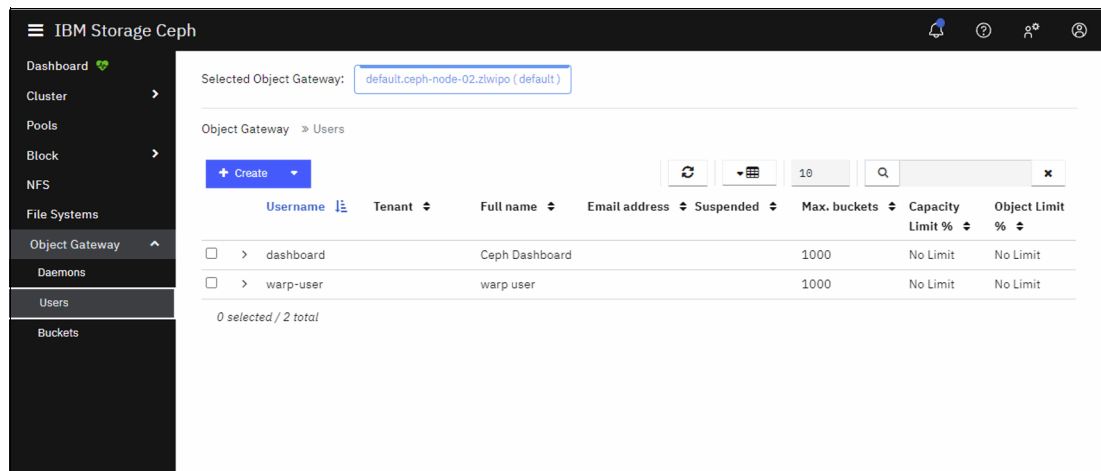


Figure 2-1 User creation page

- The user creation page prompts you for some information about the user. Complete the mandatory fields, which are indicated by the red asterisks. When finished, click **Create User** (Figure 2-2).

The screenshot shows the IBM Storage Ceph user creation interface. On the left is a dark sidebar with navigation items: Dashboard, Cluster, Pools, Block, NFS, File Systems, Object Gateway (selected), Daemons, Users, and Buckets. The main content area has a header 'Selected Object Gateway: default.ceph-node-02.zlwipo (default)'. Below it is a breadcrumb 'Object Gateway > Users > Create'. The 'Create User' form contains the following fields and options:

- User ID \***: Input field with 'watsonx' and a green checkmark.
- Show Tenant
- Full name \***: Input field with 'Watson X' and a green checkmark.
- Email address**: Input field with 'watsonx@example.org' and a green checkmark.
- Max. buckets**: Input field with 'Custom' and a green checkmark.
- Input field with '1000'.
- Suspended
- S3 key**: Section header.
- Auto-generate key
- User quota**: Section header.
- Enabled
- Bucket quota**: Section header.
- Enabled

At the bottom right of the form are two buttons: 'Cancel' and 'Create User' (highlighted in blue).

Figure 2-2 User creation

- After you create a user, the user management pane should reflect the addition of the new user. Click the arrow to see information about the newly created user (Figure 2-3).

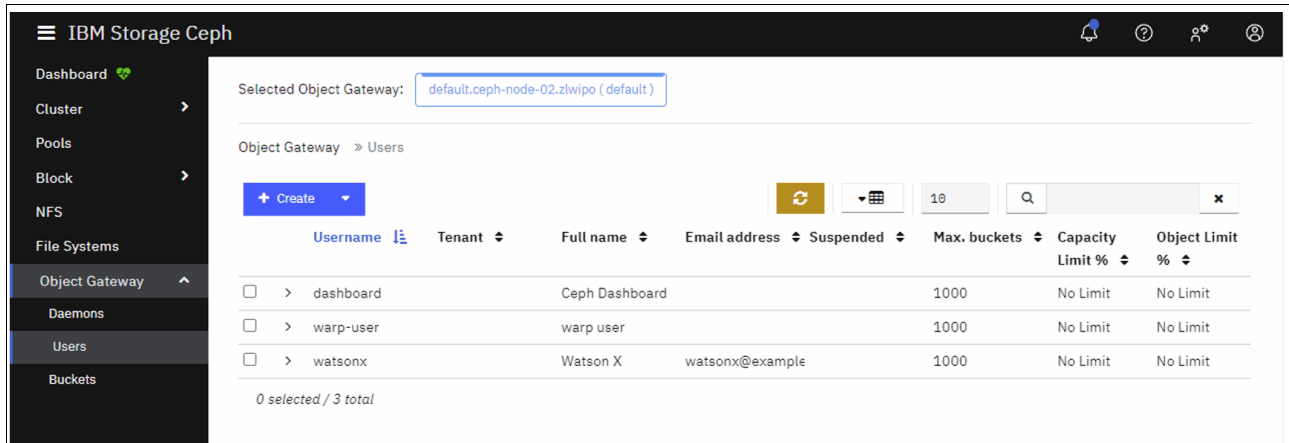


Figure 2-3 Newly created user

- The user information is displayed on the **Details** tab. From there, click the **Keys** tab (Figure 2-4).

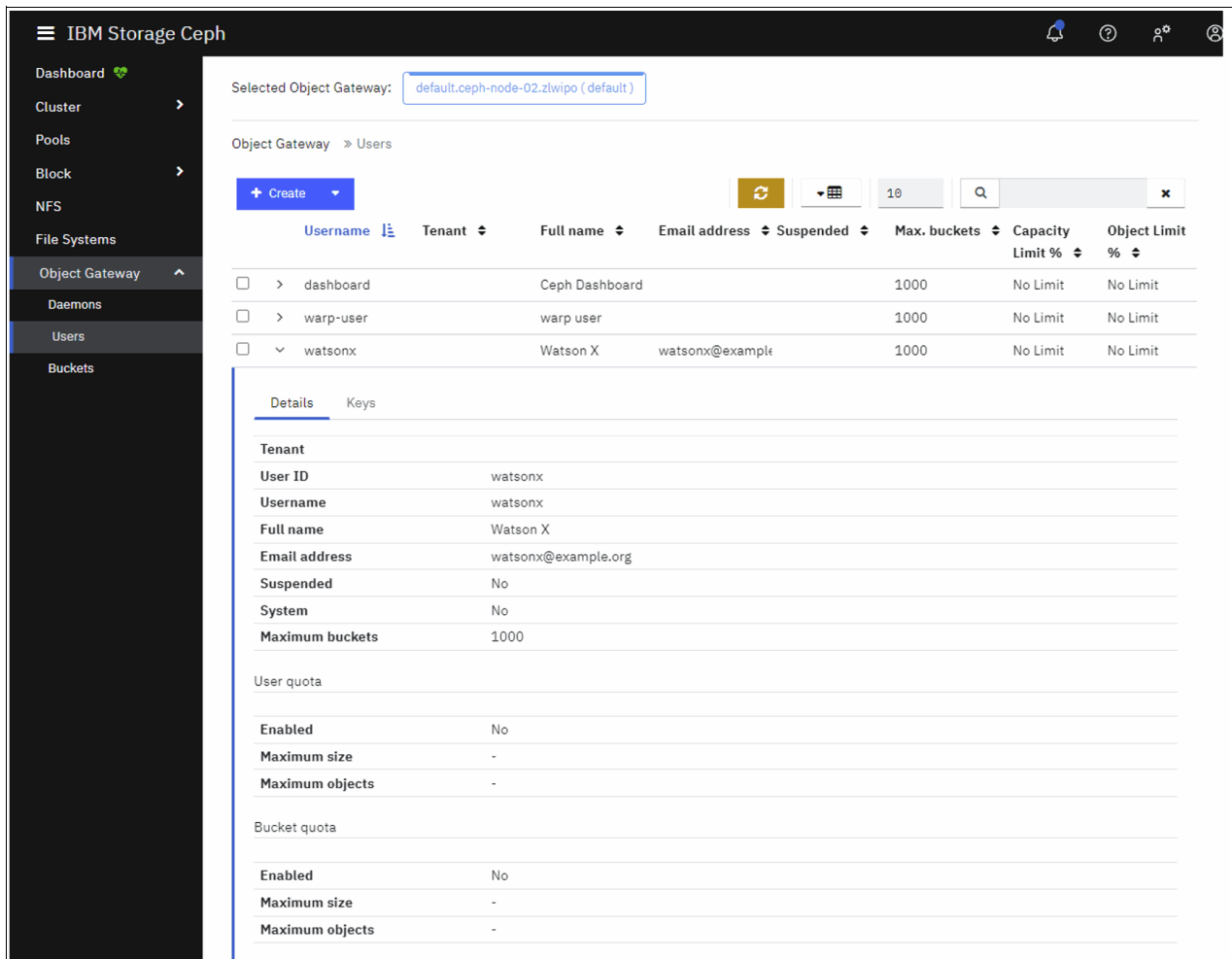


Figure 2-4 User details

- After selecting the **Keys** tab, click **Show** to view or copy the user's credentials (Figure 2-5).

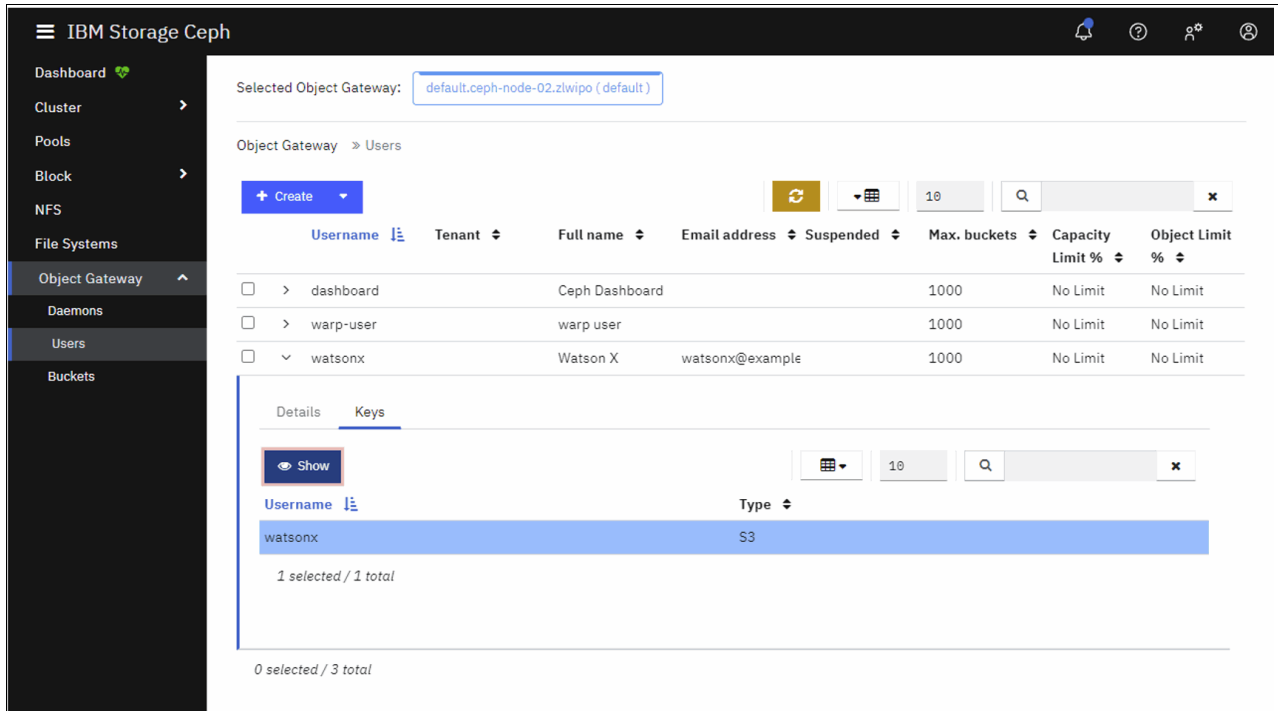


Figure 2-5 Viewing or copying the user's credentials

- The access and secret keys are needed to add a bucket into the watsonx.data Infrastructure Manager (Figure 2-6).

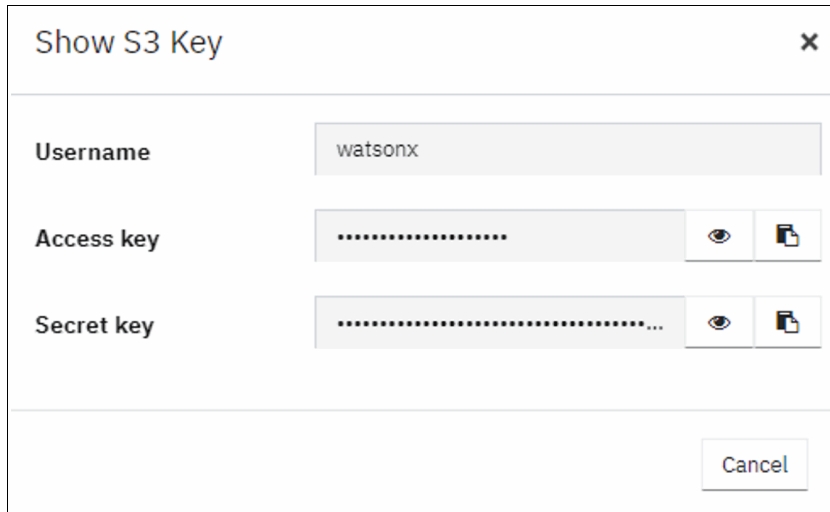


Figure 2-6 Show S3 Key window

## 2.2.2 Creating a user by using the CLI

Alternatively, you can create users by using the CLI, as described in Example 2-1.

*Example 2-1 Creating a user by using the CLI*

---

```
radosgw-admin user create \  
  --uid watsonx \  
  --display-name "Watson X"
```

---

The command output includes the access and secret keys that are needed to add a bucket to watsonx.data from the Infrastructure Manager.

## 2.3 Creating buckets

This section describes how to create buckets. To create buckets, use either the Ceph Dashboard or the CLI.

### 2.3.1 Creating a bucket with the Ceph Dashboard

To create a bucket with the Ceph Dashboard, complete the following steps:

1. Using the menu in the left pane, browse to the Object Gateway section and click the **Buckets** tab to open the pane for bucket management. Click **Create** to open the bucket creation page (Figure 2-7 on page 43).



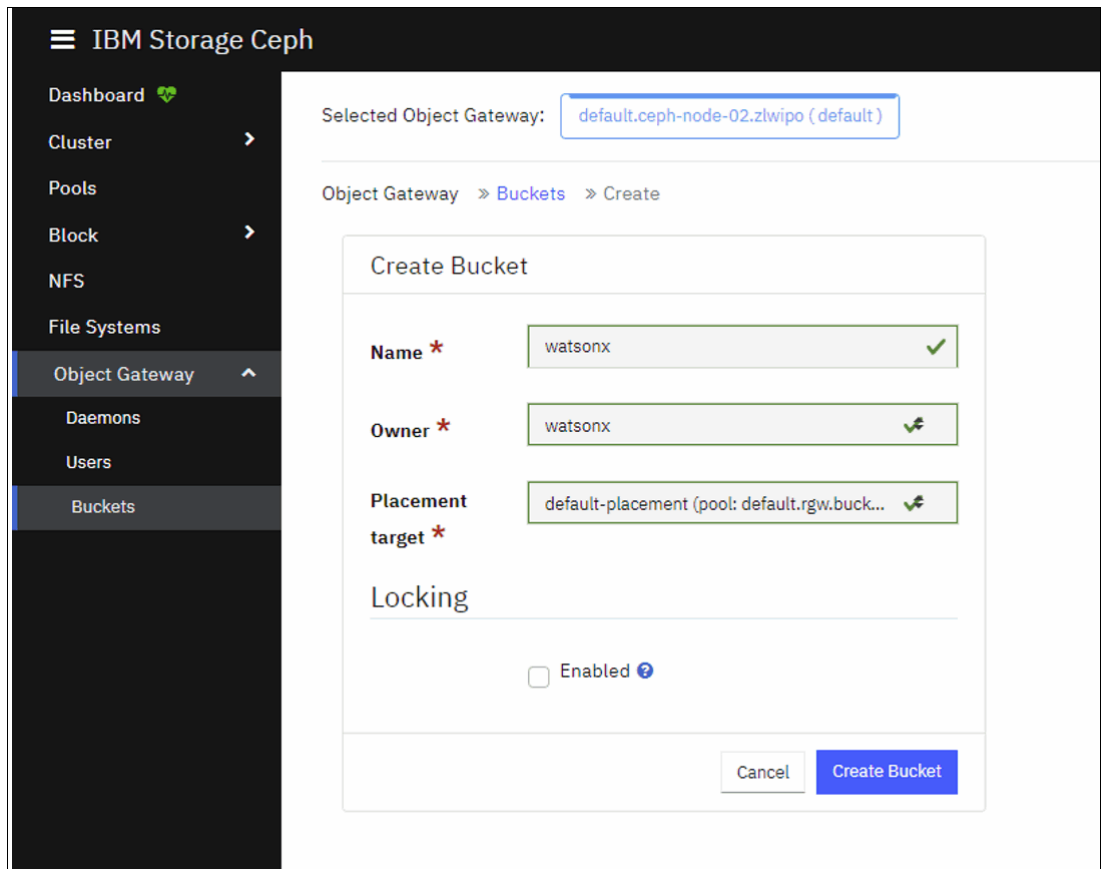


Figure 2-7 Bucket creation page

2. The bucket creation page prompts you for information about the bucket that you want to create. Complete all mandatory fields, as indicated by the red asterisks. Ensure that the user that was created in 2.2, “Creating users” on page 38 is set as the bucket owner.
3. When finished, click **Create Bucket**.

### 2.3.2 Creating a bucket by using the CLI

As a best practice, use the `s5cmd` utility for interacting with the Ceph Object Gateway through the CLI. The utility is fast and portable.

Using a credentials file eliminates the need to source environmental variables every time that you start a new shell session. The commands that are shown in Example 2-2 create a credentials file in the location that is expected by `s5cmd` and many other Amazon S3 utilities that use the S3 SDKs.

*Example 2-2 Creating a credentials file*

---

```
mkdir ~/.aws
chmod 0700 ~/.aws
cat << EOF > ~/.aws/credentials
aws_access_key=< your access key >
aws_secret_access_key=< your secret key >
EOF
```

---

With a credentials file configured, you can now create a bucket by using the following command:

```
s5cmd --endpoint-url http://s3.ceph.example.com mb s3://bucket-name
```

To avoid needing to specify the endpoint URL with each command, you can create bash aliases:

```
alias s5cmd=' s5cmd --endpoint-url http://s3.ceph.example.com'
```

The `alias` command can be added to your `.bashrc` file to ensure that it is configured whenever you start a new session.

## 2.4 Adding a bucket in IBM watsonx.data

To add a bucket in IBM watsonx.data, complete the following steps:

1. From the Lakehouse user interface, browse to the Infrastructure manager section. From the Infrastructure Manager, click **Add Component** in the left pane (Figure 2-8).

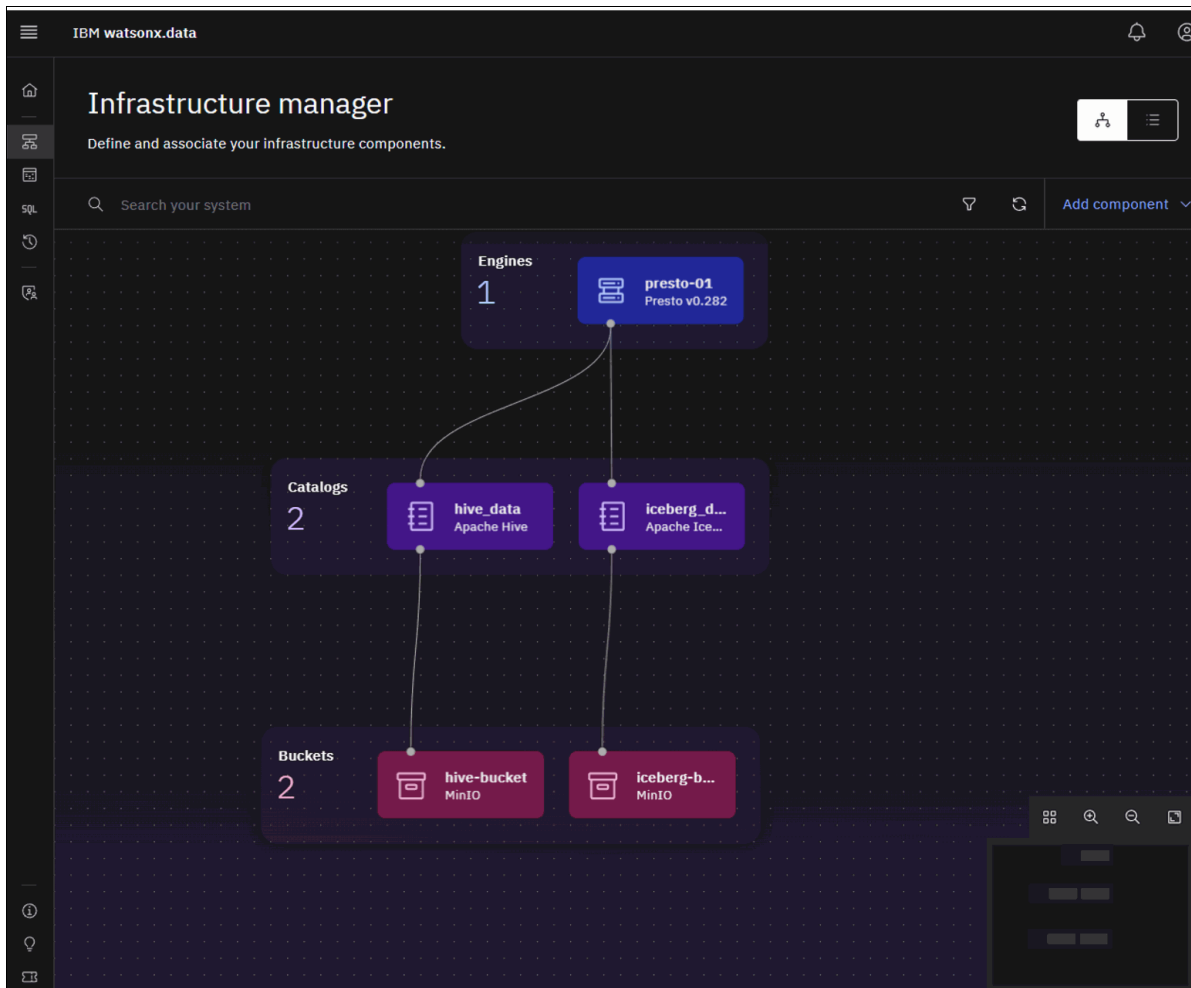


Figure 2-8 Infrastructure manager: Add Component

2. Select **Add Bucket** (Figure 2-9 on page 45).

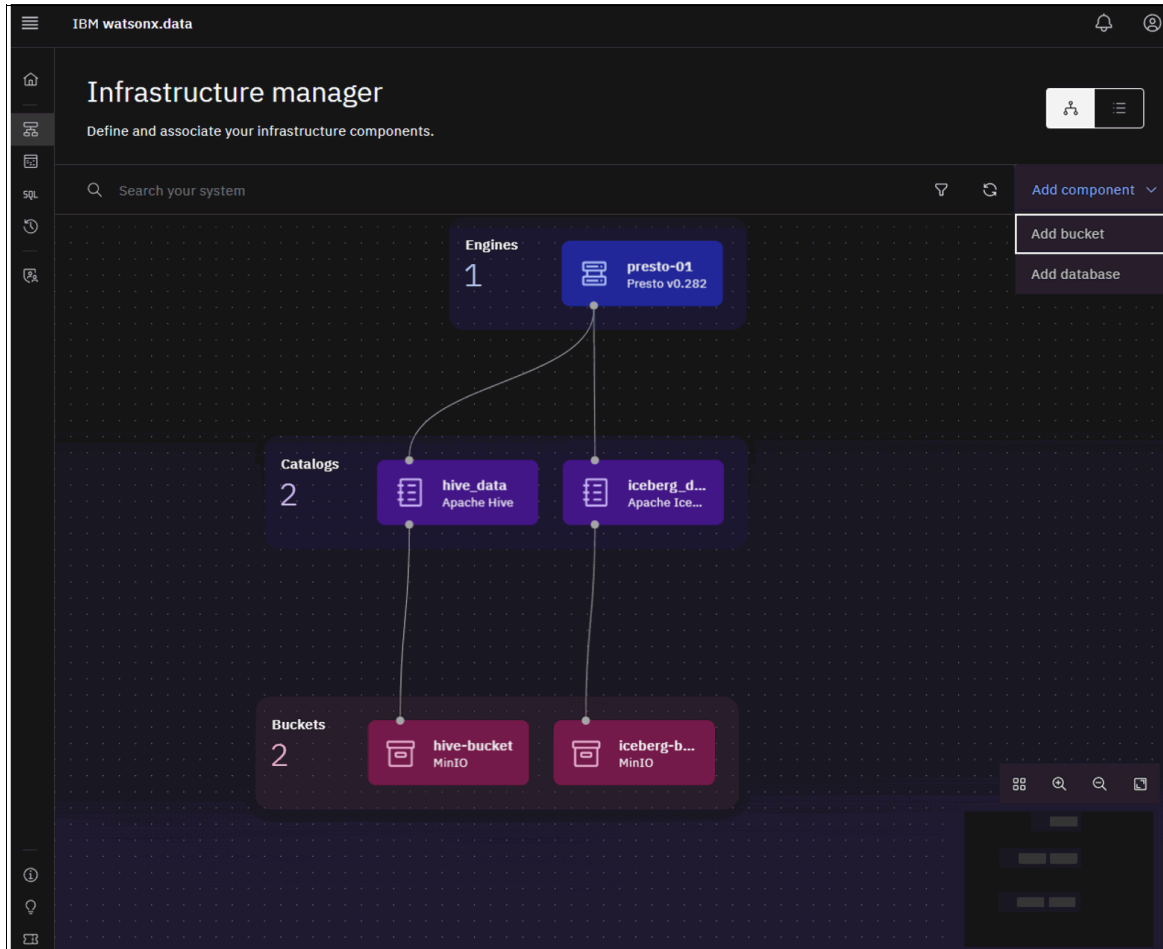


Figure 2-9 Infrastructure manager: Add Bucket

The Add Bucket prompt opens, where you provide information about the bucket being added to IBM watsonx.data.

3. Select **IBM Storage Ceph** from the **Bucket type** menu. Then, populate the fields for the bucket name, the endpoint for the IBM Storage Ceph cluster that is being used with IBM watsonx.data, and the access and secret keys (Figure 2-10).

**Add bucket**  
Register existing, externally managed object storage.

**Bucket details**

Bucket type: IBM Cloud Object Storage (dropdown)  
Region: Dallas (us-south) (dropdown)

Bucket name: Example: your-bucket-01  
Display name: Example: Your Bucket 01

Endpoint: Format: https://s3.<region>.cloud-object-storage.appdomain.clo

Access key: Enter your access key (with eye icon)  
Secret key: Enter your secret key (with eye icon)

Connection status: ⚠ Untested Test connection (with key icon)

Activation:  Activate now (terminates in-flight queries against data in any bucket)

**Associated catalog** ⓘ

Catalog type: Apache Iceberg (dropdown)  
Catalog name: Example: your\_catalog\_01

Cancel Register

Figure 2-10 Add bucket window

4. When these steps are completed, perform a connection test to verify that everything is functioning properly (Figure 2-11).

**Add bucket**  
Register existing, externally managed object storage.

**Bucket details**

Bucket type  
IBM Storage Ceph

Bucket name: watsonx      Display name: watsonx

Endpoint  
http://208.113.130.246:7480

Access key: .....      Secret key: ..... (highlighted)

Connection status  
⚠ Untested      Test connection (p)

Activation  
 Activate now (terminates in-flight queries against data in any bucket)

**Associated catalog** ⓘ

Catalog type  
Apache Iceberg

Catalog name  
Example: your\_catalog\_01

Cancel      Register

Figure 2-11 Testing the connection

5. On a successful connection, a Successful message along with a green checkmark is displayed (Figure 2-12).

The screenshot shows a dark-themed 'Add bucket' configuration window. At the top, it says 'Add bucket' and 'Register existing, externally managed object storage.' Below this is a 'Bucket details' section with the following fields: 'Bucket type' set to 'IBM Storage Ceph', 'Bucket name' and 'Display name' both set to 'watsonx', and 'Endpoint' set to 'http://208.113.130.246:7480'. There are two password fields for 'Access key' and 'Secret key', both masked with dots and having eye icons. Below these is a 'Connection status' section showing a green checkmark and the word 'Successful', with a 'Retest' button next to it. Underneath is an 'Activation' section with a checkbox labeled 'Activate now (terminates in-flight queries against data in any bucket)'. The bottom section is 'Associated catalog' with a dropdown for 'Catalog type' set to 'Apache Iceberg' and a text field for 'Catalog name' containing 'Example: your\_catalog\_01'. At the very bottom are 'Cancel' and 'Register' buttons.

Figure 2-12 Connection test successful

6. Indicate whether the bucket should be activated now (Figure 2-13 on page 49).

**Important:** Activating the bucket terminates any in-flight queries against data in any bucket.

**Add bucket**  
Register existing, externally managed object storage.

**Bucket details**

Bucket type  
IBM Storage Ceph

Bucket name: watsonx      Display name: watsonx

Endpoint  
http://208.113.130.246:7480

Access key: .....      Secret key: .....

Connection status  
Successful      Retest

Activation  
 Activate now (terminates in-flight queries against data in any bucket)

**Associated catalog**

Catalog type  
Apache Iceberg

Catalog name  
watsonx

Cancel      Register and activate now

Figure 2-13 Indicating whether the bucket should be activated now

- The bucket is associated with a catalog. Supported catalog types include Apache Iceberg and Apache Hive. *Apache Iceberg is the preferred catalog type when using IBM Storage Ceph to persist lakehouse data.* Iceberg tables are designed to work well with object stores, and engines spend less time planning queries that operate against large tables.

## 2.4.1 Associating a catalog

To associate a catalog with an engine, complete the following steps:

1. Browse to the Infrastructure manager page. Hover your cursor over the wanted catalog in the upper right and click the connected dot icon (Figure 2-14).

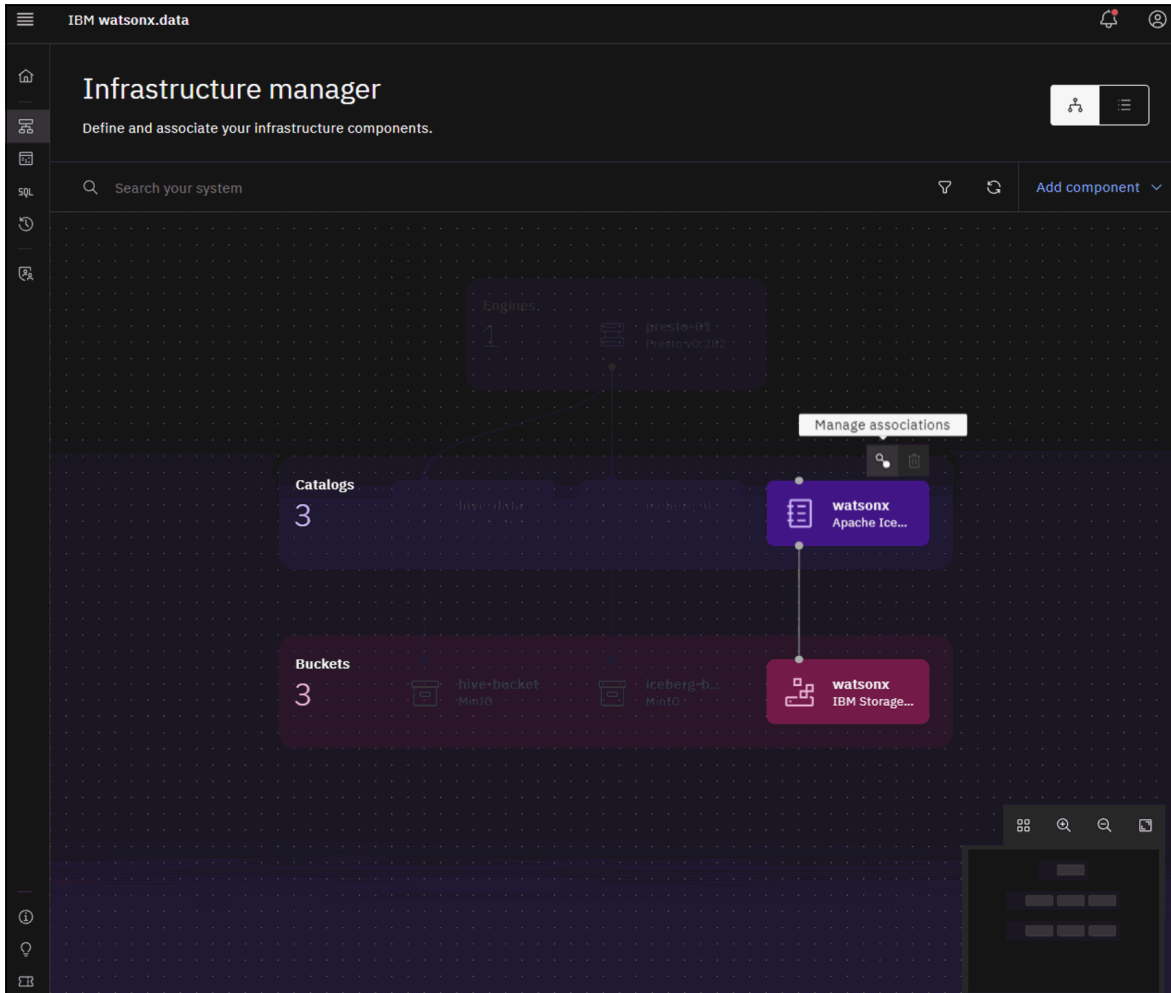


Figure 2-14 Infrastructure manager: Manage associations

2. You see a prompt where you can select an engine to associate with the catalog (Figure 2-15 on page 51).



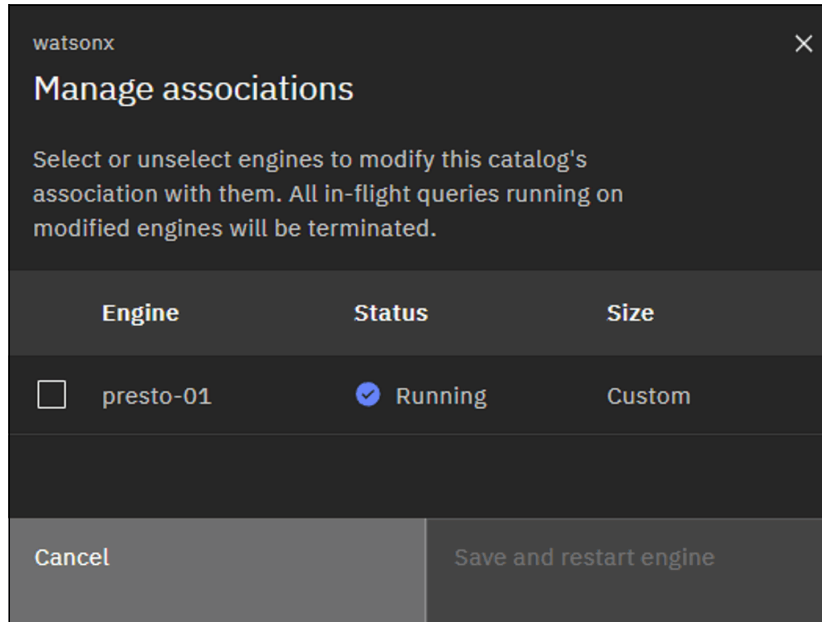


Figure 2-15 Manage associations

3. Once you select the engine, click **Save and restart engine**.

## 2.4.2 Schema organization in a bucket

Schemas should be placed in a pseudo-directory and not at the root of the bucket to avoid errors. For example, use `s3a://bucket/schema` or `s3a://warehouse/catalog/schema` instead of using `s3a://bucket/<files>`. Using the root of the bucket might result in errors. For example, the command in Example 2-3 places the schema in the `s3://watsonx/mycatalog/myschema/` directory.

*Example 2-3 Placing a schema into a directory*

---

```
touch a && s5cmd --endpoint-url s3.ceph.example.com cp a s3://watsonx/mycatalog/myschema/
```

---

Complete the following steps:

1. Returning to the Infrastructure manager window, click the maroon bucket icon to open the Bucket details window (Figure 2-16).

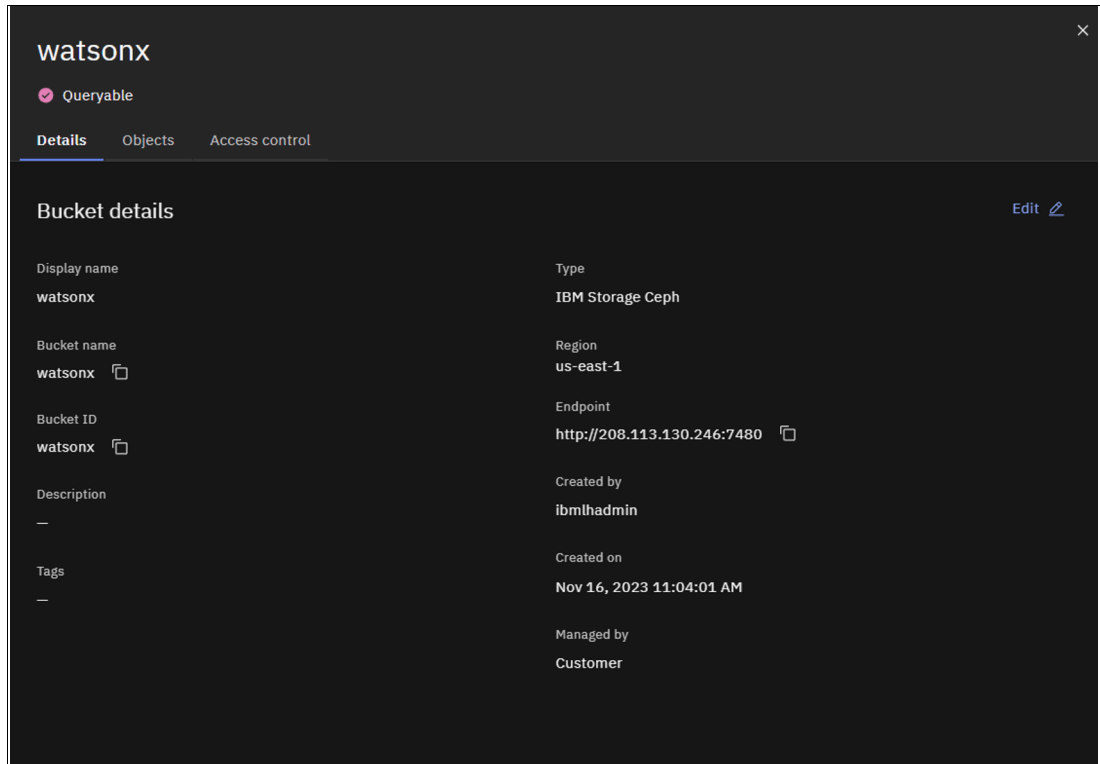


Figure 2-16 Bucket details window

2. You can view the pseudo-directory that was established through the CLI by browsing to the **Objects** tab from the Bucket details window (Figure 2-17).

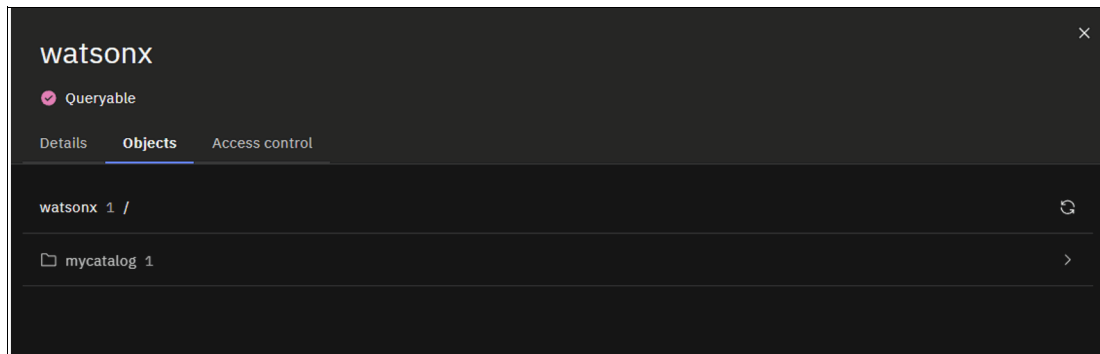


Figure 2-17 Bucket details page: Objects tab

### 2.4.3 Creating a schema

After verifying that the bucket has the necessary structure to create a schema, you can use the Data Manager window to create the schema. Complete the following steps:

1. Click **Create** (Figure 2-18 on page 53).

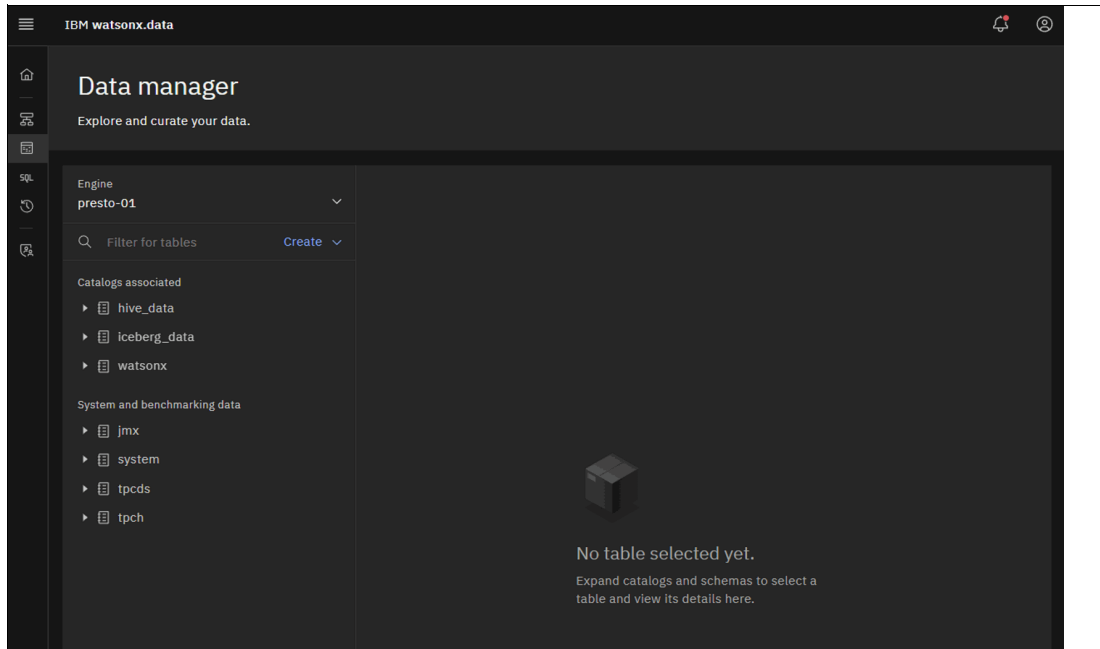


Figure 2-18 Data manager

2. To create a schema, select **Create schema** from the drop-down menu. A prompt appears, where you can define your schema's details. Select the catalog that is associated with the IBM Ceph Storage bucket. Choose a descriptive name and use the path that was created in 2.4.2, "Schema organization in a bucket" on page 51. Click **Create** (Figure 2-19).

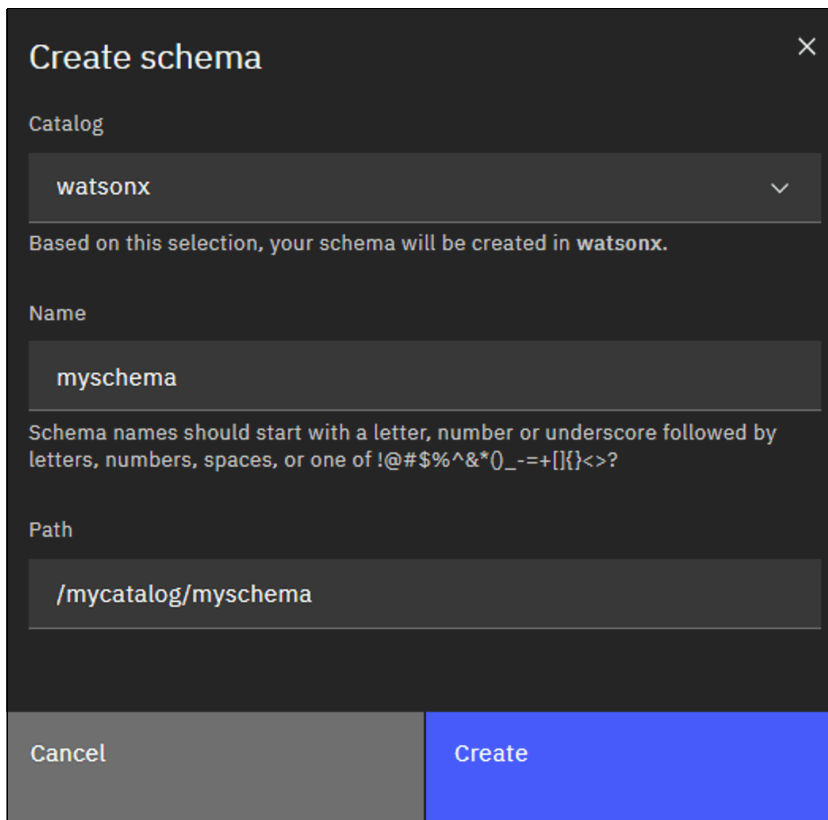


Figure 2-19 Create schema

## 2.5 Performance optimizations

Before attempting to load or ingest data to the newly created schema, it is worth taking the time to understand various techniques that can be applied during ingest to help reduce query latency.

### 2.5.1 Table format

Although different engines might support various formats, the most common formats are Hive and Apache Iceberg. As mentioned in 2.4, “Adding a bucket in IBM watsonx.data” on page 44, Apache Iceberg is a well-suited table format for use with object storage. Iceberg employs manifest lists and manifest files to track which objects correspond with a point in time for a table. In contrast, Hive tables use pseudo-directory structure. When an engine is planning a query for execution with an Iceberg table, it performs a few reads against the manifests to get a list of all the objects that must be operated on. When an engine is planning a query for execution with a Hive table, it must recursively list the pseudo-directory structure 1,000 keys at a time. *For large tables, the Hive approach can take longer and put more load on the underlying object store. In fact, this capacity was an influential factor in the design of Apache Iceberg.*

### 2.5.2 Columnar formats

Using columnar formats like Parquet and Apache ORC empowers query engines to perform projection efficiently. As a result, the engine retrieves only the row groups of relevant columns, which outperform whole-object reads. Internally, the query engine initiates the process by reading a footer containing offset information for columns and their corresponding row groups. Then, the query engine can issue parallel ranged requests for the objects containing the columnar data.

### 2.5.3 Partitioning

Partitioning tables that are based on the values in a specific column can reduce query processing time, particularly for queries involving predicate filtering (**WHERE** statements). A best practice is to partition tables containing order or sales data by a specific period, such as day, month, or year. This approach enables query engines to eliminate irrelevant partitions, which minimizes the number of reads from the storage system and accelerating query execution.

### 2.5.4 Bucketing

Bucketing can speed up queries where there is a filter on a column that was used for bucketing by using a hash function to identify the objects for the corresponding bucket. The buckets containing data that is not relevant to the query are pruned, which results in fewer reads to the storage system and faster query times. The downside of bucketing is that data cannot be inserted into the table by introducing new objects. Instead, the buckets must be rewritten.

## 2.5.5 Table statistics

When generating a query plan, table statistics are leveraged by query optimizers to make better decisions about which partitions can be pruned from a query plan. Most engines support the **ANALYZE** statement, which can be used to collect table and column statistics for Hive tables. Apache Iceberg tables improve on this task by storing partition statistics in table metadata. Statistics are calculated when writing to Iceberg tables, which have advantages over periodic table and column analysis.

## 2.6 Querying less structured data (S3-Select)

Occasionally, there are tools writing a large volume of data in formats that are less optimized for analysis, such as CSV. Many of the techniques in 2.5, “Performance optimizations” on page 54 are not possible when dealing with CSV data. Some query engines, notably Presto, can work in concert with advanced object stores to process large volumes of CSV in the most efficient way possible. This capability is enabled by S3-Select, which empowers query engines to delegate both column projection and predicate evaluation to the storage system. By offloading this processing to the storage layer, data filtering is performed where it is most efficient. This approach substantially reduces the volume of data that is transferred over the network and optimizes the computational resources that are required for query execution.

The Hive session property `s3_select_pushdown_enabled` is required to use S3-Select pushdown. Hive sessions properties can be used only when submitting queries through the CLI. If a Hive session parameter is set in the Query workspace, a successive statement is submitted as a new session without the session parameter.

For more information, see [S3 select operations \(technology preview\) - IBM Documentation](#).

### 2.6.1 Bucket versioning

Versioning can be enabled on buckets to prevent accidental deletion of data. With versioning, if an object is overwritten with new data, then the previous version is maintained until it is explicitly deleted. For further protection, MFA-delete can be configured on a versioned bucket so that delete operations require an additional multi-factor authentication token to be supplied during the request. Combined, versioning and MFA-delete can defend against accidental deletion and malicious deletion by ransomware. Versioning can be set on a bucket with the following command:

```
./s5cmd --endpoint-url http://s3.ceph.example.com bucket-version s3://watsonx
```

For more information, see the following links:

- ▶ [S3 put bucket versioning - IBM Documentation](#).
- ▶ [Ceph Object Gateway and multi-factor authentication - IBM Documentation](#).

### 2.6.2 Bucket lifecycle

The lifecycle configuration enables a set of rules to be defined that trigger actions on objects by the object store. Supported actions include moving the object from one storage class to another one and expiration. IBM Storage Ceph supports the configuration of multiple storage classes that place data on different types of storage media or use different durability strategies (that is, replication versus erasure coding). Using a lifecycle can result in cost savings through the elimination of data or by moving data to less costly storage media.

For more information, see [Bucket lifecycle - IBM Documentation](#).

### 2.6.3 Bucket notifications

Ceph Object Gateway bucket notifications enable users to receive real-time alerts when specific events occur within a bucket. These notifications can be sent to various endpoints, including HTTP, Advanced Message Queuing Protocol (AMQP) 0.9.1, and Kafka. To establish bucket notifications, users must create a notification entry that specifies the bucket, topic, and event types to monitor. Also, users can filter notifications based on key prefix or suffix, regular expression matching, metadata attributes, or object tags. Bucket notifications also offer a REST API for configuration and control.

By sending a notification when an object is synced to a zone, external systems can gain insight into object-level syncing status. The bucket notification event types `s3:ObjectSynced:*` and `s3:ObjectSynced:Created`, when configured through the bucket notification mechanism, trigger a notification event from the synced RADOS Gateway (RGW) on successful object synchronization. Both the topics and the notification configuration should be set up separately in each zone that generates notification events.

For more information, see [Bucket notifications - IBM Documentation](#).



## A real-life Object Storage as a Service case study

This study presents a real-life use case scenario from Türkiye's bank, where they have implemented an Object Storage as a Service (OSaaS) offering by using IBM Storage Ceph in their private cloud computing infrastructure.

This chapter has the following sections:

- ▶ Company profile
- ▶ Empowering enterprise storage: Object Storage as a Service in private cloud environments with IBM Storage Ceph
- ▶ Sample implementation

## 3.1 Company profile

İşbank (<https://www.isbank.com.tr/en>) was the first national bank of Türkiye, and it was established in 1924. It was the first Turkish bank to introduce internet banking, along with the country's first ATM under its brand Bankamatik in 1982. It was also the first Turkish bank to open a branch abroad. İşbank is Türkiye's largest private bank in terms of total assets, serving 20.7 million customers with 22,925 employees. İşbank operates 1,174 branches domestically. Its international network covers 34 branches and 4 representative offices that are present in 11 different countries.

Among the many other awards that the bank has received, Euromoney named İşbank Central and Eastern Europe's Best Digital Bank.<sup>1</sup> The bank was also crowned "Turkish Bank of the Year" at The Financial Times' The Banker Awards.<sup>2</sup>

## 3.2 Empowering enterprise storage: Object Storage as a Service in private cloud environments with IBM Storage Ceph

This section describes object storage concepts, business challenges, and how an OSaaS solution in a private cloud environment can help to address these challenges.

### 3.2.1 Understanding object storage: A paradigm for organizing data

*Object storage* is a data storage architecture that organizes and manages data as distinct, discrete units called *objects*. Each object typically includes the data itself, metadata (descriptive information about the object), and a unique identifier that is often called an *object ID* or *object key*. Unlike traditional file systems or block storage, which organize data hierarchically or in fixed-size blocks, object storage allows data to be stored and retrieved independently.

OSaaS is a delivery model that brings the benefits of cloud-like object storage to an organization's on-premises, private cloud infrastructure. With OSaaS, organizations can leverage the scalability, flexibility, and cost-effectiveness of object storage without having to manage the underlying hardware and software.

### 3.2.2 Business challenges

Persistent storage challenges for enterprises include the following items:

- ▶ Coping with the exponential growth of data
- ▶ Managing diverse data types efficiently
- ▶ Ensuring high availability and disaster recovery (HADR)
- ▶ Maintaining data security and compliance
- ▶ Dealing with the complexity of data management as volumes and performance requirements increase

<sup>1</sup> <https://www.euromoney.com/article/2a3xfv52m4i9d6mrd6j9c/awards/awards-for-excellence/cees-best-digital-bank-2022-isbank>

<sup>2</sup> <https://www.isbank.com.tr/en/about-us/the-banker-crowns-isbank-bank-of-the-year>



In addition to these challenges, enterprises must also deal with these other challenges:

- ▶ Balance the need for cost-effective storage solutions with the need for seamless integration with emerging technologies like cloud computing.
- ▶ Adopt a proactive approach to data management to stay competitive and meet their evolving storage needs effectively.

### 3.2.3 Benefits of the approach

Leveraging an OSaaS model with a strong focus on self-service capabilities in an on-premises private cloud environment enables enterprises to overcome the persistent storage challenges that they face.

Empowering users with self-service tools for Day-2 operations, such as creating users, updating their quotas, and monitoring usage, promotes agility and operational efficiency. This approach reduces administrative bottlenecks and empowers teams to respond swiftly to evolving storage needs, enhancing overall productivity.

A self-service model also ensures cost optimization because organizations can closely align storage resources with actual usage, eliminating waste. Also, with robust monitoring and reporting features, businesses gain valuable insights into storage patterns, enabling data-driven, decision-making, and strategic planning for their needs.

In summary, an OSaaS model with a strong focus on self-service capabilities can provide enterprises with the following benefits:

- ▶ Increased agility and operational efficiency
- ▶ Reduced administrative overhead
- ▶ Cost optimization
- ▶ Improved decision-making

As a result, OSaaS can be a valuable tool for enterprises looking to overcome their persistent storage challenges.

### 3.2.4 Architectural model

In the use case scenario with İşbank, the architectural design is centered on delivering a comprehensive self-service model within the on-premises environment. This model encompasses a range of essential functions, empowering users to efficiently manage their storage resources without requiring administrative assistance.

Users are able to perform critical tasks, such as the following ones:

- ▶ User creation
- ▶ Real-time monitoring of quota usage with proactive alerting mechanisms
- ▶ On-demand quota updates
- ▶ In-depth reporting on both usage and system performance

Also, robust documentation (see “The crucial role of documentation in self-service” on page 60) ensures standardization in processes and consistency throughout the storage management lifecycle.

By integrating these diverse components, the architecture facilitates an agile, user-centric self-service approach to object storage needs, which promotes enhanced data control and operational efficiency within the organization.

In summary, the key benefits of this architectural design include the following ones:

- ▶ **Empowered users:** Users can manage their own storage resources without requiring administrative assistance.
- ▶ **Increased efficiency:** Self-service reduces administrative overhead and enables users to respond more quickly to their storage needs.
- ▶ **Improved data control:** Users have greater visibility into their storage usage and can make informed decisions about their data.
- ▶ **Enhanced operational efficiency:** Standardized processes and consistent documentation help to streamline storage management.

This architectural design can be a valuable solution for organizations that are looking to improve their storage management practices.

### **Creating a user**

Much like the public cloud approach, the process commences with the creation of a user in İşbank. This crucial initial step grants authorized individuals the ability to effortlessly establish user accounts that are tailored to their unique storage demands and preferences.

### **Updating its quota**

Dynamic quota updates empower users to adapt their storage allocations in response to evolving data needs, eliminating the need for time-consuming administrative intervention, which is crucial for accommodating changing storage requirements.

### **Usage and performance reporting**

Reporting usage and performance metrics provides insights into storage operations, enabling İşbank users to access detailed reports about their data consumption and system performance. These insights aid decision-making by helping users better understand their storage patterns and optimize resource allocation.

### **The crucial role of documentation in self-service**

Documentation plays a pivotal role in treating your offerings as a product. It ensures clarity, consistency, and compliance throughout the self-service storage management process. Documentation aims to provide comprehensive guidance, step-by-step instructions, and best practices for users to follow. Well-maintained documentation helps users make informed decisions, promotes standardized practices, and facilitates troubleshooting. It also serves as a valuable resource for training, ensuring that users can fully harness the capabilities of the self-service model.

## 3.3 Sample implementation

The sample structure that is described in the architectural model and how it is implemented in İşbank are summarized in this section.

### 3.3.1 Creating a user

To create a Amazon S3 user, add a catalog item to the company's self-service portal, where the offerings are listed.

Whenever a new user is required, calling the catalog item uses the IBM Storage Ceph API and creates the required user (Figure 3-1).

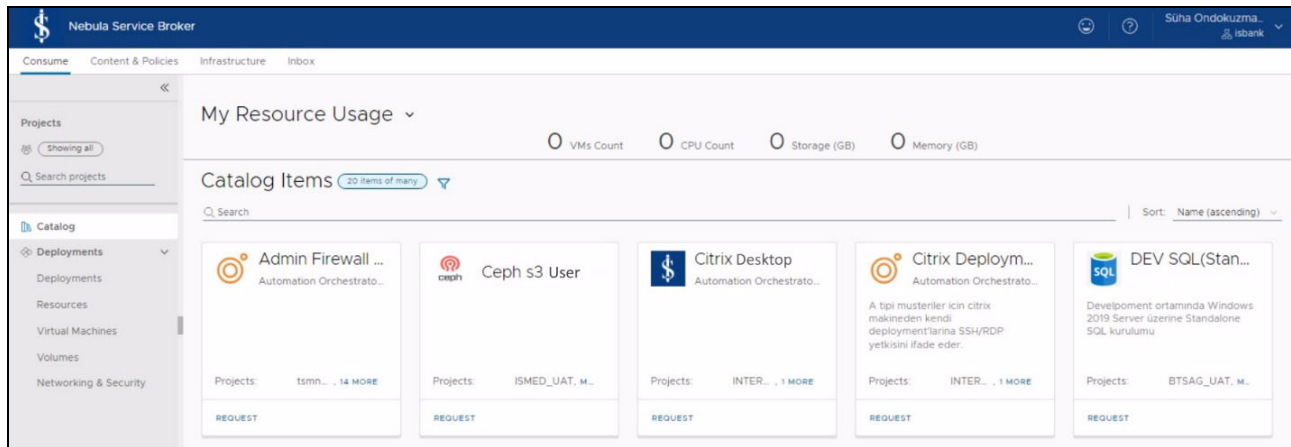


Figure 3-1 Sample portal view with a S3 user catalog item

The user that makes the request can view user-related information on the portal and have the information transmitted through email (Figure 3-2). The sample integration at İşbank was developed by using Python and JavaScript.

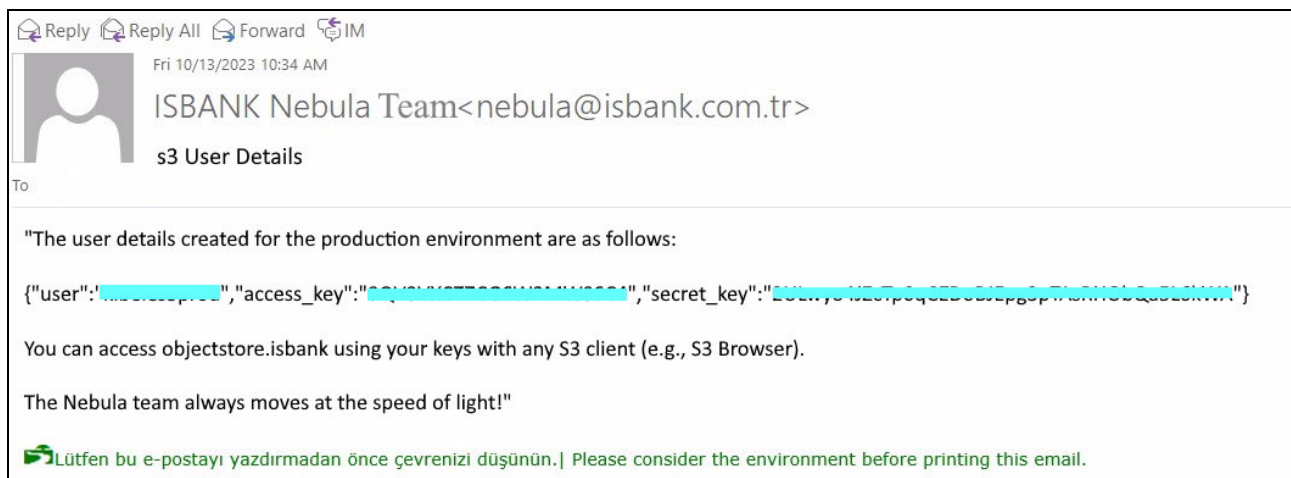


Figure 3-2 The requester can receive an email containing the keys to access the environment

### 3.3.2 Updating the quota

Capacity management is critical for ensuring environmental sustainability. In companies where storage responsibility is often delegated to technology teams, the transition to on-premises private cloud environments can lead to a decrease in awareness of storage responsibility. Reporting on users' quotas and utilization can increase awareness during this transition period.

For example, users can track their quota information, and when they reach a specified threshold, email notifications can be sent to the user or project groups that own the S3 user (Figure 3-3).

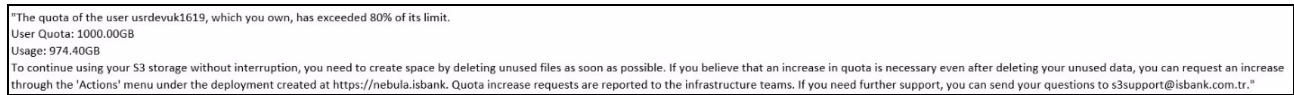


Figure 3-3 Sample alerting by email when a threshold is exceeded

Users who receive this alert can also fulfill their needs by submitting quota increase requests through the S3 user catalog item on the portal (Figure 3-4).

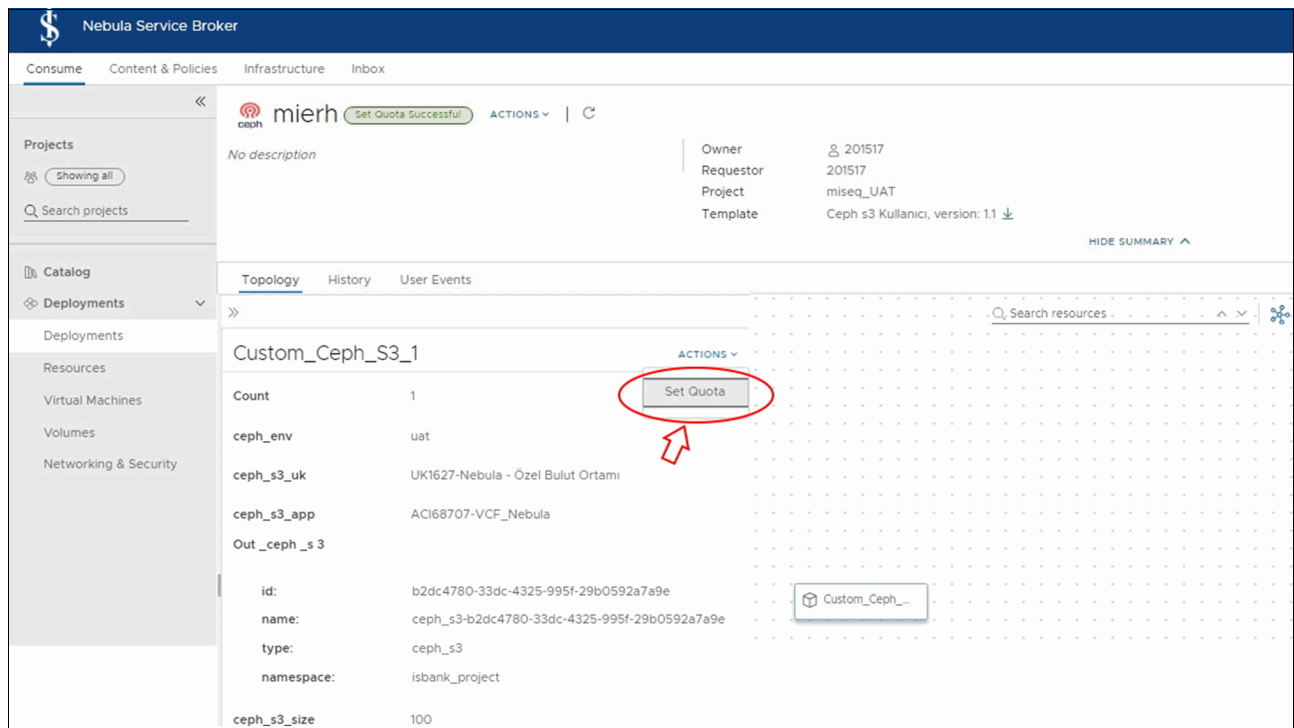


Figure 3-4 A sample view of the catalog item that is used during quota updates

Figure 3-5 on page 63 shows the IBM Storage Ceph API response that indicates that quota increase request completed.

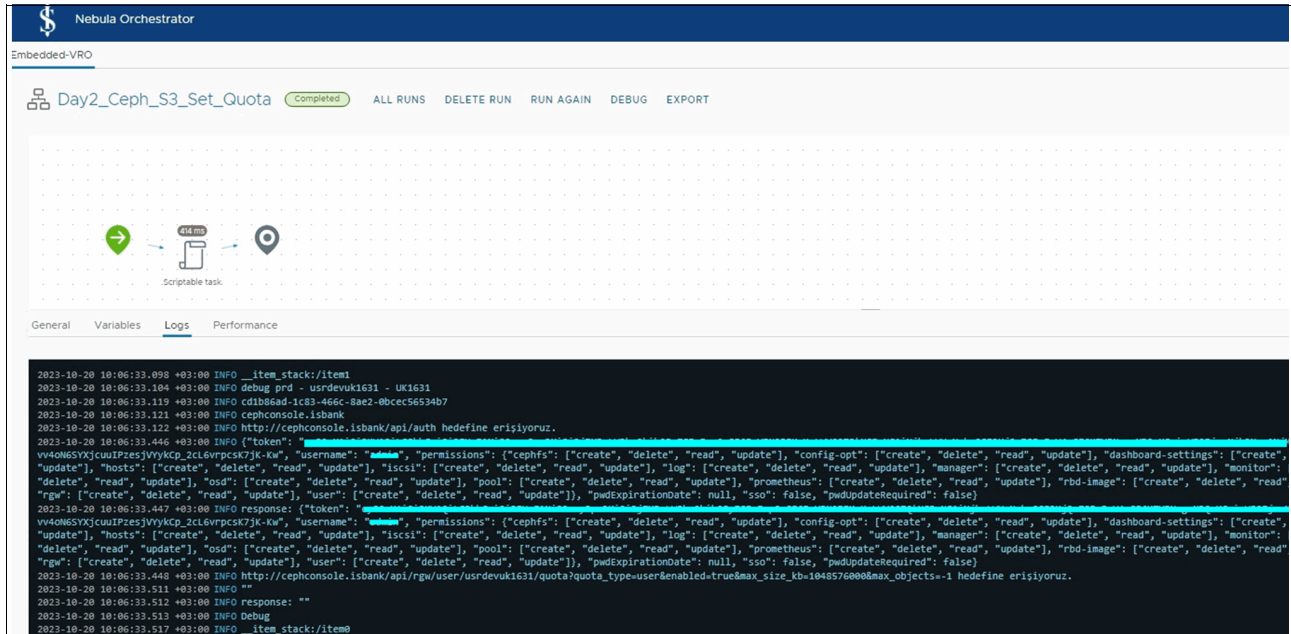


Figure 3-5 IBM Storage Ceph API response

### 3.3.3 Usage and performance reporting

The ability to inspect your usage and examine the associated performance metrics is critical for making informed planning decisions. This information can help you determine whether a performance problem is occurring and help capacity planning.

IBM Storage Ceph provides a Prometheus exporter to collect and export IBM Storage Ceph performance counters from the collection point in ceph-mgr. Enabling the Prometheus module in the mgr service initiates the collection of detailed metrics with the option to use custom scrape intervals (Figure 3-6 and Figure 3-7 on page 64).

**Tip:** The Prometheus Manager module must be restarted after changing its configuration.

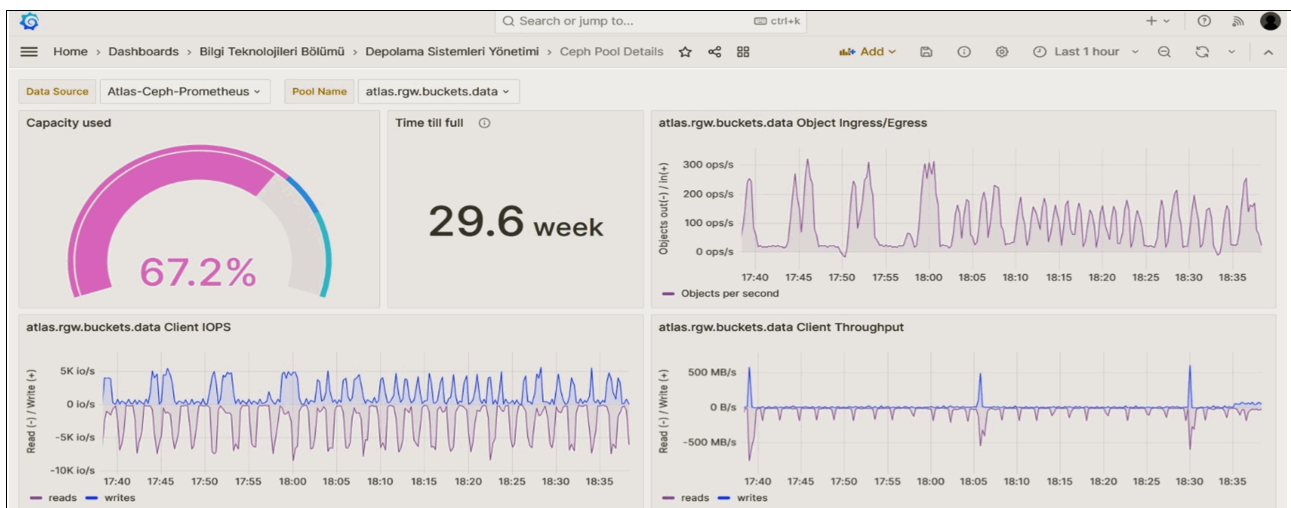


Figure 3-6 A sample Grafana dashboard showing one of the pools usage and its performance

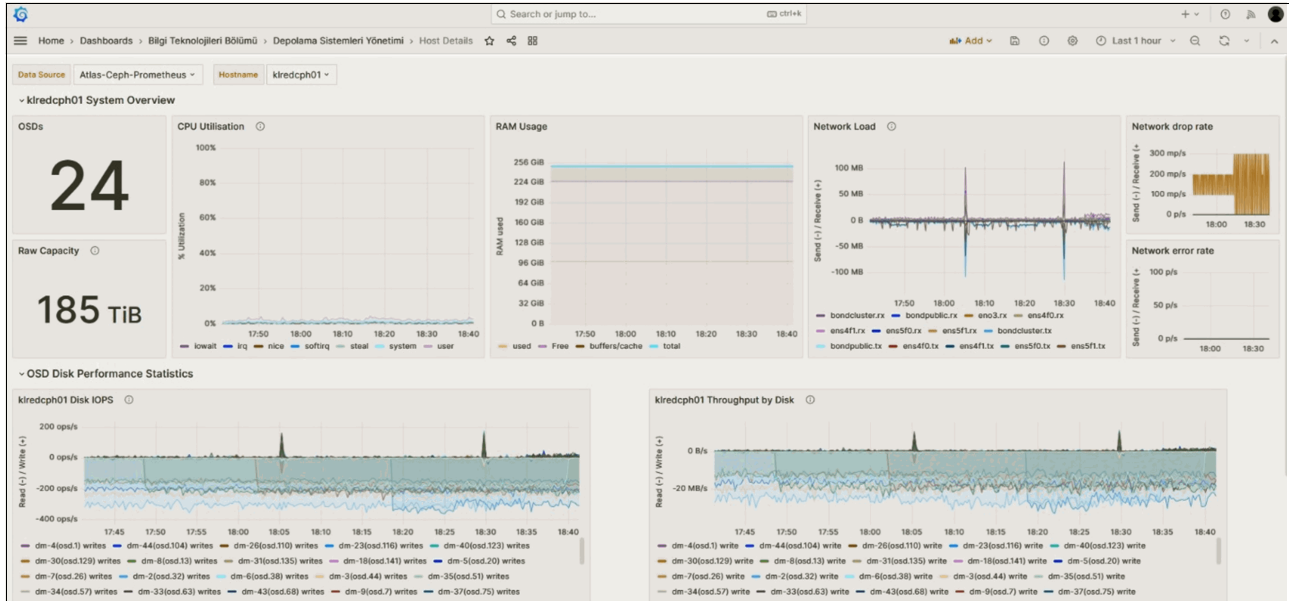


Figure 3-7 A sample Grafana dashboard showing one of the pools usage and its performance

One of the important factors that are related to performance and monitoring is logging. RADOS Gateway (RGW) operation logs can be written to the respective directory on each RGW server by using the globally configured `rgw_ops_log_file_path` setting (Figure 3-8).

```

root@kredcph01:~# ceph config dump | grep "rgw_ops_log_file_path"
global advanced rgw_ops_log_file_path /var/log/ceph/ceph-rgw-ops.json
  
```

Figure 3-8 The global configuration of the log directory for RGW operations

Using the Filebeat agent with a ready integration for Logstash, logs in the relevant directory on the RGW server are directed to Logstash for indexing. Relevant sections that are parsed from the logs can be filtered and viewed through the Kibana interface (Figure 3-9 on page 65).

**Note:** Authorization is required to enable users access to the relevant index pattern containing RGW logs.

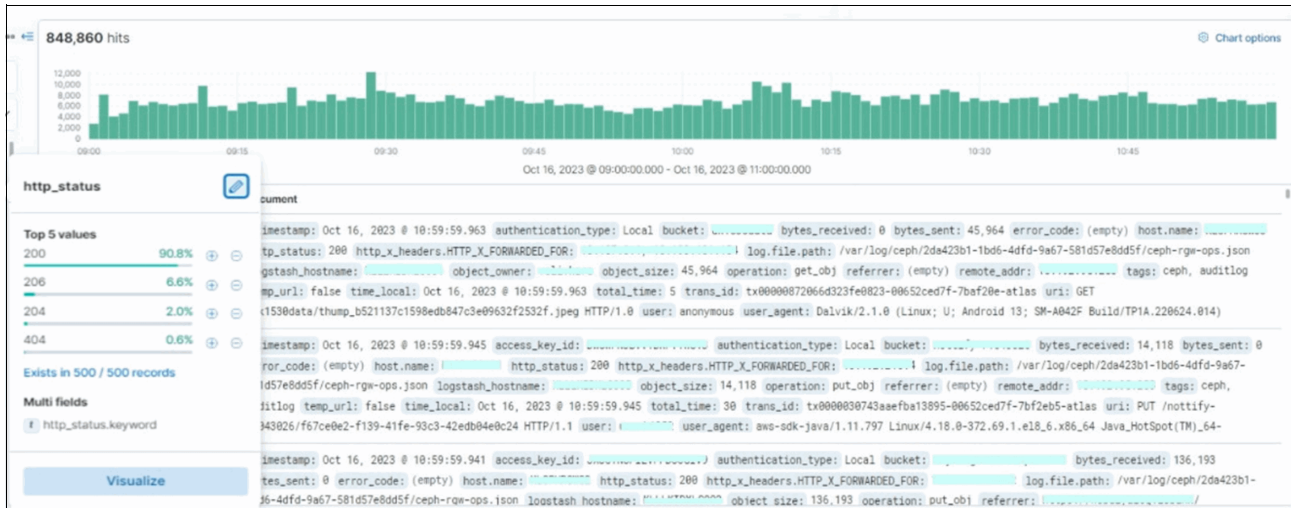


Figure 3-9 Kibana view of the IBM Storage Ceph index pattern

### 3.3.4 Documentation

Maintain an up-to-date documentation link that summarizes your S3 environment's architecture, and provides details about how to use your object storage infrastructure and valuable information such as architectural standards, usage recommendations, and S3 endpoint information. This information will benefit the admin teams that are responsible for managing the infrastructure of object storage.

Figure 3-10 shows a sample documentation view detailing all the required information for the environment.

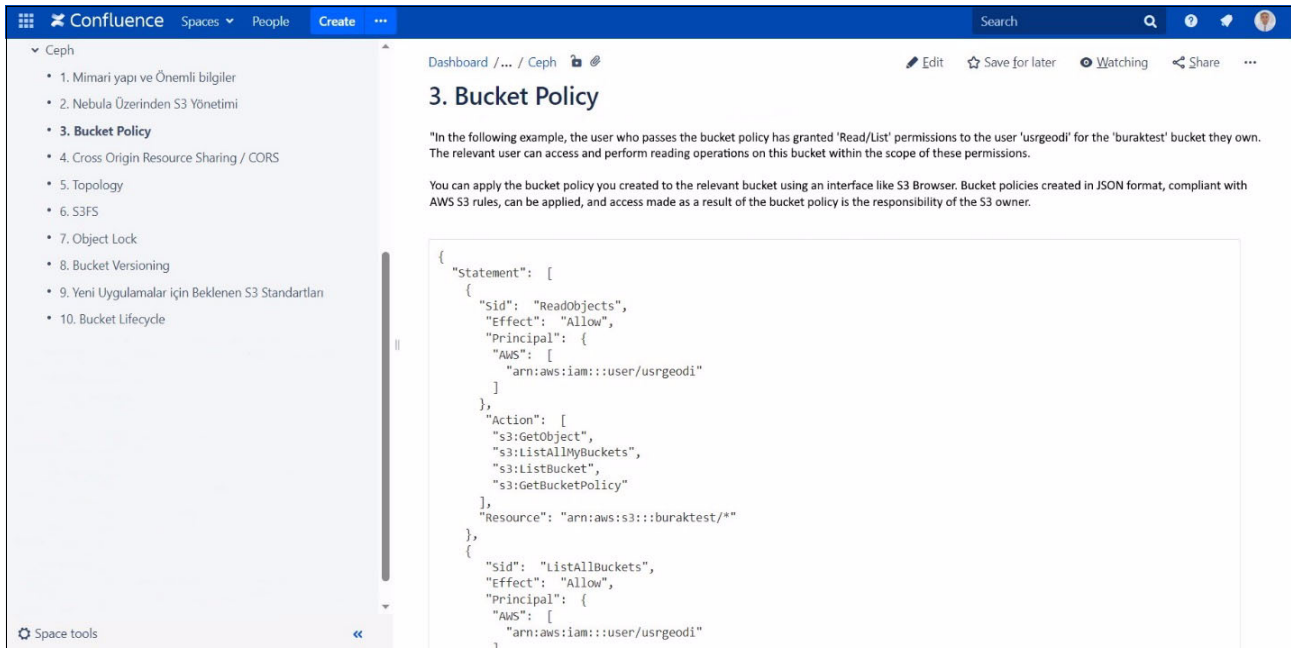


Figure 3-10 Sample documentation view detailing all the required information for the environment







# Disaster recovery and backup and archive: IBM Storage Ceph as an Amazon S3 backup target

This chapter describes using Veritas NetBackup (a backup and recovery product) to demonstrate how it can use IBM Storage Ceph as a backup target.

This chapter has the following sections:

- ▶ Introduction
- ▶ Implementing Veritas NetBackup with IBM Storage Ceph
- ▶ IBM Storage Ceph multi-site replication with Veritas NetBackup
- ▶ IBM Storage Ready Nodes for Ceph

## 4.1 Introduction

Backup and recovery are essential IT operations for data protection and business continuity. As data volumes explode, organizations struggle to find cost-effective and scalable backup solutions that meet stringent recovery time objectives (RTOs). Traditional backup appliances and public cloud options have limitations, making it difficult to achieve both cost-effectiveness and RTO compliance.

To address these challenges, many organizations are turning to on-premises object storage solutions like IBM Storage Ceph to extend cloud models for backup and recovery in a cost-effective way. IBM Storage Ceph is a software-defined data storage solution that can help lower enterprise data storage costs. By leveraging the S3-compatible interface of IBM Storage Ceph, organizations can use it as a scalable and reliable backup target.

### 4.1.1 Benefits of IBM Storage Ceph for Veritas NetBackup

This chapter describes using Veritas NetBackup (a backup and recovery product) to demonstrate how it can use IBM Storage Ceph as a backup target.

Implementing Veritas NetBackup with IBM Storage Ceph offers several advantages for organizations.

#### **Scalability and flexibility**

IBM Storage Ceph is a massively scalable object storage software that enables organizations to easily scale their backup infrastructure to hundreds of petabytes and tens of billions of objects. There is no single point of failure or bottleneck, and you can add inexpensive industry-standard servers as needed for performance and capacity requirements. IBM Storage Ceph is highly available (HA) and resilient, with default configurations that can withstand the loss of multiple nodes without compromising availability.

#### **Data protection and disaster recovery**

By using IBM Storage Ceph as a backup target, organizations can ensure data and application availability while reducing administration costs. IBM Storage Ceph supports multiple data protection methods, including 3-way replication and erasure coding with different profiles, such as 4+2, 8+3, 8+4, and so on. In addition, Ceph Object Gateway supports multi-site active/active replication for disaster recovery (DR).

#### **Efficiency and cost-effectiveness**

Unlike proprietary storage appliances, IBM Storage Ceph leverages industry-standard servers, reducing vendor lock-in and offering cost advantages compared to proprietary solutions. It enables organizations to leverage the competitive storage server market and volume media pricing. Furthermore, IBM Storage Ceph can serve as a unified storage platform, supporting block, object, and file data. IBM Storage Ceph daemons are containerized, providing better utilization of server resources and a decreased hardware footprint, improving the total cost of ownership for small clusters.

## 4.2 Implementing Veritas NetBackup with IBM Storage Ceph

To implement Veritas NetBackup with IBM Storage Ceph, organizations can implement IBM Storage Ceph as a backup object storage.

**Note:** This chapter describes IBM Storage Ceph, but this scenario also applies to Red Hat Ceph.

Configure Veritas NetBackup to use IBM Storage Ceph as the cloud storage provider by using the Amazon S3 API (Figure 4-1). Veritas NetBackup 10.1.1 or later supports IBM Storage Ceph as a backup target through the S3 API. A list of S3 cloud storage vendors that are certified for NetBackup can be found in the [NetBackup Cloud Administrator's Guide](#).

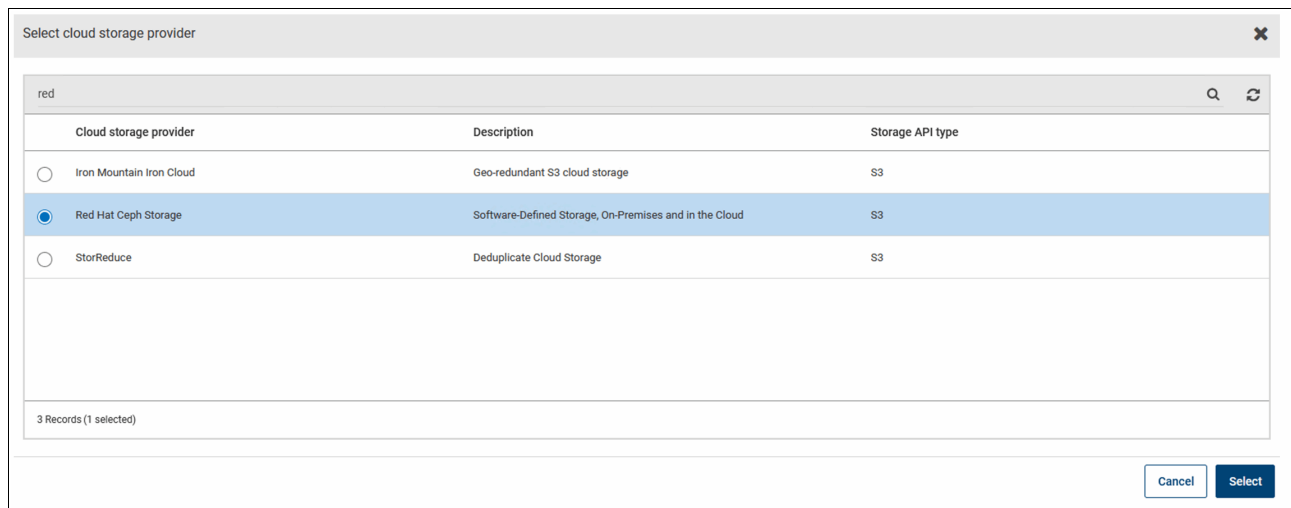


Figure 4-1 Select cloud storage provider window

Figure 4-2 shows the Veritas NetBackup with IBM Storage Ceph implementation for a single site.

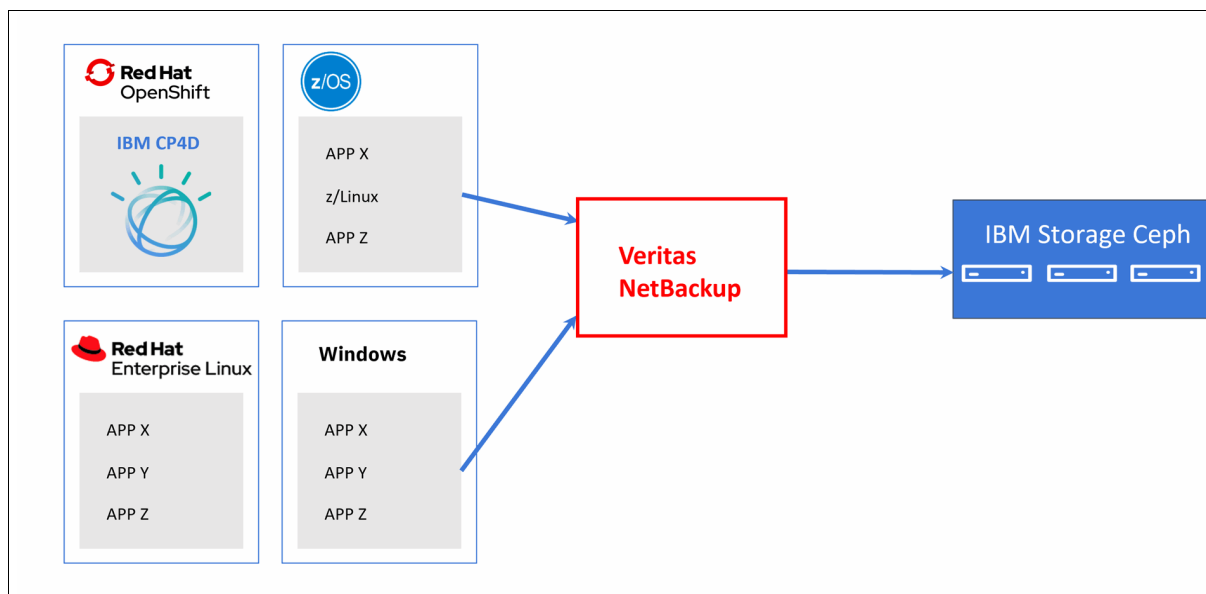


Figure 4-2 Veritas NetBackup with IBM Storage Ceph for a single site

Organizations can also implement IBM Storage Ceph by using multi-site active/active replication for a more resilient solution. In this case, the NetBackup Service URL points to a load balancer, which can fail over access to the DR Ceph cluster if the production Ceph cluster is unavailable. When NetBackup is accessing the DR Ceph cluster, backup and restore operations can continue without interruption. When the IBM Storage Ceph production cluster is back online, the data is synchronized, and the load balancer fails back the access to the IBM Storage Ceph production cluster.

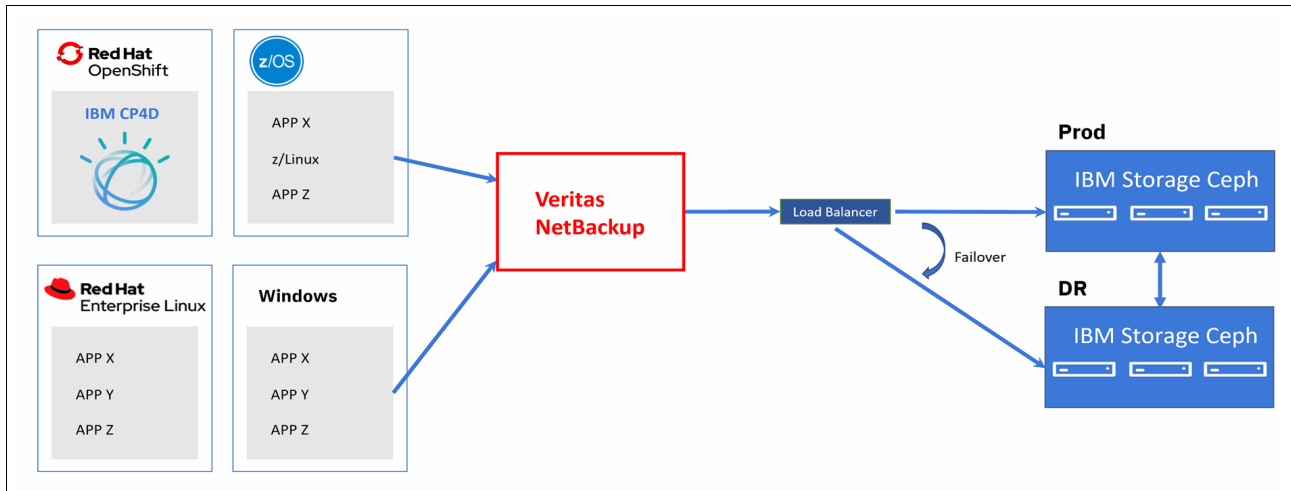


Figure 4-3 IBM Storage Ceph multi-site replication with Veritas NetBackup

### 4.2.1 Certified and supported solution

The combination of Veritas NetBackup and IBM Storage Ceph offers a certified and supported backup solution. Veritas NetBackup is a leading enterprise information archiving and data management solution provider, and IBM Storage Ceph is a scalable and reliable self-healing storage platform for storing enterprise data. The compatibility between these two solutions ensures seamless integration and a robust backup infrastructure.

### 4.2.2 IBM Storage Insights

IBM Storage Insights is an IBM Cloud storage service that allows organizations to monitor the basic health; status; and used, available, and total capacity of IBM Storage Ceph clusters. IBM Storage Insights can also monitor IBM block storage systems and non-IBM block and file storage systems, providing a holistic view of the enterprise infrastructure. IBM Storage Insights is included in IBM Storage Ceph.

## 4.3 IBM Storage Ceph multi-site replication with Veritas NetBackup

This solution is an HA IBM Storage Ceph setup for Veritas NetBackup. Two Ceph clusters are configured with multi-site RADOS Gateway (RGW) replication that is fronted by a HAProxy server for Veritas NetBackup (Figure 4-4 on page 71). For more information about the configuration of Ceph multi-site RGW replication, see [Multi-site configuration and administration](#).

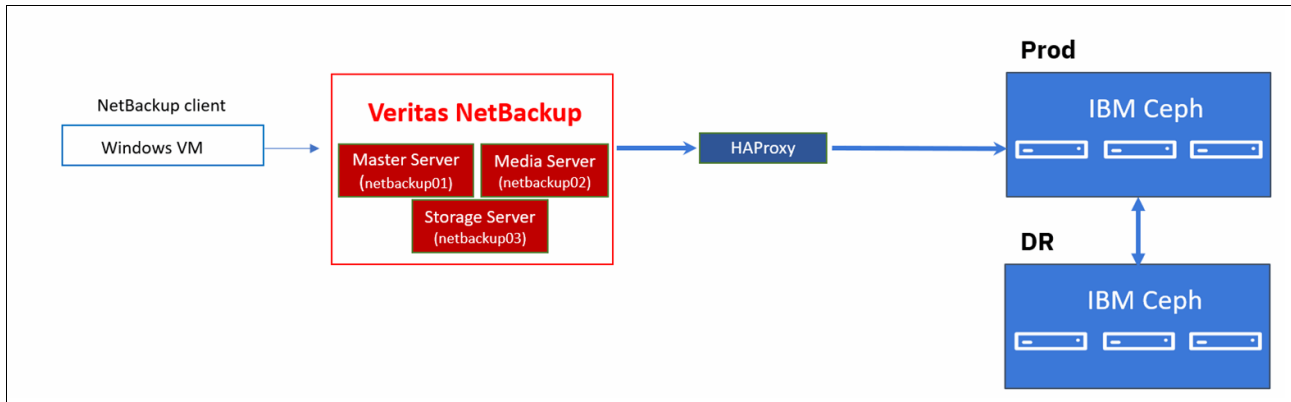


Figure 4-4 Two Ceph clusters are configured with multi-site RGW replication that is fronted by a HAProxy server for Veritas NetBackup

### 4.3.1 Configuring the Veritas NetBackup components

For more information about installing and configuring Veritas NetBackup, see the [NetBackup Installation Guide](#).

Install and set up the following components:

- ▶ Set up netbackup01 as the Master server.
- ▶ Set up netbackup02 as the Media server.
- ▶ Set up netbackup03 as the Storage server.

To configure a storage server for IBM Storage Ceph, complete the following steps:

1. Click **Add storage server** (Figure 4-5).

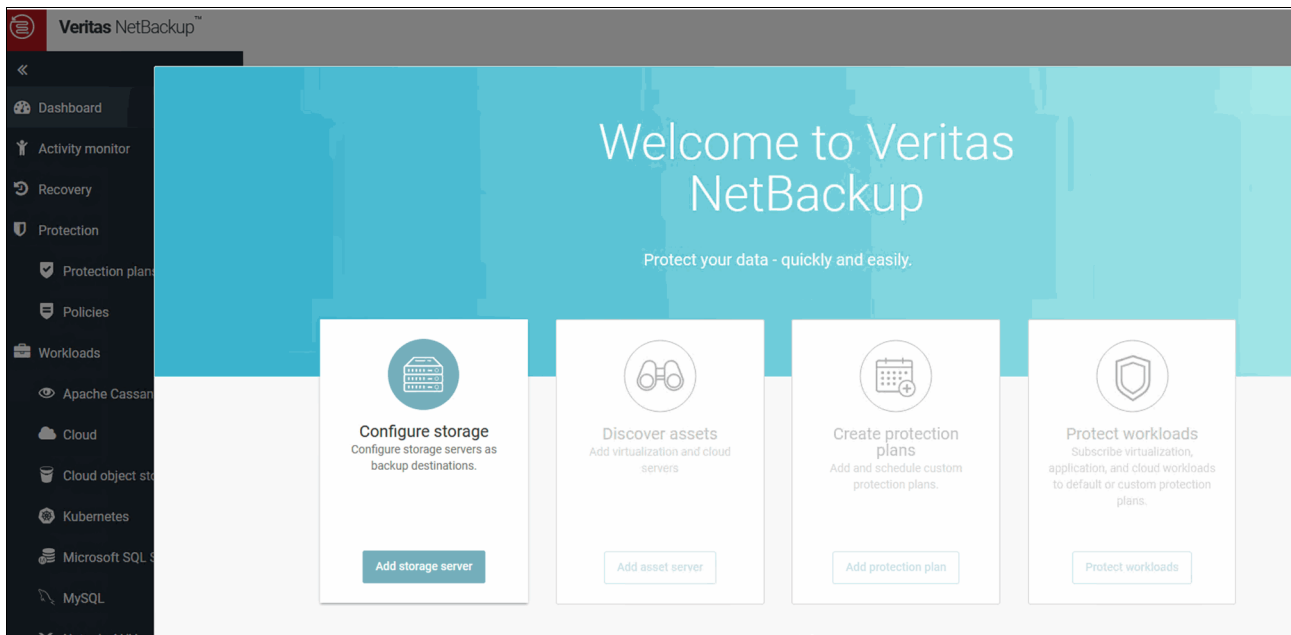


Figure 4-5 Add storage server window

2. Select **Cloud storage** and click **Start** (Figure 4-6).

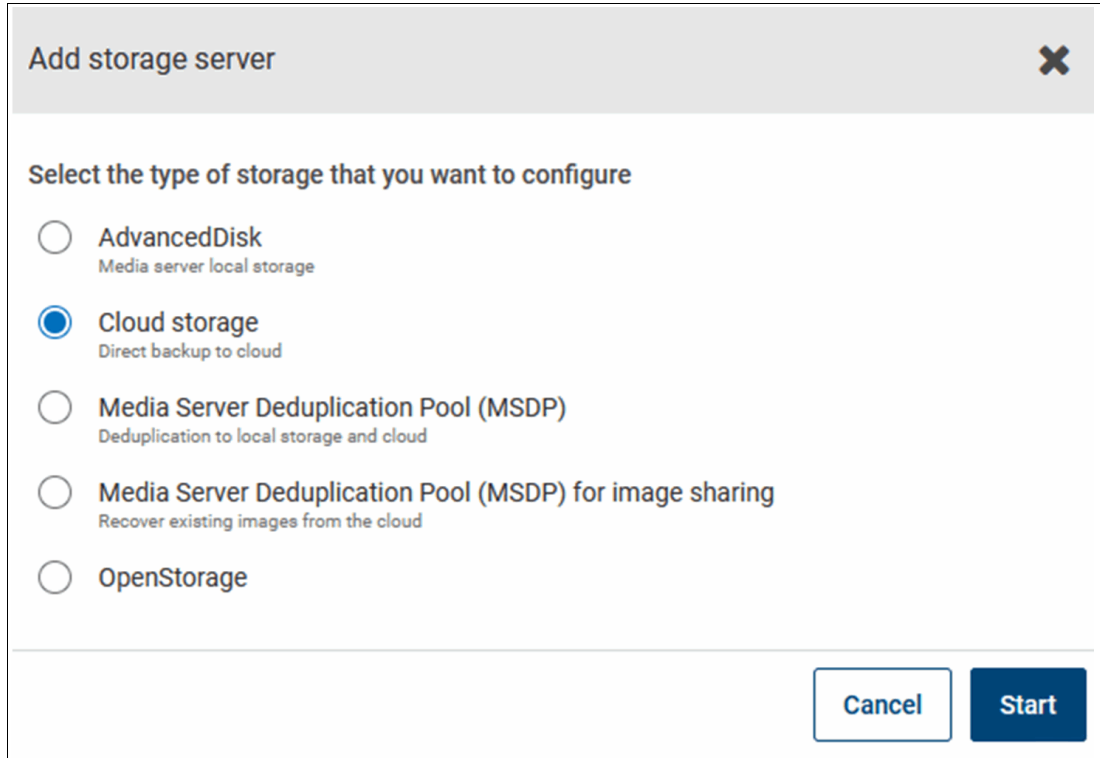


Figure 4-6 Add storage server window

3. Select **Red Hat Ceph Storage** and continue (Figure 4-7).

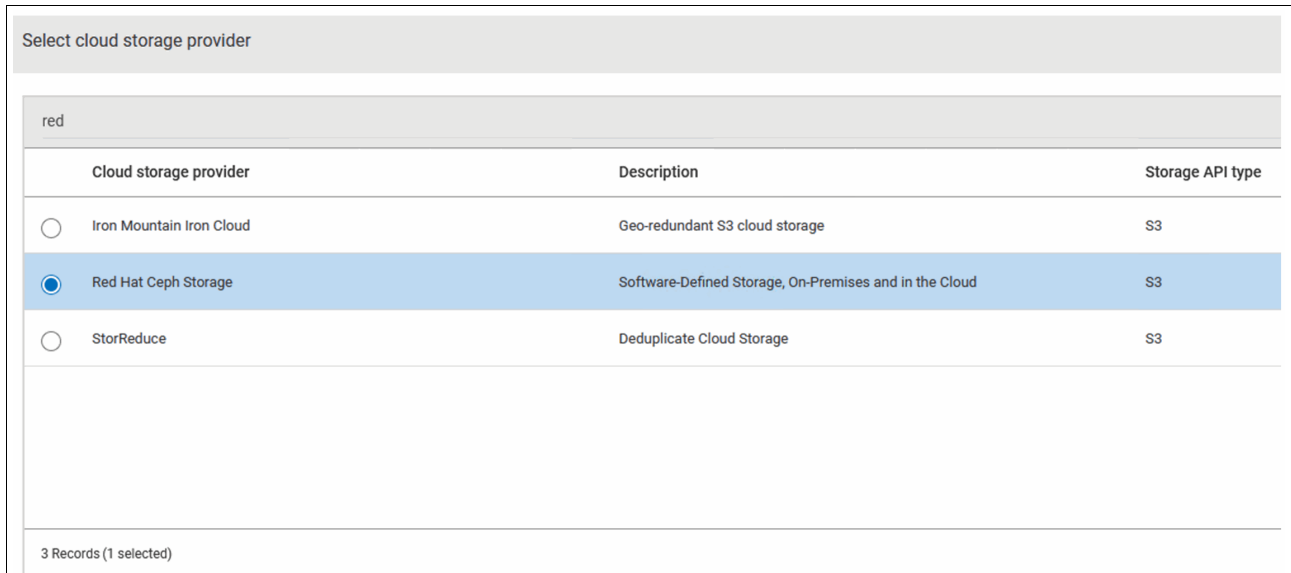


Figure 4-7 Select cloud storage provider window

4. Add a region to Red Hat Ceph Storage. The location constraint is the identifier that cloud providers use to access buckets in the associated region. For public clouds that support AWS V4 signing, you must specify the location constraint.

For on-premises Ceph clusters, both the region name and location constraint should refer to the Ceph Object Gateway zone group `rgw-zonegroup`. The zone group that is used in the Ceph cluster is `multizg`. The service URL is the endpoint for object storage, which in our case is the HAProxy server. Leave the rest of the parameters at their default values. Click **Add** (Figure 4-8).

The screenshot shows a dialog box titled "Add a region" with a close button (X) in the top right corner. The dialog contains the following fields and values:

- Region name \***: multizg
- Location constraint \***: multizg
- Service URL \***: haproxy.testlab.net
- Endpoint access style**: Path style (dropdown menu)
- HTTP port \***: 80 (dropdown menu)
- HTTPS port \***: 443 (dropdown menu)

At the bottom right of the dialog, there are two buttons: "Cancel" and "Add".

Figure 4-8 Add a region window

5. The region is added, as shown in Figure 4-9. Click **Next**.

The screenshot shows the 'Add cloud storage server' dialog box with the following details:

- Storage server name: netbackup03.testlab.net
- Cloud storage provider: Red Hat Ceph Storage
- Storage API type: Amazon S3
- Region: multizg
- Service host: haproxy.testlab.net
- Region name: multizg
- Region identifier: multizg
- Select a media server: netbackup02.testlab.net

Service host	Region name	Region identifier
haproxy.testlab.net	multizg	multizg

Figure 4-9 Adding the region name

6. In Figure 4-10, enter the access key and secret access key and click **Next**. They are the access key and secret access key of the Ceph S3 user.

The screenshot shows the 'Add cloud storage server' dialog box with the following details:

- Access key ID: password
- Secret access key: masked with dots
- Use SSL:
- Use proxy server:

Figure 4-10 Entering the access key and secret access key

7. NetBackup divides the backup data into chunks that are called objects. The performance of NetBackup to S3 storage is determined by the combination of object size, number of parallel connections, and the read/write buffer size. By default, the object size is 32 MB, and compression is enabled. Click **Next** in the next window (Figure 4-11 on page 75).



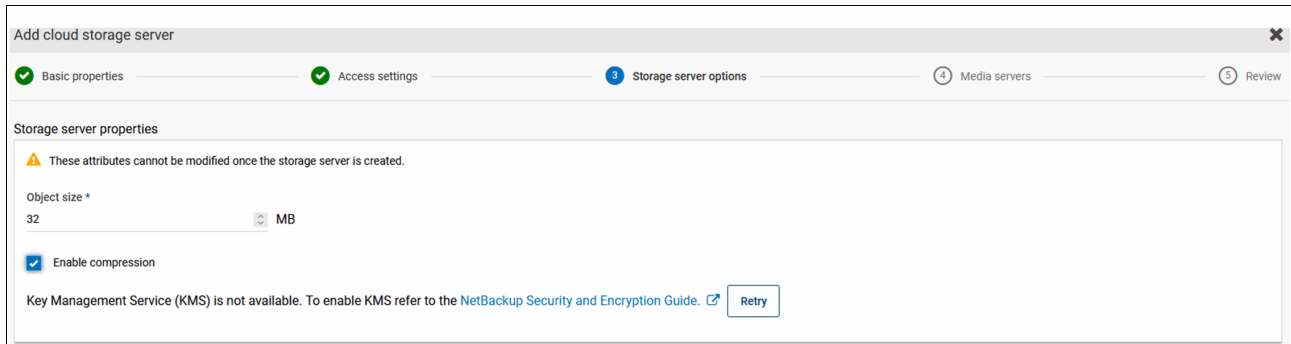


Figure 4-11 Storage server properties

8. Click **Finish** (Figure 4-12).

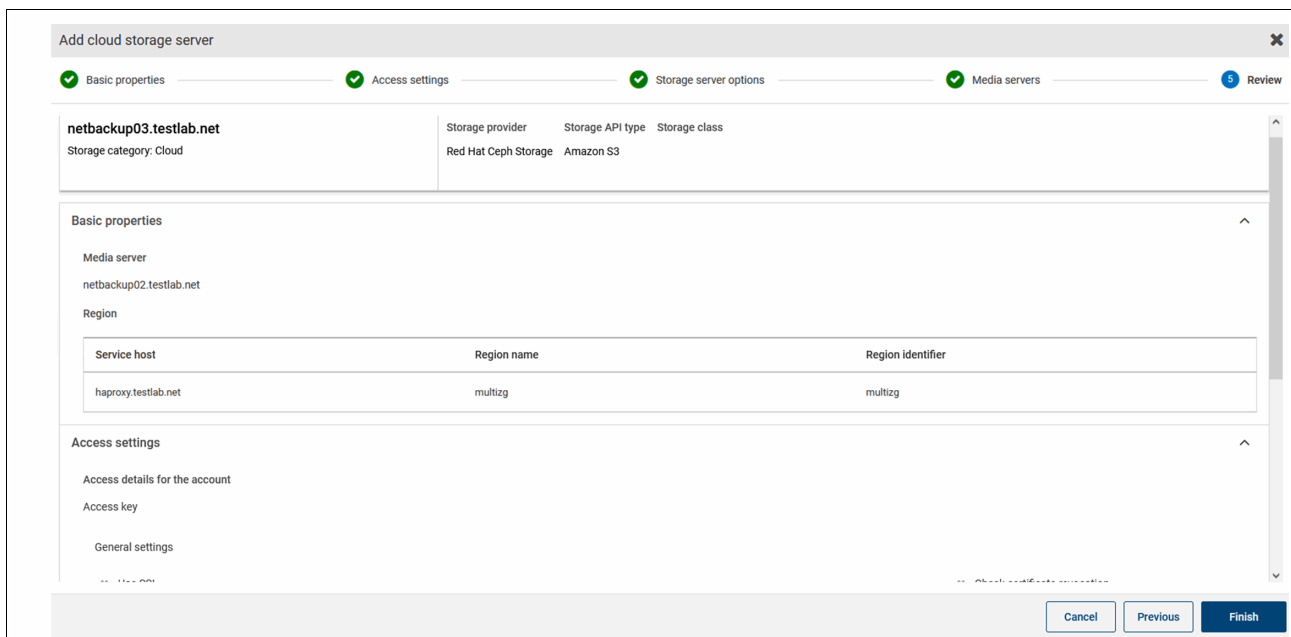


Figure 4-12 Clicking Finish to complete the configuration

## 4.4 IBM Storage Ready Nodes for Ceph

To ease the implementation of IBM Storage Ceph, IBM provides IBM Storage Ready Nodes with configuration options that have been tested and certified for Storage Ceph. Each Ready Node comes with IBM Support and Maintenance, which means that a customer has only a single support line to call for both hardware and software. Ready Nodes for Ceph come with four profiles with different HDD disk sizes: 8 TB, 12 TB, 16 TB, and 20 TB (Figure 4-13).

	Storage Ceph Node
Processor	Intel Xeon Silver 4314
# of Processors	2
RAM	256GB
OS Disks	2x240GB SSD
Data Acceleration Disks	2x3.84TB SSD
Rack Height	2U
Width	482 mm (18.97 in.)
Depth	772.11 mm (30.39 in.)
Height	86.8 mm (3.41 in.)
Weight	35.3kg (77.82 lb) max
# of Capacity Disks	12
HDD Disk Sizes	8TB, 12TB, 16TB, 20TB
Network	
2x1GbE	X
2x10GbE	X




Figure 4-13 IBM Storage Ready Nodes for Ceph

Figure 4-14 on page 77 shows an example hardware configuration of a 1.2 PB IBM Storage Ceph cluster.

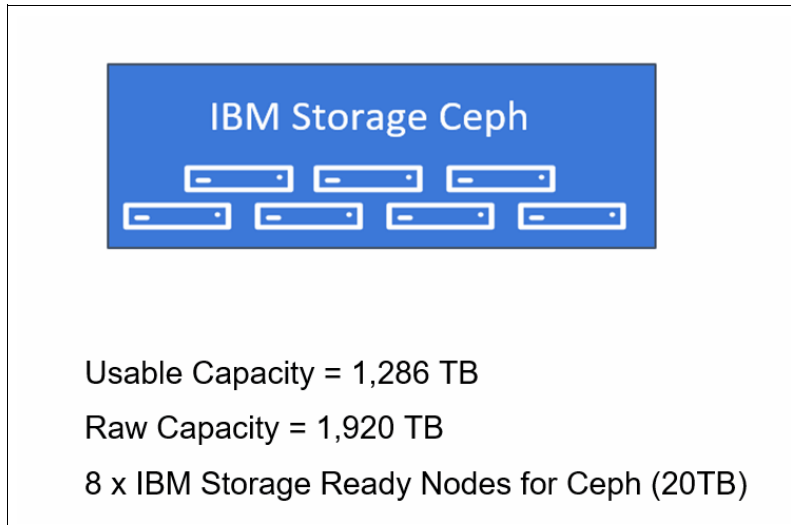


Figure 4-14 Example hardware configuration of a 1.2 PB IBM Storage Ceph cluster

#### 4.4.1 IBM Storage Ceph Editions

IBM Storage Ceph offers two editions. Both editions can be used to implement an IBM Storage Ceph cluster as a backup storage for Veritas NetBackup.

- ▶ IBM Storage Ceph Premium Edition
- ▶ IBM Storage Ceph Pro Edition

The Premium edition includes the Red Hat Enterprise Linux (RHEL) server operating system software license and support.

The Pro edition does not include RHEL server operating system software license, so organizations have the flexibility to bring their own or use their existing RHEL licenses.





# Amazon S3 bucket notifications for event-driven architectures

Objects in S3 are stored in buckets. Buckets can be created by users and can store an unlimited number of objects. This chapter shows some real-world use cases where we use bucket notifications.

This chapter has the following sections:

- ▶ Introduction
- ▶ Cloud-native data pipelines versus legacy data pipelines
- ▶ S3 bucket notifications: Components and workflows
- ▶ Event-driven data pipeline example with the S3 bucket notification feature
- ▶ Configuring event notifications for a bucket in IBM Storage Ceph
- ▶ Configuring S3 bucket notifications in Red Hat OpenShift with S3 RGW object storage by Fusion Data Foundation

## 5.1 Introduction

Objects in S3 are stored in buckets. Buckets can be created by users and can store an unlimited number of objects. When an object is created, deleted, replicated, or otherwise changed in a bucket, you can receive a notification so that you can immediately act. S3 bucket notifications can be useful for many tasks, for example:

- ▶ Automating workflows: For example, you can use S3 bucket notifications to trigger a Lambda function to resize an image file whenever a new image is uploaded to a bucket.
- ▶ Monitoring changes: You can use S3 bucket notifications to monitor changes to your data and send alerts if something unexpected happens.
- ▶ Auditing: You can use S3 bucket notifications to log all changes to your data so that you can track who made what changes and when.

You can configure S3 to generate events for a specific set of actions (such as **GET** or **PUT**) on a per-bucket basis. These events are sent to a configured endpoint, which can then trigger the user-defined steps.

Bucket event notification is a powerful feature for building cloud-native event-driven architectures, especially for data pipelines. It enables you to trigger a data transformation pipeline almost in real time, such as when an object is ingested into an S3 bucket. This approach opens a wide range of possibilities for building your cloud-native data architecture by using data event-driven pipelines. This chapter shows you some real-world use cases where we use bucket notifications.

## 5.2 Cloud-native data pipelines versus legacy data pipelines

This section compares cloud-native data pipelines and legacy data pipelines.

### 5.2.1 Legacy data pipelines

Legacy data pipelines are often tightly coupled, making it difficult to scale on demand to process higher volumes of data. For example, a server ingesting user data might need to mount a Network File System (NFS) or Common Internet File System (CIFS) share from a storage server, and the same shared file system would need to be mounted by a dedicated application server that processes the incoming data in batch mode (Figure 5-1 on page 81).

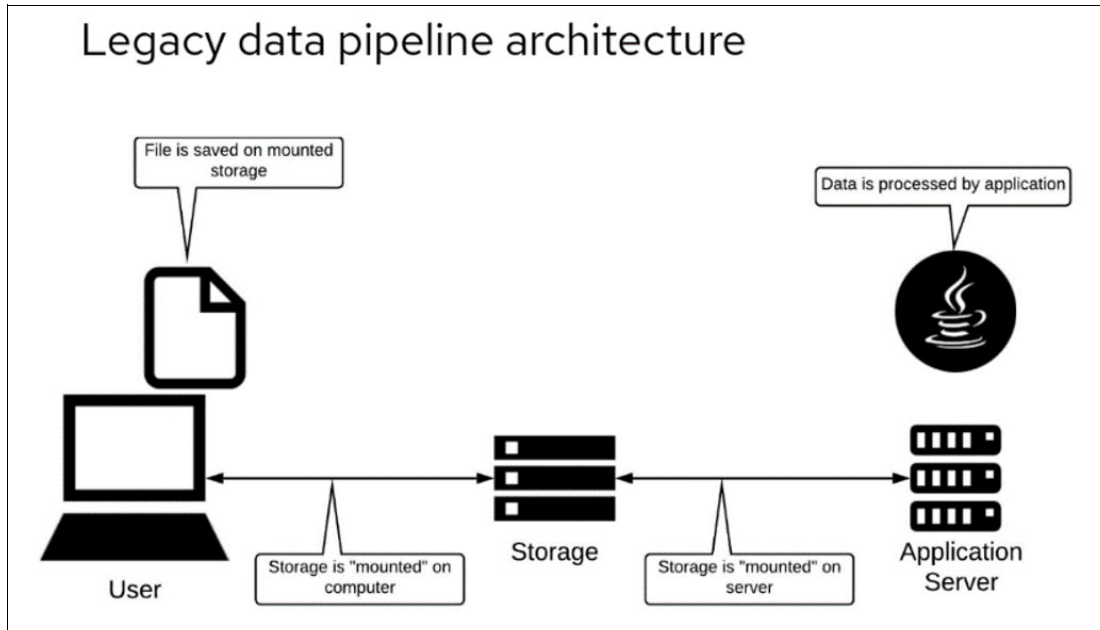


Figure 5-1 Legacy data pipeline architecture pattern

## 5.2.2 Cloud-native data pipelines

Cloud-native architectures decouple pipeline components, making them scalable on demand, sometimes automatically. For example, client data can be ingested into an S3 object store by using simple HTTP/S communication through the S3 API. The S3 HTTP endpoint is accessible from anywhere with network connectivity to port 443.

Then, you can use the bucket notification feature in IBM Storage Ceph to send a message to a Kafka topic whenever a bucket is uploaded or deleted. Then, the topic can be used by a serverless Red Hat OpenShift service like Knative, which can scale from 0 to the number of compute services that is needed to process the ingested data in the S3 object store (Figure 5-2).

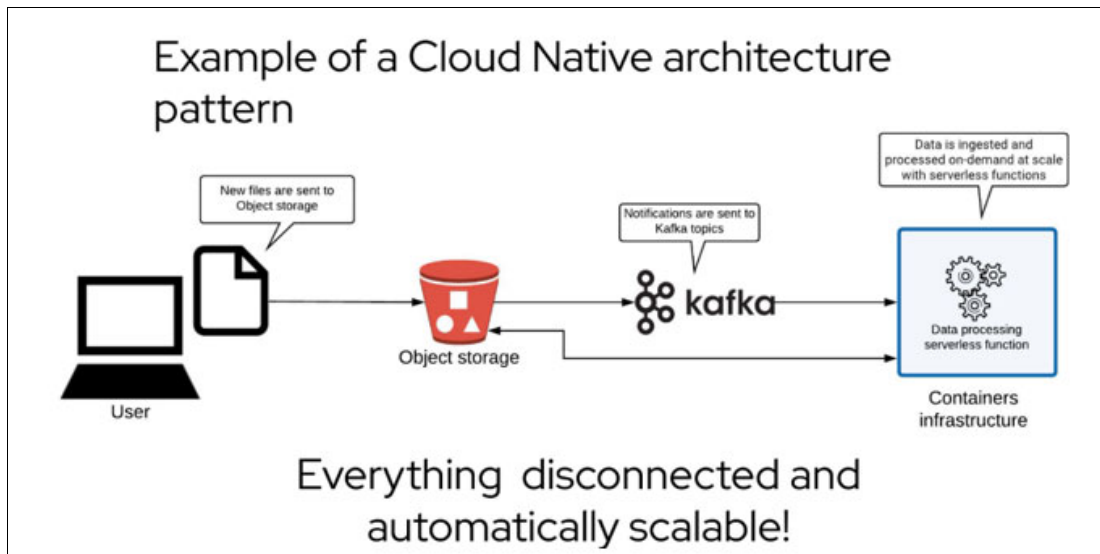


Figure 5-2 Cloud-native architecture pattern

## 5.3 S3 bucket notifications: Components and workflows

This section describes S3 bucket notifications.

### 5.3.1 Bucket notification workflow

Bucket notifications provide a mechanism for sending information out of Ceph through the RADOS Gateway (RGW) when certain events happen on the bucket.

IBM Storage Ceph supports sending events to the following endpoints:

- ▶ Kafka
- ▶ HTTP endpoint
- ▶ Advanced Message Queuing Protocol (AMQP)

As a prerequisite, one of those endpoints must be configured and accessible from the Ceph nodes where the RGW services are running. Then, create a topic in RGW that points to the preconfigured endpoint where you want to send events for each action that happens on the bucket, such as object **PUT**, **DELETE**, and others.

After you configure the S3/RGW topic, define, at the bucket level, which topic the bucket will send notifications to, and which actions will trigger new events to be sent to the topic.

Here is an example of the workflow steps:

1. Create a dedicated topic in Kafka that is called *BucketEvents*. The Kafka endpoint is `kafka.example.com:9092`.
2. Configure a topic in RGW called *KafkaTopic*. Configure the endpoint so that it points to `kafka.example.com:9092`.
3. Configure a notification in the bucket DemoBucket:
  - a. Select the topic to use: `KafkaTopic`
  - b. Select the Actions that trigger an event:
    - `s3:ObjectCreated:*`
    - `s3:ObjectRemoved:*`

Now, you start receiving notifications in the Kafka topic that is called `BucketEvents` when a user puts or deletes an object in the bucket `DemoBucket`.

### 5.3.2 Bucket notification reliability

This section describes synchronous and asynchronous notifications.

#### Synchronous notifications

When the original bucket notification was released, notifications were sent synchronously as part of the operation that triggered them. In this mode, the operation is acknowledged (ACK) only after the notification is sent to the topic's configured endpoint. Therefore, the round-trip time (RTT) of the notification (the time it takes to send the notification to the topic's endpoint plus the time it takes to receive the acknowledgment) is added to the latency of the RGW operation itself.



As you can imagine, this implementation has its drawbacks in certain situations:

- ▶ Even if the message is acknowledged by the endpoint, it is considered successfully delivered and will not be retried.
- ▶ When the messaging endpoint is down, using synchronous notifications slow down your production S3 operations in the RGW because the RGW does not get an acknowledgment until the messaging endpoint times out.
- ▶ Synchronous notifications are not retried, so if an event or message is produced while the endpoint is unavailable or down, it is not sent.

### **Asynchronous notifications**

Persistent bucket notifications were introduced with the IBM Storage Ceph 5.3 Pacific version. The idea behind this new feature was to enable reliable and asynchronous delivery of notifications from the RGW to the endpoint that was configured at the topic.

With persistent notifications, RGW retries sending notifications even if the endpoint is down or a network disconnect occurs during the operation (that is, notifications are retried if not successfully delivered to the endpoint).

## **5.4 Event-driven data pipeline example with the S3 bucket notification feature**

This section describes some examples of how the S3 bucket notification feature is being leveraged in real life to create an event-driven data pipeline.

### **5.4.1 Automated chest X-ray analysis for pneumonia detection**

The Hospital has a new project to automatically analyze chest X-rays and detect possible cases of pneumonia in patients. This use case is a perfect example for using a cloud-native pattern for an automated data pipeline to fulfill the use case requirements.

So, we have X number of hospitals (edge) providing chest X-ray images, some normal and some with pneumonia. The images are sent to a central site, a laboratory where we have an AI/machine learning (ML) model that can be trained to infer the images and determine whether the new images that are being processed at the hospitals are from patients with pneumonia.

All data modeling and training can be done with AI tools, for example, watsonx.ai.

We have an AI/ML model, but we need to ensure that it can scale on demand as the number of hospitals that uses the service increases over time, leading to an increased number of images that must be processed.

We must analyze images as they are uploaded into the system at the hospitals and give the results of the analysis to the doctor. We also want to send all images to the central site so that we can retrain the model for improvements and then seamlessly redeploy the updated model at all the edge sites (hospitals).

We implement an automated data pipeline for chest X-ray analysis. At each hospital (remote/edge site), the following steps occur:

1. Chest X-rays are ingested into an S3 object store that is based on IBM Storage Ceph.
2. The S3 IBM Ceph Object store sends notifications to a Kafka topic.
3. A Knative Eventing KafkaSource Listener triggers a Knative Serving function when it receives a message from a Kafka topic.
4. An ML-trained model running in a pod/container assesses the risk of pneumonia for incoming images.
5. The application notifies the doctor of the results.
6. If the model's certainty in the result is low, anonymize the image data and send it to an S3 object storage that is at the Central Science Lab.

At the main site (Central Science Lab), the following steps occur:

1. The ingested anonymized images go through a human (manual) assessment.
2. Depending on the results of the manual assessment, the images are uploaded to the S3 object storage bucket for pneumonia or to the S3 object storage bucket for normal images.
3. Again, by using S3 bucket notifications, we send an event to the Kafka topic.
4. A Knative Event triggers to retrain the AI/ML model with the new images.
5. When the process finishes, it triggers a continuous integration and continuous deployment workflow to deploy the new updated model to all remote sites/hospitals.

Figure 5-3 shows the chest X-ray architecture diagram.

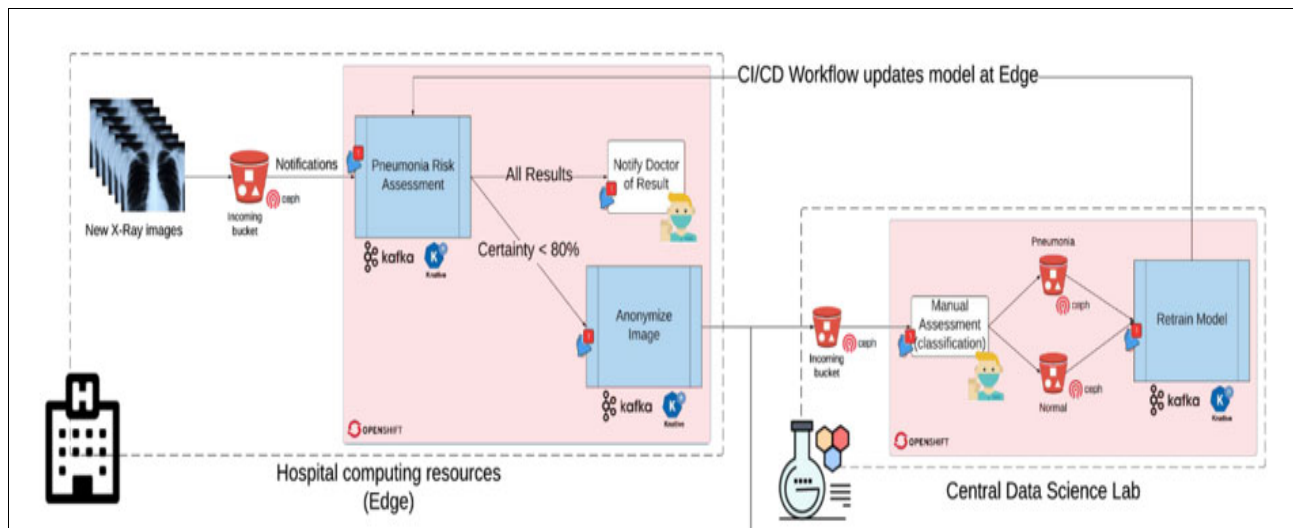


Figure 5-3 Chest X-ray architecture diagram

This pipeline is showcased by Guillaume Moutier from Red Hat in this Red Hat OpenShift Commons YouTube video (slides are available at [GitHub](#)). There is also information about how to demonstrate this event-driven pattern at [About the Medical Diagnosis pattern](#).

## 5.5 Configuring event notifications for a bucket in IBM Storage Ceph

This section describes a step by step basic example of configuring event notifications for a bucket in IBM Storage Ceph.

### 5.5.1 Prerequisites

Prerequisites must be in place before you can proceed:

1. An RGW service must be configured and running. In our example, we have an RGW service running on node `ceph-node02` on port 8080 (Example 5-1).

*Example 5-1 RGW service that is configured and running*

---

```
$ ceph orch ps | grep rgw
rgw.objectgw.ceph-node02.mrhvgq ceph-node02 *:8080      running (23h) 2m ago
23h   68.6M      - 16.2.10-208.e18cp 23e040ac1e4f f0d74eba3452
```

---

2. A running Kafka cluster with an accessible endpoint is required. In this example, we are running Kafka in stand-alone mode on the server `workstation.example.com`, and listening to requests on port 9092 (Example 5-2).

*Example 5-2 Running Kafka in stand-alone mode on server workstation.example.com*

---

```
$ systemctl -a | grep -E '(kafka|zookeeper)' kafka.service
loaded active running Apache Kafka
zookeeper.service
loaded active running zookeeper
```

```
$ ss -nl | grep 9092
tcp LISTEN 0      50
*:9092                *.*
```

---

3. Create a Kafka topic to publish events when an S3 operation occurs on the bucket. In this example, we create a topic that is called `BucketEvents` (Example 5-3).

*Example 5-3 Creating a topic in Kafka*

---

```
$ /opt/kafka/bin/kafka-topics.sh --create --bootstrap-server workstation:9092
--topic bucketevents
Created topic bucketevents.
$ /opt/kafka/bin/kafka-topics.sh --describe --bootstrap-server workstation:9092
--topic bucketevents
Topic: storageTopicId: frr_g8dYRgu17UzENFuubAPartitionCount: 1ReplicationFactor: 1
Configs: segment.bytes=1073741824
Topic: storagePartition: 0Leader: 1Replicas: 1Isr: 1
```

---

4. When the Kafka topic is set up, create a set of S3 credentials (an access key and secret key) through the RGW. Then, use an S3 CLI client, such as the AWS CLI, to create a bucket that is called demobucket (Example 5-4).

**Note:** You can use any other S3 client, We downloaded and installed the AWS CLI client by following the instructions at [Install or update to the latest version of the AWS CLI](#).

*Example 5-4 Creating a bucket*

---

```
$ aws configure
AWS Access Key ID []: S3user1
AWS Secret Access Key []: S3user1key
Default region name [default]: default
Default output format []: json

$ radosgw-admin user create --uid='user1' --display-name='First User'
--access-key='S3user1' --secret-key='S3user1key'

$ aws --endpoint http://ceph-node02:8080 s3 mb s3://demobucket --region default
make_bucket: demobucket
$ aws --endpoint http://ceph-node02:8080 s3 ls
2023-09-27 06:04:27 demobucket
```

---

All the prerequisites are covered. You can start the configuration.

## 5.5.2 Configuring RGW S3 event bucket notifications

To configure RGW S3 event bucket notifications, complete the following steps:

1. Configure the RGW topic. In this topic, we specify our notification endpoint, which in this example is the Kafka server running on `workstation.example.com:9092`. There are several ways to create the required topic in RGW, but in this example, we use the AWS CLI.

To configure our RGW topic, create a JSON file with the topic attributes that you want to set. In this example, we use the following attributes:

- **“push-endpoint”**: The Kafka endpoint to where we want to send the events.
- **“verify-ssl”**: Because we are using an example, we have not configured SSL in Kafka.
- **“kafka-ack-level”**: We wait for the Kafka endpoint ACK before considering the event that is sent.
- **“persistent”**: Setting this option to “true” configures async messaging (available with IBM Storage Ceph 5.3 and later) (Example 5-5 on page 87).

#### Example 5-5 Creating the JSON file

---

```
$ cat << EOF > topic.json
{
  "push-endpoint": "kafka://workstation.example.com:9092",
  "verify-ssl": "False",
  "kafka-ack-level": "broker",
  "persistent": "true"
}
EOF
```

---

2. When the JSON file is created, you can use the AWS CLI to create the RGW topic. The name of the RGW topic matches the name of the Kafka topic that we created previously (bucketevents) (Example 5-6).

#### Example 5-6 Creating the RGW topic

---

```
$ aws --endpoint=http://ceph-node02:8080 sns create-topic --name bucketevents
--attributes=file://topic.json
{
  "TopicArn": "arn:aws:sns:default::bucketevents"
}
$ aws --endpoint=http://ceph-node02:8080 sns list-topics
{
  "Topics": [
    {
      "TopicArn": "arn:aws:sns:default::bucketevents"
    }
  ]
}
```

---

3. Set up the bucket notification. We add a bucket notification to our bucket demobucket. This notification sends an event to the RGW topic bucketevents whenever an object is created or deleted (Example 5-7).

#### Example 5-7 Setting up the bucket notification

---

```
$ cat << EOF > notification.json
{
  "TopicConfigurations": [
    {
      "Id": "kafkanotification",
      "TopicArn": "arn:aws:sns:default::bucketevents",
      "Events": [
        "s3:ObjectCreated:*",
        "s3:ObjectRemoved:*"
      ]
    }
  ]
}
EOF
```

---

In Example 5-7, we set the ID/name of our topic to kafkanotifications. We set TopicArn to the Amazon Resource Name (ARN) of the RGW topic that we created in Example 5-6. Finally, we specified the actions that trigger events, which in this case are object creation and deletion (Example 5-7). To get a list of all supported actions/events, see [S3 Bucket Notifications Compatibility](#).

When the JSON notify file is created, you can apply its configuration to the demobucket (Example 5-8).

*Example 5-8 Applying the JSON file's configuration to the demobucket*

---

```
$ aws --endpoint=http://ceph-node02:8080 s3api
put-bucket-notification-configuration --bucket demobucket
--notification-configuration file://notification.json
$ aws --endpoint http://ceph-node02:8080 s3api
get-bucket-notification-configuration --bucket demobucket
{
  "TopicConfigurations": [
    {
      "Id": "kafkanotification",
      "TopicArn": "arn:aws:sns:default::bucketevents",
      "Events": [
        "s3:ObjectCreated:*",
        "s3:ObjectRemoved:*"
      ]
    }
  ]
}
```

---

Now, if the setup is working as expected, any new object creation or deletion sends a bucket notification to the Kafka topic.

### 5.5.3 Validating the bucket notification configuration

Now that we finished configuring our bucket notification for bucket demobucket, we verify that the events are reaching the Kafka topic when we create or delete an object inside the bucket. Complete the following steps:

1. Configure a Kafka consumer on the topic bucketevents. Use Kafka CLI tools that are provided with the RPM to connect a client and ingest the messages from the bucketevents topic. When you run `kafka-console-consumer`, it runs in the foreground waiting to ingest messages from the topic (Example 5-9).

*Example 5-9 Configuring a Kafka consumer*

---

```
$ /opt/kafka/bin/kafka-console-consumer.sh --bootstrap-server workstation:9092
--topic bucketevents --from-beginning | jq .
```

---

2. Now, create and delete an object in the demobucket to trigger a bucket notification event (Example 5-10).

*Example 5-10 Triggering a bucket notification event*

---

```
$ aws --endpoint http://ceph-node02:8080 s3 cp /etc/hosts s3://demobucket/hosts
upload: ../etc/hosts to s3://demobucket/hosts
```

---

3. If we move back to the terminal where we had our consumer waiting for events, we can now see that the client has ingested the event notification that was created by the creation of an object in the demobucket under the JSON path `.Records.s3`, and that we have information about our bucket and object (Example 5-11 on page 89).

*Example 5-11 The client ingested the event notification that was created by the creation of an object*

---

```
$ /opt/kafka/bin/kafka-console-consumer.sh --bootstrap-server workstation:9092 --topic
kafkatopic --from-beginning | jq .
"Records": [
  {
    "eventVersion": "2.2",
    "eventSource": "ceph:s3",
    "awsRegion": "default",
    "eventTime": "2023-09-27T14:11:27.582138Z",
    "eventName": "ObjectCreated:Put",
    "userIdentity": {
      "principalId": "user1"
    },
    "requestParameters": {
      "sourceIPAddress": ""
    },
    "responseElements": {
      "x-amz-request-id":
"2c814c32-819c-4508-985a-bde6e18e46eb.24154.870028257509586350",
      "x-amz-id-2": "5e5a-default-default"
    },
    "s3": {
      "s3SchemaVersion": "1.0",
      "configurationId": "kafkanotification",
      "bucket": {
        "name": "demobucket",
        "ownerIdentity": {
          "principalId": "user1"
        },
        "arn": "arn:aws:s3:::demobucket",
        "id": "2c814c32-819c-4508-985a-bde6e18e46eb.24172.3"
      },
      "object": {
        "key": "hosts",
        "size": 1330,
        "eTag": "b7828f6b873e43d1bacec3670a9f4510",
        "versionId": "",
        "sequencer": "0F381465861F2223",
        "metadata": [
          {
            "key": "x-amz-content-sha256",
            "val": "1ef29c6abb4b7bc3cc04fb804b1850aecf84dc87493330b66c31bef5209680f3"
          },
          {
            "key": "x-amz-date",
            "val": "20230927T141127Z"
          }
        ]
      },
      "tags": []
    },
    "eventId": "1695823887.589438.b7828f6b873e43d1bacec3670a9f4510",
    "opaqueData": ""
  }
]
```

---

In a real-life situation, you would have an application ingesting events from this topic and taking specific actions when an event is received.

If you have issues with the S3 bucket notification configuration and the events from the RGW are not reaching Kafka, look at the RGW logs. Enable debug mode in the RGW subsystem. For more information about how to enable debug mode for the RGW service, see [How to enable debug logs in RADOS Gateway in Red Hat Ceph Storage?](#)

## 5.6 Configuring S3 bucket notifications in Red Hat OpenShift with S3 RGW object storage by Fusion Data Foundation

You can use S3 bucket notifications inside Red Hat OpenShift with Fusion Data Foundation (FDF). [IBM Storage Fusion](#) includes FDF. The engine that powers the FDF is Ceph. With FDF, you get Ceph deployed inside your Red Hat OpenShift cluster. (There is also the possibility of connecting FDF in external mode to a deployed IBM Storage Ceph stand-alone cluster.)

Deploying Ceph in Red Hat OpenShift provides the advantages of all the automation and lifecycle management that are provided by Red Hat OpenShift Operators. In FDF, the Rook Operator takes care of deploying Ceph. If you deploy on a Red Hat OpenShift on-premises cluster, you get the RGW service, which provides S3 object storage functions.

One of the advantages of working with FDF is its simplicity of use. As an example, we show how to configure S3 bucket notifications in three simple steps with the help of the custom resources (CRs) that are provided by Rook.

### 5.6.1 Prerequisites

Prerequisites must be in place before you can proceed:

1. Ensure that the Red Hat OpenShift cluster is running (Example 5-12).

*Example 5-12 Ensuring that the Red Hat OpenShift cluster is running*

---

```
$ oc get clusterversion
NAME      VERSION  AVAILABLE  PROGRESSING  SINCE   STATUS
version  4.13.13  True       False        4h50m   Cluster version is 4.13.13
```

---

2. Ensure that IBM Storage Fusion with FDF is deployed (Example 5-13).

*Example 5-13 Ensuring that IBM Storage Fusion with Fusion Data Foundation is deployed*

---

```
$ oc get csv -n ibm-spectrum-fusion-ns
NAME                DISPLAY              VERSION  REPLACES              PHASE
isf-operator.v2.6.1  IBM Storage Fusion  2.6.1    isf-operator.v2.6.0   Succeeded
```

```
$ oc get storagecluster -n openshift-storage
NAME                AGE    PHASE  EXTERNAL  CREATED AT              VERSION
ocs-storagecluster  4h39m  Ready  External  2023-10-02T10:36:03Z   4.13.2
```

```
$ oc get cephcluster -n openshift-storage
NAME                DATADIRHOSTPATH  MONCOUNT  AGE    PHASE
MESSAGE            HEALTH           EXTERNAL  FSID
ocs-storagecluster-cephcluster  /var/lib/rook    3         4h40m  Ready
Cluster created successfully HEALTH_OK
684c0d97-8fea-471d-84fb-b78c91a1e769
```

---



3. Configure an RGW `cephobjectstores` CR. This CR deploys the RGW service, which provides access to the Ceph Storage S3 RGW endpoint. Also, create an Red Hat OpenShift service and an external route if S3 connectivity is required from outside the Red Hat OpenShift cluster.

Here is an improved version of the text. We are deploying the service without SSL for simplicity. For production deployments, HTTPS/SSL is a best practice. The RGW service is deployed with FDF for on-premises deployments. If you are running on a cloud deployment, you can get instructions about how to set up the `cephobjectstores` CR at [Enable use of the RGW on an OCS internal deployment](#) (Example 5-14).

*Example 5-14 Configuring a RADOS Gateway*

---

```
$ oc get cephobjectstores.ceph.rook.io
NAME                                PHASE
ocs-storagecluster-cephobjectstore Ready
$ oc get svc rook-ceph-rgw-ocs-storagecluster-cephobjectstore
NAME                                TYPE          CLUSTER-IP
EXTERNAL-IP  PORT(S)      AGE
rook-ceph-rgw-ocs-storagecluster-cephobjectstore ClusterIP     172.30.66.167
<none>       80/TCP      143m
```

---

4. Create the Strimzi Kafka Operator (Example 5-15).

*Example 5-15 Creating the Strimzi Kafka Operator*

---

```
$ oc get kafka
NAME          DESIRED KAFKA REPLICAS  DESIRED ZK REPLICAS  READY  WARNINGS
kafka-clus   2                       1                     True
```

---

5. Create a Strimzi Kafka topic that is called `bucketevents` (Example 5-16).

*Example 5-16 Creating a Strimzi Kafka topic*

---

```
$ oc get kafkatopic bucketevents
NAME          CLUSTER  PARTITIONS  REPLICATION FACTOR  READY
bucketevents  kafka-clus  4           2                    True
```

---

## 5.6.2 Configuring RGW S3 event bucket notifications with FDF

The same concepts that we covered in 5.5.2, “Configuring RGW S3 event bucket notifications” on page 86 for IBM Storage Ceph stand-alone also apply here, with the difference that we leverage the CRs that are available in the Rook Operator.

### Notifications

A `CephBucketNotification` defines what bucket actions trigger the notification and which topic to send notifications to. A `CephBucketNotification` may also define a filter, based on the object's name and other object attributes. Notifications can be associated with buckets that are created by using `ObjectBucketClaims` by adding labels to an `ObjectBucketClaim`.

`CephBucketTopic`, `CephBucketNotification`, and `ObjectBucketClaim` must all belong to the same namespace. If a bucket was created manually (not by using `ObjectBucketClaim`), notifications on this bucket should also be created manually. However, topics in these notifications may reference topics that were created by using `CephBucketTopic` resources.

## Topics

CephBucketTopic represents an endpoint (of types Kafka, AMQP 0.9.1, or HTTP), or a specific resource inside this endpoint (such as a Kafka or an AMQP topic, or a specific URI in an HTTP Server). CephBucketTopic also holds any more information that is needed for a CephObjectStore RGW to connect to the endpoint. Topics do not belong to a specific bucket or notification. Notifications from multiple buckets may be sent to the same topic, and one bucket (by using multiple CephBucketNotifications) may send notifications to multiple topics.

To set up a topic, complete the following steps:

1. Configure the topic by using the CephBucketTopic CR that is provided by Rook. In this example, the topic is named `bucketevents`. Most of the parameters are explained in 5.5, “Configuring event notifications for a bucket in IBM Storage Ceph” on page 85. For more information about the parameters are used in this example, see [Object Bucket Notifications](#).

For the URI for Kafka, we use the service FQDN because we access the Kafka endpoint from a different namespace from where Kafka is running. In this example, Kafka is running in the namespace `data-pipeline`, and our topic is configured in the `openshift-storage` namespace (see Example 5-17).

### *Example 5-17 Accessing the Kafka endpoint*

---

```
$ oc project openshift-storage
$ cat << EOF > topic_crd.yaml
apiVersion: ceph.rook.io/v1
kind: CephBucketTopic
metadata:
  name: bucketevents
  namespace: openshift-storage
spec:
  objectStoreName: ocs-storagecluster-cephobjectstore
  objectStoreNamespace: openshift-storage
  opaqueData: my@email.com
  persistent: true
  endpoint:
    kafka:
      uri: kafka://kafka-clus-kafka-brokers.data-pipeline.svc:9092
      disableVerifySSL: true
      ackLevel: broker
      useSSL: false
EOF
$ oc create -f topic_crd.yaml
cephbuckettopics.ceph.rook.io/bucketevents created
$ oc get cephbuckettopics.ceph.rook.io/bucketevents
NAME          PHASE
bucketevents  Ready
```

---

2. Create the bucket notification by using the CR CephBucketNotification that is provided by Rook. In the specs, we specify the name of the RGW topic, where the events are sent when triggered, and also the events that we want to send to the topic. In this example, only the S3 **PUT** and **COPY** actions trigger an event notification to Kafka (see Example 5-18).

### *Example 5-18 Creating the bucket notification*

---

```
$ cat << EOF > notification_crd.yaml
apiVersion: ceph.rook.io/v1
kind: CephBucketNotification
```

```

metadata:
  name: bucket-notification
  namespace: openshift-storage
spec:
  topic: bucketevents
  events:
    - s3:ObjectCreated:Put
    - s3:ObjectCreated:Copy
EOF
$ oc create -f notification_crd.yaml
cephbucketnotifications.ceph.rook.io/bucket-notification created
$ oc get cephbucketnotifications.ceph.rook.io/bucket-notification
NAME                                AGE
bucket-notification                 29m

```

---

- Using the RGW storage class, create an Object Bucket Claim (OBC). The OBC creates a bucket and user credentials in RGW for us. By using bucket notification labels, OBC also configures the bucket notification, In our case, we use the bucket notification bucket-notification (see Example 5-19).

*Example 5-19 Creating an Object Bucket Claim*

---

```

$ oc get sc | grep rgw
ocs-storagecluster-ceph-rgw  openshift-storage.ceph.rook.io/bucket  Delete
Immediate                    false                               15h
$ cat << EOF > bucket-obc.yaml
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: ceph-noti-bucket
  labels:
    bucket-notification-bucket-notification: bucket-notification
spec:
  generateBucketName: databucket2
  storageClassName: ocs-storagecluster-ceph-rgw
EOF
$ oc create -f bucket-obc.yaml
ObjectBucketClaim/ceph-noti-bucket created
$ oc get obc
NAME                                STORAGE-CLASS          PHASE    AGE
ceph-noti-bucket                    ocs-storagecluster-ceph-rgw  Bound    12h

```

---

### 5.6.3 Validating the bucket notification configuration

To validate the bucket notification configuration, complete the following steps:

1. Verify that the OBC notify labels work as expected and that the notification is configured in the new bucket. Use the AWS CLI to extract the credentials (access and secret keys) of the OBC from a secret with the same name (see Example 5-20).

*Example 5-20 Extracting the credentials*

---

```
$ oc extract secret/ceph-noti-bucket --to=-
# AWS_ACCESS_KEY_ID
GATKJ4RQMOXS7R3LHG1L
# AWS_SECRET_ACCESS_KEY
FxLV39RLABvL2GRweTIHTPYgS5IUkL6aQF1RbpA3
```

---

2. Add the credentials to our AWS CLI credentials file `$HOME/.aws/credentials` with a profile name of `notify` (see Example 5-21).

*Example 5-21 Adding the credentials*

---

```
$ cat << EOF >> .aws/credentials
[notify]
aws_access_key_id = GATKJ4RQMOXS7R3LHG1L
aws_secret_access_key = FxLV39RLABvL2GRweTIHTPYgS5IUkL6aQF1RbpA3
EOF
```

---

3. We are using the AWS CLI client from inside Red Hat OpenShift Container Platform, so we use the Red Hat OpenShift Container Platform service FQDN as the RGW S3 endpoint. We can get the Red Hat OpenShift Container Platform service FQDN from the OBC configmap with the same name (see Example 5-22).

*Example 5-22 Getting the Red Hat OpenShift Container Platform service FQDN from the OBC configmap with the same name*

---

```
$ oc get cm/ceph-noti-bucket -o json | jq .data
"BUCKET_HOST":
"rook-ceph-rgw-ocs-storagecluster-cephobjectstore.openshift-storage.svc",
"BUCKET_NAME": "databucket2-f9c6479b-3a22-45b4-9564-a9507b4d5d93",
"BUCKET_PORT": "80",
"BUCKET_REGION": "",
"BUCKET_SUBREGION": ""
```

---

4. Now that the AWS CLI is configured, run the `get-bucket-notification s3api` command to check whether the configuration is in place for the bucket `databucket2` (see Example 5-23).

*Example 5-23 Checking whether the configuration is in place for the bucket databucket2*

---

```
$ aws --profile notify
--endpoint=http://rook-ceph-rgw-ocs-storagecluster-cephobjectstore.openshift-storage.svc s3api get-bucket-notification-configuration --bucket
databucket2-f9c6479b-3a22-45b4-9564-a9507b4d5d93 | jq .
"TopicConfigurations": [
  {
    "Id": "bucket-notification",
    "TopicArn": "arn:aws:sns:ocs-storagecluster-cephobjectstore::bucketevents",
    "Events": [
```

```

        "s3:ObjectCreated:Put",
        "s3:ObjectCreated:Copy"
    ]
}
]

```

5. As a final verification, upload something to the bucket and check whether it triggers the event and sends it to the Kafka topic bucketevents (see Example 5-24).

*Example 5-24 Uploading something to the bucket and checking whether it triggers the event*

```

$ aws --profile notify
--endpoint=http://rook-ceph-rgw-ocs-storagecluster-cephobjectstore.openshift-storage.svc s3 cp /etc/resolv.conf
s3://databucket2-f9c6479b-3a22-45b4-9564-a9507b4d5d93
upload: ../../etc/resolv.conf to
s3://databucket2-f9c6479b-3a22-45b4-9564-a9507b4d5d93/resolv.conf

```

6. To double-check that the configuration is functional and that the events are reaching their destination topic bucketevents, use the Kafka consumer that is available in the Kafka pod (see Example 5-25).

*Example 5-25 Double-checking that our configuration is functional and that the events are reaching their destination topic*

```

$ oc exec -n data-pipeline -it kafka-clus-kafka-0 -- bash
$ /opt/kafka/bin/kafka-console-consumer.sh --bootstrap-server
kafka-clus-kafka-brokers:9092 --topic bucketevents --from-beginning
{"Records":[{"eventVersion":"2.2","eventSource":"ceph:s3","awsRegion":"ocs-storage
cluster-cephobjectstore","eventTime":"2023-10-03T07:00:19.271850Z","eventName":"Ob
jectCreated:Put","userIdentity":{"principalId":"obc-openshift-storage-ceph-noti-bu
cket-684c0d97-8fea-471d-84fb-b78c91a1e769"},"requestParameters":{"sourceIPAddress"
:""},"responseElements":{"x-amz-request-id":"479fd891-23fe-4435-8006-31a4be9cc7ff.
74097.11704580900520310930","x-amz-id-2":"12171-ocs-storagecluster-cephobjectstore
-ocs-storagecluster-cephobjectstore"},"s3":{"s3SchemaVersion":"1.0","configuration
Id":"bucket-notification2","bucket":{"name":"databucket2-056d7d8b-b626-4ad6-99fe-b
a5e147561bd","ownerIdentity":{"principalId":"obc-openshift-storage-ceph-noti-bucke
t2-684c0d97-8fea-471d-84fb-b78c91a1e769"},"arn":"arn:aws:s3:ocs-storagecluster-cep
hobjectstore:databucket2-056d7d8b-b626-4ad6-99fe-ba5e147561bd","id":"479fd891-23f
e-4435-8006-31a4be9cc7ff.75784.1"},"object":{"key":"resolv.conf","size":920,"eTag"
:"698a41f5f188c0d3a9e4298fde4631f4","versionId":"","sequencer":"03BC1B65C32C3712",
"metadata":[{"key":"x-amz-content-sha256","val":"ebdf560272a77357195c39e98340b77e1
8c8a8ce2025ee950e9e0c7b01467ab8"}, {"key":"x-amz-date","val":"20231003T070018Z"}]},
"tags":[]},"eventId":"1696316419.305605.698a41f5f188c0d3a9e4298fde4631f4","opaqueD
ata":"my@email.com"]}]

```

With this final verification, we conclude that the configuration is working and that we can configure S3 bucket notifications in the Red Hat OpenShift application buckets.





# IBM Storage Fusion disaster recovery

This chapter covers IBM Storage Fusion disaster recovery (DR) configurations.

This chapter has the following sections:

- ▶ Introducing Fusion Data Foundation
- ▶ IBM Storage Ceph stretch mode
- ▶ Fusion Metro-DR for RPO Zero

## 6.1 Introducing Fusion Data Foundation

*IBM Storage Fusion Data Foundation (FDF)* is a highly integrated collection of cloud storage and data services for Red Hat OpenShift Container Platform. It is available as part of the Red Hat OpenShift Container Platform service catalog, and it is packaged as an operator to facilitate simple deployment and management.

FDF services are primarily made available to applications in the form of storage classes that represent the following components:

- ▶ Block storage devices catering primarily to database (DB) workloads, such as Red Hat OpenShift Container Platform logging and monitoring, and PostgreSQL.
- ▶ A shared and distributed file system catering primarily to software development, messaging, and data aggregation workloads, such as Jenkins build sources and artifacts, Wordpress uploaded content, Red Hat OpenShift Container Platform registry, and messaging by using JBoss AMQ.
- ▶ Multi-cloud object storage featuring a lightweight S3 API endpoint that can abstract the storage and retrieval of data from multiple cloud object stores.
- ▶ On-premises object storage featuring a robust S3 API endpoint that scales to tens of petabytes and billions of objects and primarily targeting data-intensive applications, such as the storage and access of row, columnar, and semi-structured data with applications like Spark, Presto, Red Hat AMQ Streams (Kafka), and even machine learning frameworks like TensorFlow and PyTorch.

FDF integrates a collection of software projects:

- ▶ *Ceph* to provide block storage, a shared and distributed file system, and on-premises object storage.
- ▶ *Ceph CSI* to manage provisioning and the lifecycle of persistent volumes and claims.
- ▶ *NooBaa* to provide a Multicloud Object Gateway (MCG).
- ▶ *FDF*, *rook-ceph*, and *NooBaa* operators to initialize and manage FDF services.

**Note:** Throughout this document, IBM Storage FDF and Red Hat OpenShift Data Foundation are used interchangeably because they have the same functions.

### 6.1.1 Introducing Fusion disaster recovery

DR is the process of restoring an organization's essential applications and services after a disruption. It is a crucial part of any business continuity plan, which aims to ensure that business operations can continue even during major adverse events.



The FDF DR capability allows DR across multiple Red Hat OpenShift Container Platform clusters. It is categorized into three types:

- ▶ Metro-DR

Metro-DR ensures business continuity during data center unavailability without data loss. In the public cloud, it is similar to protecting against an availability zone failure.

- ▶ DR with stretch cluster

The stretch cluster solution ensures business continuity with no-data loss DR protection with FDF synchronous replication in a single Red Hat OpenShift cluster. It is stretched across two data centers with low latency and one arbiter node.

- ▶ Regional-DR (technology preview)

Regional-DR ensures business continuity during the unavailability of a geographical region by accepting some loss of data in a predictable amount. In the public cloud, it is similar to protecting from a regional failure. This solution is based on asynchronous volume-level replication for block and file volumes.

Zone failure in Metro-DR and region failure in Regional-DR are expressed by using the terms recovery point objective (RPO) and recovery time objective (RTO):

- ▶ RPO

RPO measures how often you back up or snapshot persistent data. In practice, it represents the maximum amount of data that can be lost or must be reentered after an outage.

- ▶ RTO

RTO is the maximum amount of time a business can be down without incurring unacceptable losses. It answers the question, "How long can your system take to recover after you are notified of a business disruption?"

DR strategies can be grouped into two main categories:

- ▶ Active/passive

In an active/passive DR strategy, all traffic is routed to one data center while the other is in standby mode. In a disaster, traffic might be down for a short period as operations are switched to the standby data center. This strategy is best suited for situations where only two data centers are available.

- ▶ Active/active

In an active-active DR strategy, the workload is spread across all available data centers. If a data center is lost due to a disaster, services can continue to operate without interruption.

Table 6-1 summarizes the different offerings and corresponding use cases.

Table 6-1 Table reflecting the use case per solution

Offering	Definition	Use case
Regional-DR	<ul style="list-style-type: none"> <li>▶ A DR solution across two Red Hat OpenShift clusters in two different geographical locations that are connected by a WAN network.</li> <li>▶ Based on asynchronous volume level replication for block and file volumes.</li> </ul>	Sustaining the loss of a data center or a geographical region.
Metro-DR	<ul style="list-style-type: none"> <li>▶ No-data loss DR protection with FDF and Red Hat OpenShift Data Foundation based synchronous replication between two Red Hat OpenShift clusters in two data centers with low latency.</li> <li>▶ Based on an IBM Storage Ceph stretched cluster.</li> </ul>	<ul style="list-style-type: none"> <li>▶ Sustaining the loss of a cluster in a campus or Metro DR scenario.</li> <li>▶ Follows the best practice DR principles with fault-isolated local clusters.</li> </ul>
DR with stretch cluster	No-data loss DR protection with FDF and Red Hat OpenShift Data Foundation based Synchronous replication in a single Red Hat OpenShift cluster, which is stretched across two data centers with low latency and arbiter node.	<ul style="list-style-type: none"> <li>▶ Sustaining component failures in a campus or Metro deployment.</li> <li>▶ Benefits from quick recovery by using high availability and disaster recovery (HADR) principles.</li> </ul>

## 6.1.2 Regional-DR overview

*Regional-DR* is a multi-cluster configuration that spans regions and data centers. It uses asynchronous data replication, which can result in some data loss, but it protects against a wide range of failures.

The Regional-DR solution contains Red Hat Advanced Cluster Management for Kubernetes (RHACM) and Red Hat OpenShift Data Foundation or FDF components to provide application and data mobility across Red Hat OpenShift Container Platform clusters.

Here are the details of the Regional-DR offering:

- ▶ Two Red Hat OpenShift Container Platform and FDF internal clusters.
- ▶ FDF and Red Hat OpenShift Data Foundation are not supported externally with Regional-DR.
- ▶ Red Hat Advanced Cluster Manager is required (Version 2.7 or later).
- ▶ Supported Platforms: Public cloud and on-premises, which are supported by FDF.
- ▶ Data types: Block and file.
- ▶ Latency: No limits.
- ▶ Network: WAN.
- ▶ Arbiter: Not required.
- ▶ Number of zones or sites: 2.

- ▶ RPO: Less than 5 minutes (best practice).
- ▶ RTO: Between 5 and 15 minutes (depends on the application start-up time).

Figure 6-1 shows the IBM FDF Regional-DR offering.

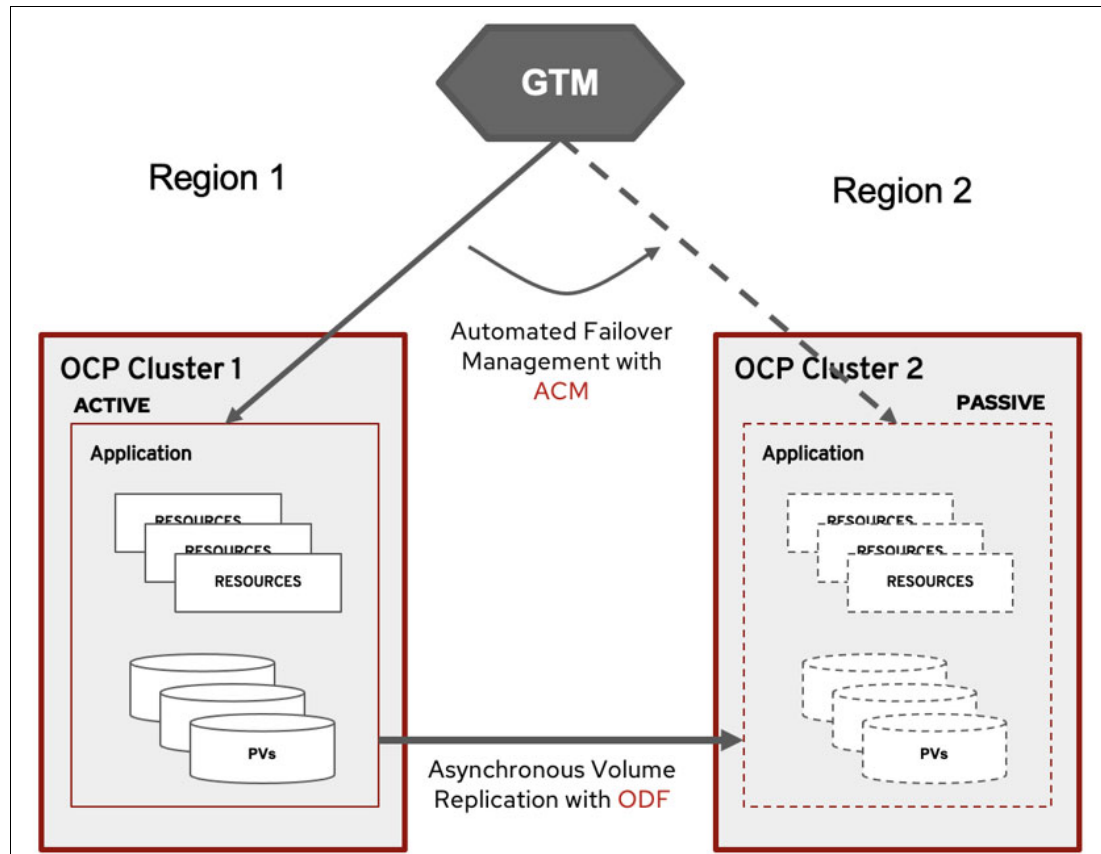


Figure 6-1 IBM Fusion Data Foundation Regional-DR offering

### 6.1.3 Metro-DR overview

Metro-DR ensures business continuity during a data center outage with no data loss (synchronous replication). In the public cloud, it is similar to protecting against an availability zone failure.

Here are the details of the Metro-DR offering:

- ▶ Two Red Hat OpenShift Container Platform and FDF clusters are deployed in separate data centers that connect to an external IBM Storage Ceph cluster.
- ▶ A single IBM Storage Ceph stretch cluster across three zones with the third zone having only one MON running as a tie-breaker. A single IBM Storage Ceph stretch cluster serves both FDF clusters.
- ▶ Red Hat Advanced Cluster Manager is required (Version 2.7 or later).
- ▶ Supported platforms: VMware and bare metal only.
- ▶ Data types: Block and file.
- ▶ Latency: <10 ms round-trip time (RTT) for data nodes, and <100 ms RTT for an arbiter/MON node.
- ▶ Network: High bandwidth (campus network).

- ▶ Arbiter: A third zone or site is required for the arbiter, with <100 ms RTT latency required.
- ▶ Number of zones or sites: 3, with the third zone running with arbiter/MON only (can be a virtual machine (VM)).
- ▶ RPO: 0.
- ▶ RTO: Between 5 and 15 minutes (depends on the application start-up time).

Figure 6-2 shows the IBM FDF Metro-DR offering.

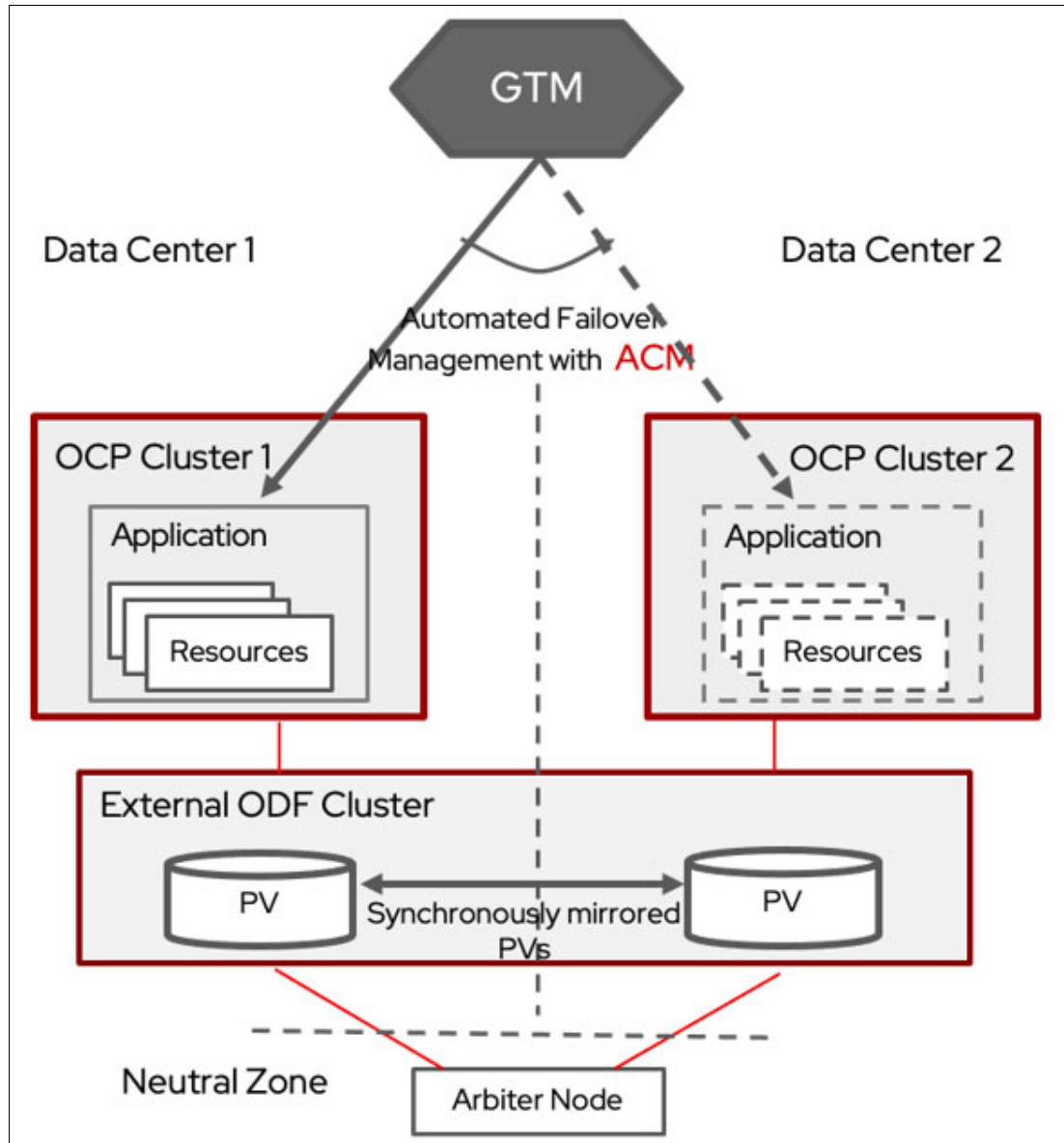


Figure 6-2 IBM Fusion Data Foundation Metro-DR offering

#### 6.1.4 Red Hat OpenShift DR with a stretched cluster

A stretch cluster solution ensures business continuity with no-data loss DR protection with FDF synchronous replication in a single Red Hat OpenShift cluster, which is stretched across two data centers with low latency and one arbiter node.

Here are the details of the Stretched Cluster Red Hat OpenShift DR offering:

- ▶ One Red Hat OpenShift and one FDF internal cluster that is stretched across 2 or 3 data centers.
- ▶ FDF is deployed internally. No external IBM Storage Ceph is required.
- ▶ No Advanced Cluster Manager is required.
- ▶ Supported platforms: VMware and bare metal only.
- ▶ Data types: Block, file, and object.
- ▶ Latency: <10 ms RTT for all nodes.
- ▶ Network: High bandwidth (campus network).
- ▶ A third zone for Ceph and Red Hat OpenShift arbiter is required.
- ▶ Three zones, with the third zone having a control plane and MON running on an arbiter node.
- ▶ RPO: 0.
- ▶ RTO: < 1 min (depending on the application).

Figure 6-3 shows the IBM FDF stretched mode offering.

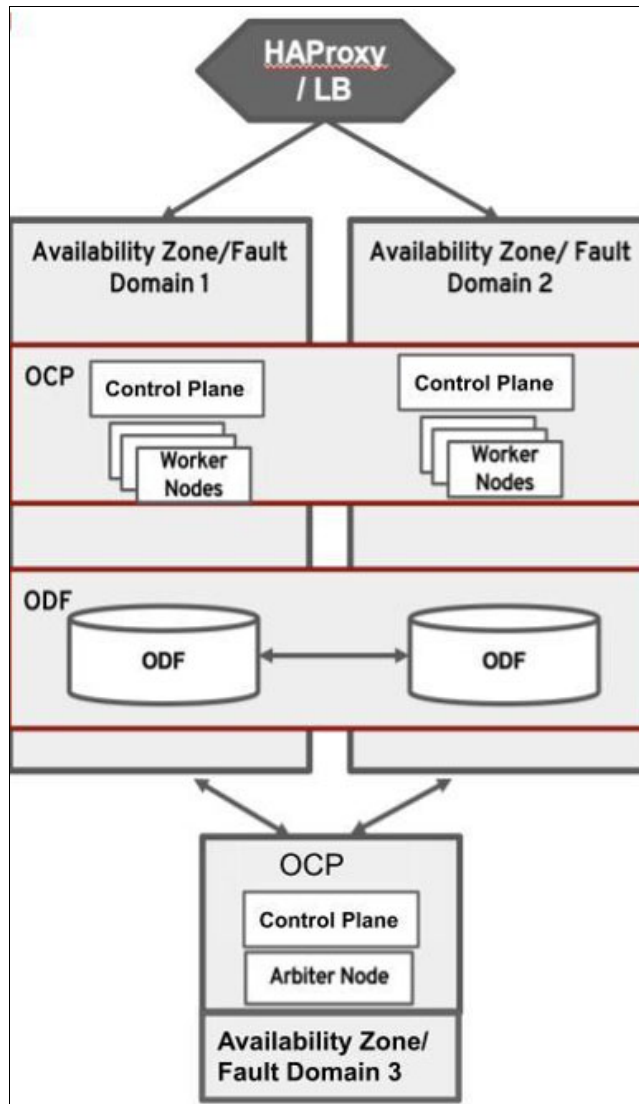


Figure 6-3 IBM Fusion Data Foundation stretched mode

## 6.2 IBM Storage Ceph stretch mode

A *stretch cluster* is a group of servers in separate data centers that are connected by a WAN or campus network. These clusters have fast, low-latency connections that resemble a LAN but with a limited number of links. However, in this scenario, there is a higher likelihood of network splits and the loss of an entire data center, which can affect a significant portion of the cluster.

Ceph is designed for reliable networks and cluster components, with random failures assumed across the CRUSH map. For switch failures, Ceph is designed to route around the loss and maintain data integrity with the remaining object storage daemons (OSDs) and monitors. However, sometimes such as a stretched-cluster deployment, where a significant portion of the cluster relies on a single network component, stretch mode might be necessary to ensure data integrity.

Ceph supports two standard configurations: a two-site setup and a three-site setup. In the two-site setup, Ceph expects each site to have a copy of the data and a third site with a tiebreaker monitor. This monitor selects a winner if the network connection fails and both data centers are operational. The tiebreaker monitor can be a VM and might have high latency compared to the main sites.

The Ceph standard configuration is designed to withstand multiple network or data center failures without compromising data availability. If enough Ceph servers are restored after a failure, the cluster recovers. If a data center is lost but a quorum of monitors is still present, Ceph maintains availability if the cluster has enough data copies to meet the `min_size` configuration option, or the CRUSH rules cause the cluster to rereplicate the data until the `min_size` configuration option is met.

IBM Storage Ceph, when configured in stretched mode, offers synchronous replications between sites for all of its clients: RBD (block), Ceph File System (CephFS) (SharedFS), and RADOS Gateway (RGW) (Object).

Metro-DR is a synchronous replication solution. Write I/O at the primary site is replicated to the secondary site before returning the write acknowledgment, which increases the latency of write workloads.

By default, Ceph uses a replication factor of 4, meaning that two copies of each data object are stored at the primary site and two copies are stored at the secondary site.

Both of the previous points, synchronous replication, and having to do four copies of the data degrades the input/output operations per second (IOPS) performance that you can achieve with Metro-DR, which is especially noticeable with random writes of small blocks.

A in-depth analysis into Metro-DR concepts is available in this Red Hat OpenShift Data Foundation video series:

- ▶ [Metro-DR part 1. RHCS Stretched + Red Hat OpenShift Data Foundation External](#)
- ▶ [Metro-DR part 2. RHCS demo + Connect Red Hat OpenShift Data Foundation External](#)
- ▶ [Metro-DR part 3. ACM + Failover/Failback App Demo](#)
- ▶ [Metro-DR Stateful Application Disaster Recovery with Zero Data Loss](#)

## 6.2.1 Avoiding stretched cluster issues by using Ceph stretched mode

Ceph has strict policies to maintain data integrity and consistency always. Even in a network failure or loss of Ceph nodes, the system automatically restores itself to normal functioning without requiring manual intervention from the operator.

Although Ceph prioritizes data integrity and consistency, stretched clusters might compromise data availability:

- Inconsistent networks.

If a *netsplit* (a disconnection between two servers that splits the network into two pieces) occurs, Ceph might not be able to mark OSDs as down and remove them from the acting placement groups (PGs). This failure to mark down OSDs occurs even though the primary PG cannot replicate data (a situation that under normal non-netsplit circumstances results in the marking of affected OSDs as down and their removal from the PG). If this event happens, Ceph cannot satisfy its durability guarantees, and I/O is blocked.

- Constraints do not guarantee the replication of data across data centers.

However, it might seem that the data is correctly replicated across data centers. For example, in a scenario with two data centers (A and B) and a CRUSH rule that targets three replicas with a `min_size` of 2, the PG might go active with two replicas in A and zero replicas in B. Thus, if A is lost, the data is lost and Ceph cannot operate on it. This situation is difficult to avoid by using only standard CRUSH rules.

## 6.2.2 Stretched mode architecture layout

When using stretch mode in IBM Storage Ceph, there are certain important architectural considerations.

In this section, Figure 6-4 is explained in detail by describing the most important aspects to consider regarding networking, server hardware, and Ceph component placement.

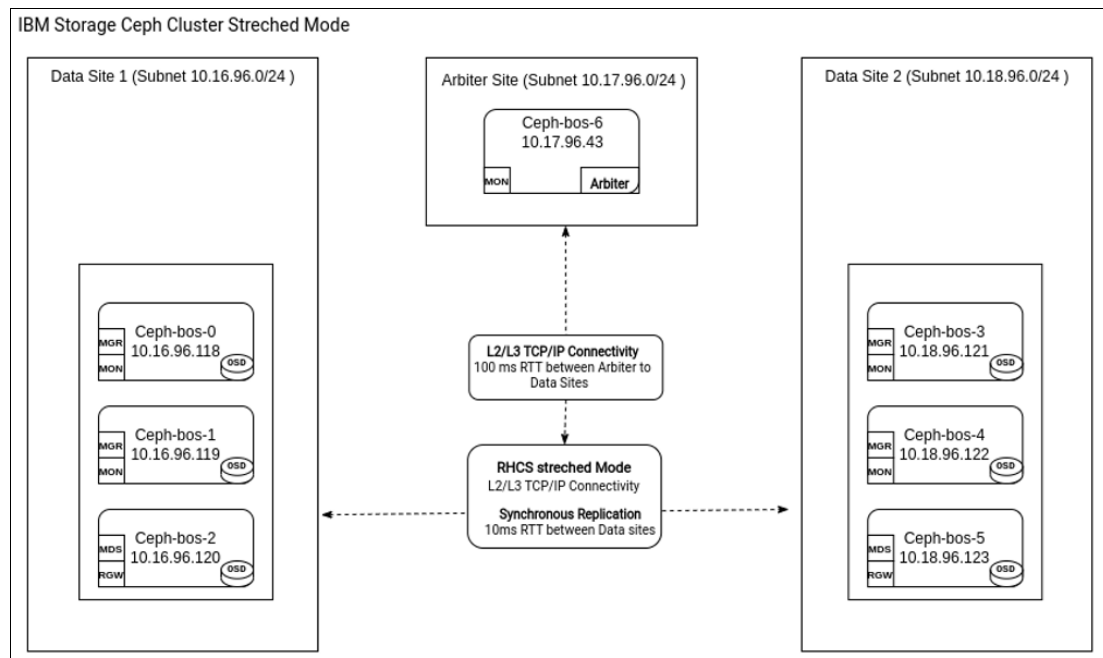


Figure 6-4 IBM Storage Ceph stretched mode



## Network

In a stretched architecture, the network is crucial for the overall health and performance of the cluster.

IBM Storage Ceph has Level 3 routing capabilities to enable different subnets and Classless Inter-Domain Routing (CIDR) on each site and communication between Ceph servers and components through L3 routing.

IBM Storage Ceph can be configured with two different networks, that is, public and cluster. The Ceph public network must be accessible and connected between all three sites (data plus arbiter site) because all IBM Storage Ceph services such as MONS, OSDs, and RGW require communication over the public network. The private or cluster network is used by the OSD services, so the network is configured only on the two data sites where the OSDs are. There is no need to configure the cluster network on the arbiter site.

Network reliability is of huge importance. Route flapping between the three sites introduces data availability and performance issues into the cluster. The network must be accessible and have consistent latency. Frequent spikes in network latency can lead to false alarms and other problems.

Regarding latency, a minimum of 10 ms RTT between the data sites is required to run an IBM Storage Ceph cluster in stretched mode. By data sites, we mean the sites with servers containing OSDs (disks). The latency to the arbiter site can be as high as 100 ms RTT, so it might be beneficial to deploy the arbiter node as a VM in a cloud provider, if possible.

Suppose that the arbiter node is configured on a cloud provider or a remote network that goes through a WAN. If so, a best practice is to set up a VPN between the data sites and the arbiter site, and enable encryption in transit. The messenger v2 encryption feature encrypts the communications between the different Ceph services so that all the communications between the MON in the arbiter site and the other components on the data sites are encrypted.

Because of how Ceph works, latency between sites in a stretch cluster affects both the stability of the cluster and the performance of every write. Ceph follows a strong consistency model, which requires every write to be copied to all OSDs that are configured in the replica count before the client receives an acknowledgment (ACK). The client does receive the ACK for a write until two copies are written to each site, which adds the RTT between sites to the latency of every write workload.

Another important consideration is the throughput between the data sites. The maximum client throughput is limited by the connection (Gbps) between the sites. Recovery from a failed node or site is also important. When a node fails, Ceph always performs I/O recovery from the primary OSD, which means that 66% of the recovery traffic is remote, reading from the other site, and using the inter-site bandwidth that is shared with client I/O.

By default, Ceph always reads data from the primary OSD regardless of its location. This situation can cause a significant amount of cross-site read traffic. The `read_from_local_replica` feature (primary OSD affinity) forces clients to read from the local OSD (the OSD in the same site) instead, regardless of whether it is the primary OSD. Testing has shown that the `read_from_local_replica` feature improves read performance, especially for smaller block sizes, and reduces cross-site traffic. This feature is now available for RBD (block) in the standard internal deployment of FDF in Version 4.13 and 4.14, and work is underway to make it available for Metro-DR deployments. This feature is vital for stretched site setups because it reduces the number of times that you must contact OSDs on the other site for reads, which reduces inter-site traffic.

## Hardware requirements

The requirements and recommendations for the IBM Storage Ceph Nodes hardware are the same as the normal (non-stretched) deployments, except for a couple of differences we cover in this section. For more information, see [IBM Storage Ceph Hardware](#).

Here are the important differences:

- ▶ IBM Storage Ceph in stretched mode supports only full-flash configurations. HDD spindles are not supported. The reasoning is that if you lose a full data center for what might be a long period, you still have two replicas or copies of the data that is available on the remaining site. Replica 2 configurations are supported by only flash media.
- ▶ IBM Storage Ceph supports only replica 4 as the replication schema. Any other replication or erasure coding scheme is not supported. You must calculate your total usable storage.
- ▶ IBM Storage Ceph cluster can be used for only the Metro-DR use case. The cluster cannot be shared for other IBM Storage Ceph use cases, such as an S3 object store for external workloads.
- ▶ IBM Storage Ceph in stretched mode can be deployed in only bare metal or VMware platforms.
- ▶ VMware deployments of IBM Storage Ceph are supported for the Metro-DR use case. Here are some best practices for VMware deployments:
  - Must have reserved resources for CPU and memory for the Ceph VMs.
  - On the VMs, set the latency sensitive option to high.
  - In general, apply all VMware recommendations for low-latency use cases.

VMware Storage that is provided to IBM Storage Ceph for the Metro-DR use case supports using virtual machine disk (VMDK) files for the VMs. However, Ceph performs only as well as the data store that provides the VMDKs. If you use a backing store that uses an underperforming VSAN, expect Ceph performance to be abysmal. Conversely, if you use a dedicated all-flash SAN data store for a specific set of VMs, expect Ceph performance to be better.

- ▶ Avoid using VSAN data stores for Ceph node drives because the performance will not be satisfactory for any I/O-intensive applications.

## Component placement

Ceph services (MONs, OSDs, RGWs, and others) must be placed in a way that eliminates single points of failure and ensures that the cluster can tolerate the loss of a full site without affecting client service.

Here are the requirements for component placement:

- ▶ MONs: A minimum of five MONs is required: Two MONs per data site, and one MON on the arbiter site. This configuration ensures that quorum is maintained with more than 50% of MONs available even when a full site is lost.
- ▶ MGR: You can configure two or four managers, with a minimum of one manager per data site. As a best practice, use four managers to provide high availability (HA) with an active/passive pair on the remaining site in a data site failure.
- ▶ OSDs: Distribute them equally across all the nodes in both data sites. Custom CRUSH rules providing two copies in each site (by using four copies) must be created when configuring the stretch mode in the Ceph cluster.

- ▶ **RGWs:** As a best practice, use at least four RGW services, two per data site to ensure that you can still provide HA for object storage on the remaining site in a site failure.
- ▶ **MetaData Server (MDS):** For CephFS Metadata services, as a best practice, the minimum is four MDS services, two per data site. In a site failure, you still have two MDS services on the remaining site, that is, one active and the other one on standby.

## 6.2.3 Deployment example

This section describes some deployment examples.

### Deploying the Ceph cluster

The deployment of Ceph is documented in Chapter 1, “IBM Storage Ceph Dashboard” on page 1, and in the official documentation at [Installing](#).

During the cluster bootstrap that uses the `cephadm` deployment tool, you can inject a service definition YAML file that does most of the cluster configuration in a single step. Example 6-1 shows an example of how to use a template when deploying an IBM Storage Ceph cluster that is configured in stretch mode. This example must be tailored to your deployment.

*Example 6-1 Example of deploying an IBM Storage Ceph cluster that is configured in stretch mode*

---

```
cat <<EOF > /root/cluster-spec.yaml
service_type: host ## service_type host, adds hosts to the cluster
addr: 10.0.40.2 ## IP addresses, hostnames, and others must be replaced
hostname: ceph1 ## depending on each individual use case
location:      ## Only an example template
  root: default
  Data center: DC1 ## DC1 is the label that we set in the crush map
labels:
  - osd      ## Placement of the services will be done with labels
  - mon      ## This host can be scheduled to run OSD, MON, and MGR
  - mgr      ## services
---
service_type: host
addr: 10.0.40.3
hostname: ceph2
location:
  Data center: DC1
labels:
  - osd
  - mon
  - mgr
---
service_type: host
addr: 10.0.40.4
hostname: ceph3
location:
  Data center: DC1
labels:
  - osd
  - mds
  - rgw
---
service_type: host
addr: 10.0.40.5
```

```

hostname: ceph4
location:
  Data center: DC1
labels:
  - osd
  - mds
  - rgw
---
service_type: host
addr: 10.0.48.2  ## The hosts for DC2 are on another Subnet/CIDR
hostname: ceph4
location:
  root: default
  Data center: DC2 ## Datacenter 2 label for the crush map
labels:
  - osd
  - mon
  - mgr
---
service_type: host
addr: 10.0.48.3
hostname: ceph5
location:
  Data center: DC2
labels:
  - osd
  - mon
  - mgr
---
service_type: host
addr: 10.0.48.4
hostname: ceph6
location:
  Data center: DC2
labels:
  - osd
  - mds
  - rgw
---
service_type: host
addr: 10.0.48.5
hostname: ceph7
location:
  Data center: DC2
labels:
  - osd
  - mds
  - rgw
---
service_type: host  ## The arbiter node. It has no OSDs.
addr: 10.0.49.2    ## so also no Datacenter crush map label
hostname: ceph8
labels:
  - mon
---

```

```

service_type: mon
placement:
  label: "mon"
---
service_type: mds
service_id: mds_filesystem_one
placement:
  label: "mds"
---
service_type: mgr
service_name: mgr
placement:
  count: 4
  label: "mgr"
---
service_type: osd
service_id: all-available-devices
service_name: osd.all-available-devices
placement:
  label: "osd"
spec:
  data_devices:
    all: true ## With this filter all available disks will be used
---
service_type: rgw
service_id: objectgw
service_name: rgw.objectgw
placement:
  count: 4
  label: "rgw"
spec:
  rgw_frontend_port: 8080
EOF

```

---

After the YAML file is modified, use the `--apply-spec` parameter with the `cephadm bootstrap` command so that the YAML file configuration is fed into `cephadm` to get the cluster into the state that is described in the YAML file, as shown in Example 6-2.

*Example 6-2 The cephadm bootstrap command*

```

# cephadm bootstrap --ssh-user=deployment-user --mon-ip 10.0.40.78 --apply-spec
/root/cluster-spec.yaml --registry-json
/root/registry.json

```

---

## 6.2.4 Configuring Ceph in stretched mode

The steps to configure Ceph in stretched mode are documented in [Configuring IBM Storage Ceph stretch mode](#).

IBM Storage Ceph in stretched mode supports only replica 4, so you must have two copies of the data available on each site. Even if you lose a full site, you have two copies of the data that is available.

Using replica 4 reduces performance because Ceph must make one extra write compared to the standard protection schema of replica 3. This extra copy is a factor in lowering the performance because Ceph waits for all replicas to acknowledge the write before responding to the client.

**Tip:** There are several ways to mitigate the performance impact of using replica 4, such as using a high-performance storage back end, increasing the number of OSDs, and using Ceph features such as caching and read-from-local-replica.

How does Ceph write two copies per site? Ceph uses the CRUSH map to determine where to store object replicas. The CRUSH map is a logical representation of the physical hardware layout, with a hierarchy of buckets such as rooms, racks, and hosts.

To configure a stretch-mode CRUSH map, define two data centers under the root bucket, and then define the hosts in each data center. For example, the following terminal output, as shown in Example 6-3, shows a stretch-mode CRUSH map with two data centers, DC1 and DC2, each with three Ceph hosts.

*Example 6-3 The ceph osd tree command*

---

```
# ceph osd tree
```

ID	CLASS	WEIGHT	TYPE NAME	STATUS	REWEIGHT	PRI-AFF
-1		0.58557	root default			
-3		0.29279	data center DC1			
-2		0.09760	host ceph-bos-0			
5	ssd	0.04880	osd.5	up	1.00000	1.00000
10	ssd	0.04880	osd.10	up	1.00000	1.00000
-4		0.09760	host ceph-bos-1			
1	ssd	0.04880	osd.1	up	1.00000	1.00000
6	ssd	0.04880	osd.6	up	1.00000	1.00000
-5		0.09760	host ceph-bos-2			
4	ssd	0.04880	osd.4	up	1.00000	1.00000
9	ssd	0.04880	osd.9	up	1.00000	1.00000
-7		0.29279	data center DC2			
-6		0.09760	host ceph-bos-3			
2	ssd	0.04880	osd.2	up	1.00000	1.00000
7	ssd	0.04880	osd.7	up	1.00000	1.00000
-8		0.09760	host ceph-bos-4			
3	ssd	0.04880	osd.3	up	1.00000	1.00000
8	ssd	0.04880	osd.8	u[	1.00000	1.00000
-9		0.09760	host ceph-bos-5			
0	ssd	0.04880	osd.0	up	1.00000	1.00000
11	ssd	0.04880	osd.11	up	1.00000	1.00000

---

To achieve the goal of having two copies per site, define the failure domain at the pool level. For example, if you take the pool `rbdpool`, you can see that it is using a `crush_rule` with an ID of 1 (Example 6-4).

*Example 6-4 Defining the failure domain at the pool level*

---

```
# ceph osd pool ls detail | grep rbdpool
pool 8 'rbdpool' replicated size 4 min_size 1 crush_rule 1 object_hash rjenkins
pg_num 32 pgp_num 32 autoscale_mode on last_change 4045 lfor 4045/4045/4045 flags
hashpspool,selfmanaged_snaps stripe_width 0 application
rbd
```

---

If you check the failure domain that is configured on `crush_rule 1`, you see that the rule is going to select two hosts from each data center to store the four copies of each client write to the pool (Example 6-5).

*Example 6-5 Checking the failure domain*

---

```
# ceph osd crush rule dump stretch_rule

{
  "rule_id": 1,
  "rule_name": "stretch_rule",
  "type": 1,
  "steps": [
    {
      "op": "take",
      "item": -1,
      "item_name": "default"
    },
    {
      "op": "choose_firstn",
      "num": 0,
      "type": "Data center"
    },
    {
      "op": "chooseleaf_firstn",
      "num": 2,
      "type": "host"
    },
    {
      "op": "emit"
    }
  ]
}
```

---

Figure 6-5 shows the correlation between the OSD crush map and the crush rule that is defined in Example 6-4 on page 112.



Figure 6-5 IBM Storage Ceph Crush map and Crush rule

When stretch mode is enabled, the OSDs take only active PGs when they peer across data centers if both are alive, with the following constraints:

- ▶ Pools increase in size from the default 3 to 4, with two copies on each site.
- ▶ OSDs may connect only to monitors in the same data center.
- ▶ New monitors do not join the cluster if they do not specify a location.

If all the OSDs and monitors from a data center become inaccessible at once, the surviving data center enters a degraded stretch mode, which implies the following items:

- ▶ This mode issues a warning, reduces the pool's `min_size` to 1, and allows the cluster to go active with data in the remaining site.
- ▶ The pool size parameter is not changed, so you also get warnings that the pools are too small.
- ▶ The stretch mode flag prevents the OSDs from creating extra copies in the remaining data center (only two copies are kept, as before).



When the missing data center comes back, the cluster enters recovery stretch mode by triggering the following actions:

- ▶ The mode changes the warning and allows peering, but still requires only OSDs from the data center that was running the whole time.
- ▶ When all PGs are in a known state and are not degraded or incomplete, the cluster migrates back to the regular stretch mode, where the following actions occur:
  - The cluster ends the warning.
  - The cluster restores `min_size` to its starting value (2) and requires both sites to peer.
  - The cluster stops requiring the always-alive site when peering (so you can fail over to the other site, if necessary).

## 6.3 Fusion Metro-DR for RPO Zero

This section describes Fusion Metro-DR for RPO Zero.

### 6.3.1 Architecture: External versus internal Fusion Data Foundation deployments

#### Internal deployment

Deploying IBM Storage FDF entirely within Red Hat OpenShift Container Platform has all the benefits of operator-based deployment and management. You can use the internal-attached device approach in the GUI to deploy FDF in internal mode by using the local storage operator and local storage devices.

Simplicity of deployment and management are the highlights of running FDF services internally on Red Hat OpenShift Container Platform.

#### External deployment

IBM Storage FDF exposes the IBM Storage Ceph services running outside of the Red Hat OpenShift Container Platform cluster as storage classes.

The external approach is best used when in the following situations:

- ▶ The storage requirements are significant (600+ storage devices).
- ▶ Multiple Red Hat OpenShift Container Platform clusters must use storage services from a common external cluster.
- ▶ Another team, such as Site Reliability Engineering (SRE), storage, or others, must manage the external cluster that is providing storage services. This team might exist.
- ▶ There might be a storage team with already deployed Ceph Standalone clusters that like to customize and tune their IBM Storage Ceph clusters.
- ▶ You must have DR with RPO 0 by using the Metro-DR solution.

## 6.3.2 Metro-DR solution components

This section describes the Metro-DR solution components.

### Red Hat Advanced Cluster Management for Kubernetes

RHACM can manage multiple clusters and application lifecycles. It serves as a control plane in a multi-cluster environment.

### IBM Storage Ceph

IBM Storage Ceph is a massively scalable, open, and software-defined storage platform that combines the most stable version of the Ceph storage system with a Ceph management platform, deployment utilities, and support services. It lowers the cost of storing enterprise data and helps organizations manage exponential data growth. The software is a robust and modern petabyte-scale storage platform for public or private cloud deployments.

### Fusion Data Foundation

FDF can provision and manage storage for stateful applications in an Red Hat OpenShift Container Platform cluster. Ceph backs it as the storage provider, whose lifecycle is managed by Rook in the FDF component stack. Ceph-CSI provides the provisioning and management of Persistent Volumes for stateful applications.

### Red Hat OpenShift DR

Red Hat OpenShift DR is a DR orchestrator for stateful applications across a set of peer Red Hat OpenShift clusters, which are deployed and managed by using RHACM, and provide cloud-native interfaces to orchestrate the lifecycle of an application's state on persistent volumes. It provides the following functions:

- ▶ Protecting an application and its state relationship across Red Hat OpenShift clusters.
- ▶ Failing over an application and its state to a peer cluster.
- ▶ Relocating an application and its state to the previously deployed cluster.

Red Hat OpenShift DR is split into three components:

- ▶ IBM Storage FDF Multicluster Orchestrator  
Installed on the hub cluster with RHACM, it orchestrates configuration and peering of FDF clusters for Metro-DR relationships.
- ▶ IBM Storage FDF DR Hub Operator  
Automatically installed as part of IBM Storage FDF Multicluster Orchestrator installation on the hub cluster to orchestrate failover or relocation of DR-enabled applications.
- ▶ IBM Storage FDF DR Cluster Operator  
Automatically installed on each managed cluster that is part of a Metro-DR relationship to manage the lifecycle of all persistent volume claims (PVCs) of an application.

## 6.3.3 Metro-DR architecture and layout

Figure 6-6 on page 117 illustrates a Metro-DR architecture, which features three Red Hat OpenShift clusters, one on each site. The data sites host Red Hat OpenShift clusters that run applications, and the arbiter site houses the ACM Hub cluster, which is responsible for initiating application failover and failback.

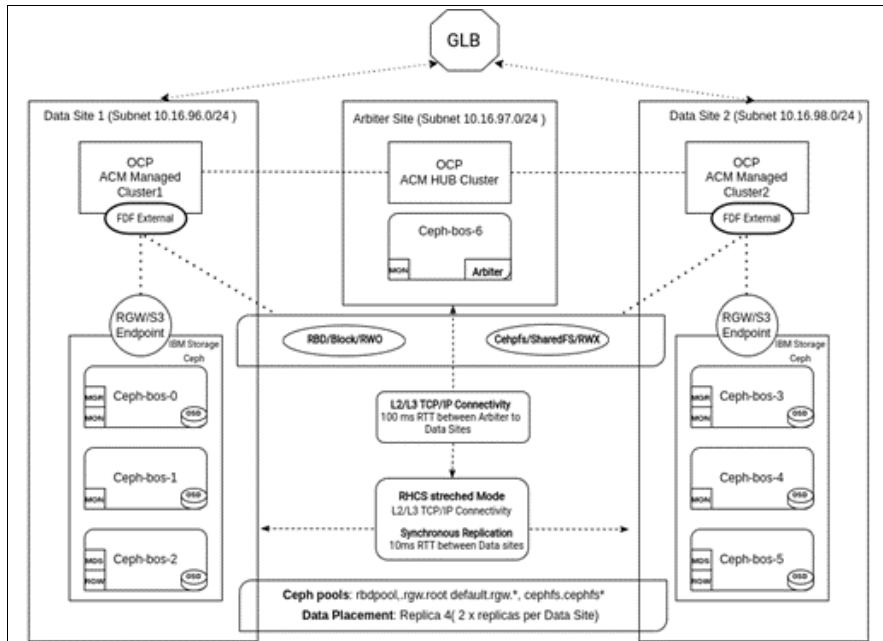


Figure 6-6 IBM Fusion Data Foundation Metro-DR architecture

Both Red Hat OpenShift clusters in the data sites are managed ACM clusters. Each Red Hat OpenShift Container Platform cluster deploys its instance of FDF, and uses the external mode deployment so that each FDF in external mode connects to the same IBM Storage Ceph cluster in stretched mode.

Applications that are protected by Metro-DR are active/passive, which means that the application can be active on only one site at a time. During failover or failback, you can switch the active application from one site to the other site. You can have active applications running on both data sites simultaneously, for example, you can have APP1 running active at Site 1 and passive at Site 2, and APP 2 running active at Site 2 and passive at Site 1. But, independent of the PV access type RWO or RWX, there is support *only* for active/passive for Metro-DR protected applications.

Concurrent access to the same volume by both Red Hat OpenShift clusters that use a single stretched IBM Storage Ceph cluster can corrupt the file system. Avoid this situation. To do so, Metro-DR uses a Ceph fencing mechanism that is called *blocklisting*. You can fence any client by IP or CIDR, and the DR operator takes care of fencing off all worker nodes on the failed site before starting the application on the secondary site.

With Metro-DR, once you start the failover, it is going to be a failover of all applications that Metro-DR protects. You do not have granular, per application failover in Metro-DR because you must fence off all worker nodes that are part of the Red Hat OpenShift cluster to ensure that no file system or PV is still being accessed on the primary/failed site.

By default, failover and failback are triggered manually by an operator, which means that a human must decide to start a failover, and there is no automatic failover based on application health checks. This automation can be easily developed with the tools that are available, but the task would require custom development from the user.

Regarding capacity planning, you must have enough available resources for CPU and memory on the Red Hat OpenShift Container Platform clusters to match all the applications that are protected with Metro-DR.

The global load balancer (GLB) that is shown in Figure 6-6 on page 117 is not included in the Metro-DR solution. Instead, site failure detection and redirection are handled by the site-level components. In a site failure, client requests are automatically redirected to the remaining site. Although application-level health checks can be implemented, they are not as crucial in Metro-DR compared to Regional-DR because Metro-DR involves failing over the entire site, which results in all application requests being redirected to the remaining site.

### 6.3.4 Metro-DR application failover and failback workflow

Currently, two application deployment methods are supported when DR protection is required:

- ▶ ACM Subscription-based application: For an example of deploying an application by using ACM subscriptions, see [Creating subscription-based sample applications](#).
- ▶ ACM ArgoCD ApplicationSet-based application: For an example of deploying an application by using ACM ArgoCD ApplicationSets, see [Creating ApplicationSet-based applications](#).

The failover and failback workflows for both sets of applications are the same at a high level:

1. The monitoring system detected a problem with Site 1. Health checks are failing, and the application is inaccessible. After the operator verifies that there is an issue with Site 1, the failover workflow starts.
2. Fence all worker nodes in Site 1.
3. After the worker nodes are fenced, trigger the failover from the ACM HUB UI.
4. The application PV metadata is restored in Site 2, making the PV available for the application to start.
5. ACM HUB deploys the application in Site 2 by using the PV to restore all previous data.
6. The application successfully starts on Site 2.
7. The GLB detects that the application is healthy on Site 2 and redirects client traffic to Site 2.
8. Service is recovered, and the app is running again with zero data loss.
9. At some point, Site 1 is recovered, and all the FDF and Red Hat OpenShift Data Foundation worker nodes are running again after a restart.
10. Remove the fencing that was in place for the worker nodes in Site 1.
11. Reallocate the service to Site 1, bringing the service back to its original state.

There is a visual slide deck of the Failover and Failback workflow at [Flowchart Maker and Online Diagram Software](#). There is also a YouTube video by Annett Clewett that shows the failover and failback of an application by using a queue from the IBM MQ series product. You can find this video at [YouTube](#).

### 6.3.5 Metro-DR deployment example

For more information about the deployment steps for Metro-DR, see the [Metro-DR deployment workflow](#).



# Amazon S3 enterprise user authentication and authorization

This chapter describes IBM Storage Ceph Object Storage authentication methods, and using the S3 API Secure Token Service (STS) and Identity and Access Management (IAM) roles to provide S3 users with the means to authenticate against an Identity Provider (IDP) and gain access to specific S3 data sets based on their role.

This chapter has the following sections:

- ▶ Introducing RADOS Gateway Auth methods
- ▶ Step-by-step deployment guide: S3 enterprise user authentication/authorization
- ▶ IBM Storage Ceph STS configuration
- ▶ Testing Assume Role With Web Identity for role-based access control

## 7.1 Introducing RADOS Gateway Auth methods

To access object storage by using the S3 protocol, you must provide an S3 access key and a secret key. The content of the access key might vary depending on the implementation of the S3 API, but it serves as your identity. When you send a request to the S3 API, the API checks whether it knows your s3 access key and looks up the associated secret key. The secret key is a shared secret between you and the S3 API and should not be sent over the wire to S3. Instead, the secret key is used to create a Hash-based Message Authentication Code (HMAC) to sign each request that is sent to s3, where it is validated and, on successful validation, ran. S3 treats a successful validation as proof that you hold the secret key and are the identity that you sent.

The following options are available to authenticate users against IBM Storage Ceph Object Storage:

- ▶ RADOS Gateway (RGW) local database (DB) authentication
- ▶ LDAP authentication
- ▶ Keystone authentication
- ▶ STS:
  - With LDAP (STS-lite)
  - With keystone (STS-lite)
  - With OpenID Connect (OIDC) authentication

### 7.1.1 Introducing the RGW Secure Token Service

Amazon STS is a set of APIs that return a temporary set of S3 access and secret keys. IBM Storage Ceph supports a subset of Amazon STS APIs. Users authenticate against STS and on success receive a short-lived S3 access and secret key that can be used in subsequent requests.

IBM Storage Ceph Object STS provides availability to authenticate object storage users against your enterprise IDP (LDAP, AD, and others) through an OIDC (SSO) provider. STS also avoids using S3 long-lived keys. STS provides temporary and limited privilege credentials that enhance your object storage security policy.

STS can be configured to work with several authentication methods, such as LDAP, keystone, or OIDC (Figure 7-1 on page 121).

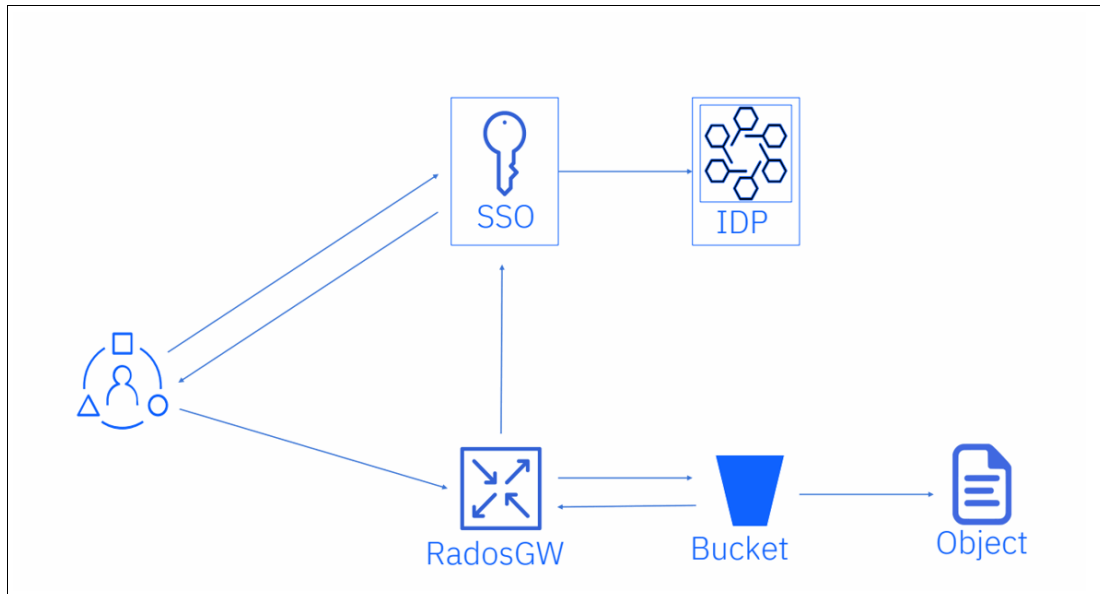


Figure 7-1 Security Token Service configuration methods

## 7.1.2 Introducing STS with an OIDC provider

With OAuth authentication, you use an OIDC service to authenticate. You must have an OIDC or OAuth2 compatible service. Red Hat tested the integration with RGW with the following OIDC products: Red Hat SSO and Keycloak IDPs.

With this method, the authentication of a user follows this high-level workflow:

1. The user authenticates against an OIDC to get a JSON Web Token (JWT).
2. The user calls the `AssumeRoleWithWebIdentity` API against the STS API, and passes in the role that they want to access and the path to the JWT.
3. RGW checks with the OIDC provider about the validity of the JWT.
4. STS creates and provides temporary credentials for the user. The user can access the S3 resources with a specific role.

For more information, see Figure 7-2.

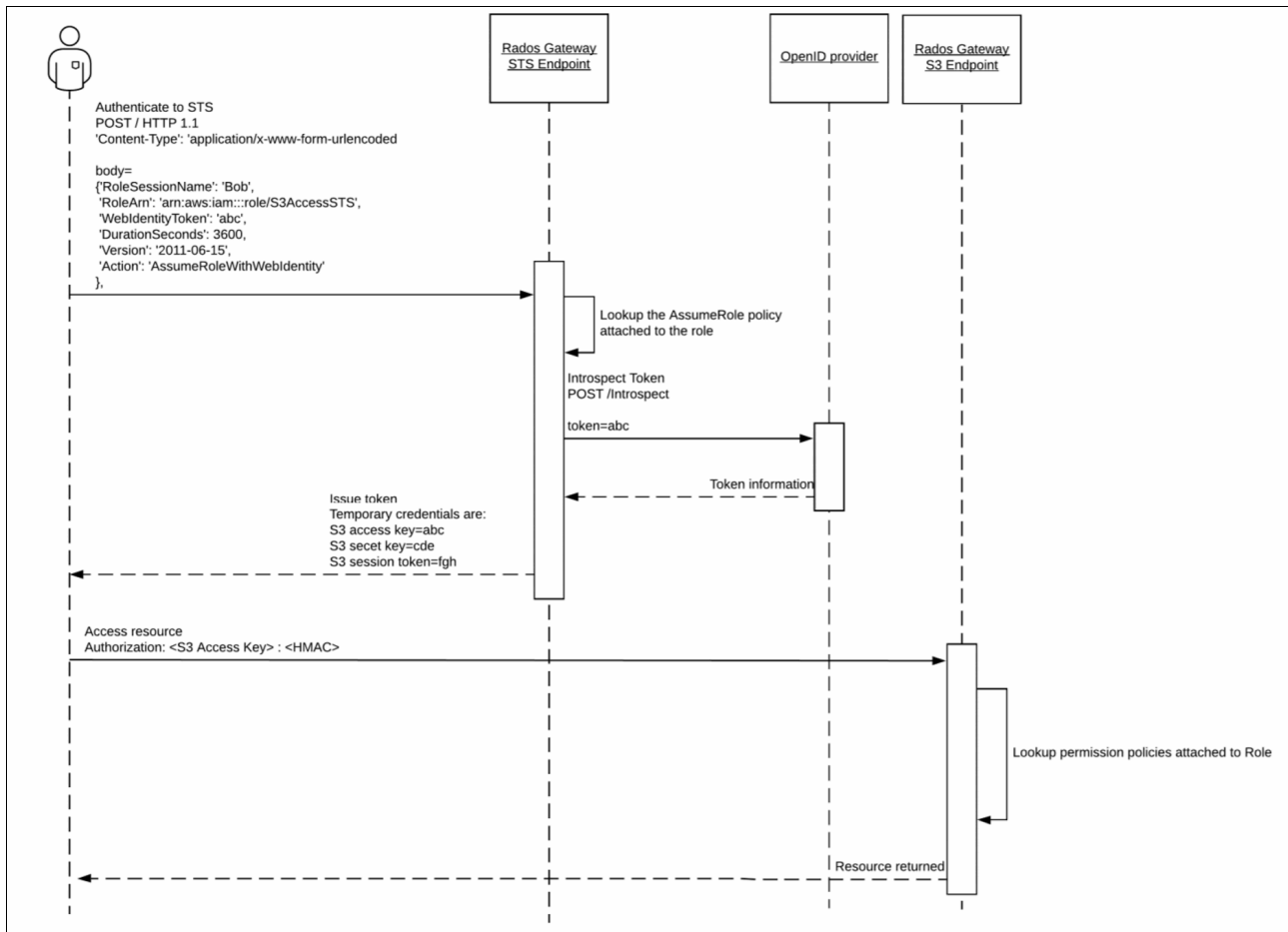


Figure 7-2 High-level authentication workflow

### 7.1.3 Introducing IAM role policies

To understand IAM role policies, you must understand the concepts of an Amazon Resource Name (ARN), permission policies, a bucket policy, and roles.

An *ARN* is used to identify a user, a group, or a role among them. The ARN of a username has the general format of `arn:aws:iam:::user1`. ARNs are important because they are required in the permission policy language that S3 uses.

A *permission policy* instructs what actions are allowed or denied on a set of resources by a set of principals. Principals are identified through an ARN.

Permission policies are used in two instances: bucket policy and role policy.

A *bucket policy* determines what actions can be performed inside a S3 bucket, and it is used to restrict who can read/write objects. The bucket policy is administered through the S3 API, and it can be self-managed by users if they have the appropriate permissions defined.



A *role* is similar to a user and has its own ARN in the format of `arn:aws:iam::role/rolename`. In a bucket policy, instead of giving access to users based on their user ARN, a role ARN can be used instead. Users can assume the role based on another permission policy, which is referred to as the `assume-role-policy-doc` policy. The `assume-role-policy-doc` grants access to the roles if the user can meet its conditions.

With IAM role policies, during STS authentication, users can request to assume a role and inherit all the S3 permissions that are configured for that role by an RGW administrator so that the users can configure role-backed access control (RBAC) or attribute-based access control (ABAC) authorization policies (Figure 7-3).

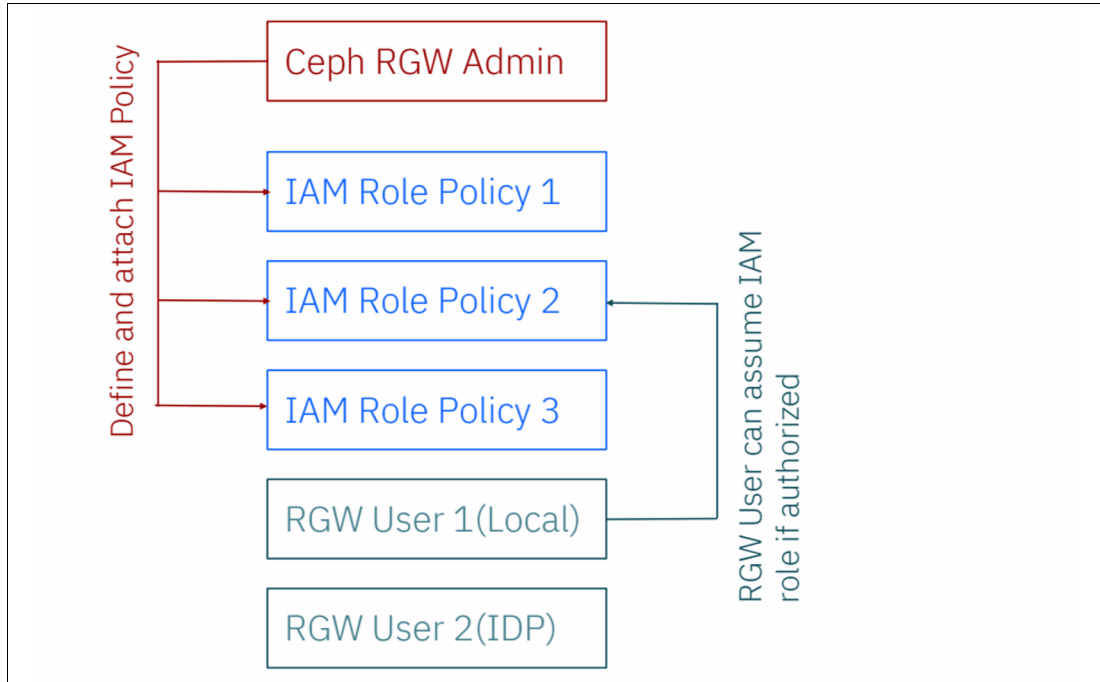


Figure 7-3 IAM role policies

ABAC authorization mode is an authorization model that evaluates attributes (or characteristics) rather than roles to determine access (Figure 7-4).

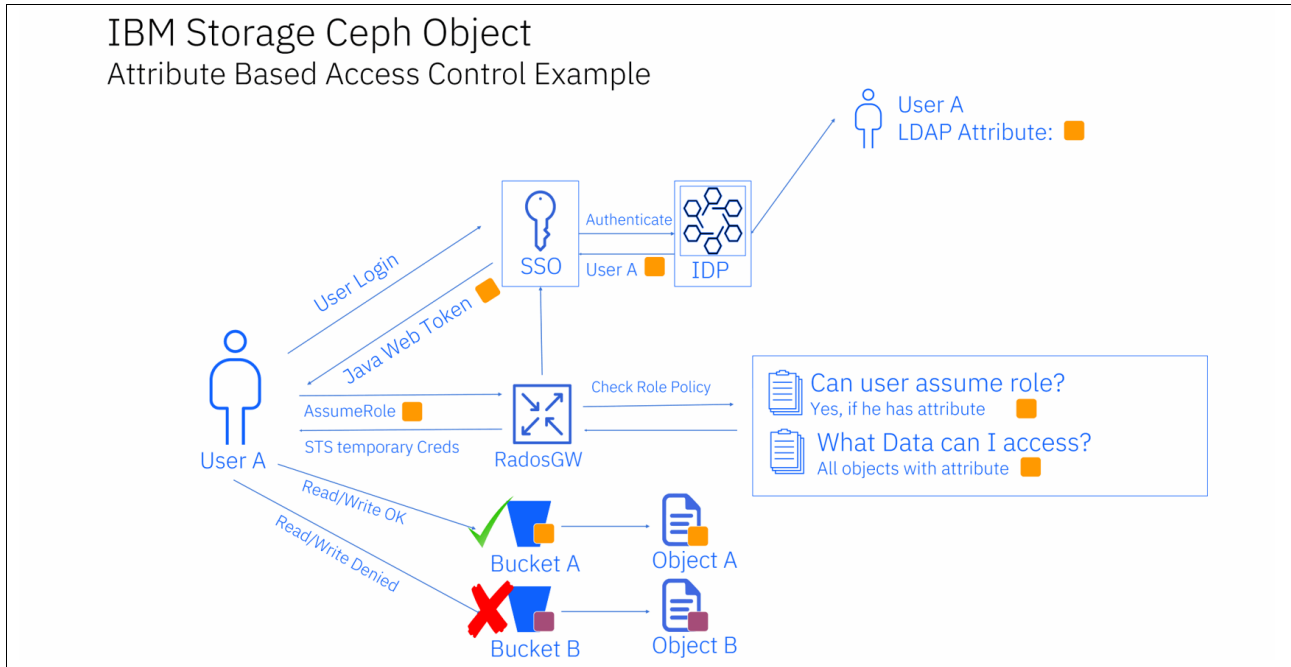


Figure 7-4 Attribute-based access control

## 7.2 Step-by-step deployment guide: S3 enterprise user authentication/authorization

The main focus of this chapter is to show an example of how to configure IBM Storage Ceph Object Storage to authenticate against an SSO (OIDC) and use IAM roles for authorization and shared data access.

This section describes the basic deployment and configuration of Keycloak and Red Hat Identity Management Server (IdM) because you need an SSO (OIDC) and an IDP to test the STS/IAM functions. The configuration that we are using for Keycloak and IdM is only for functional testing. For more information about putting a production deployment in place, see [RHSSO \(Keycloak\)](#) and [IDM](#).

## 7.2.1 Installing Red Hat SSO (Keycloak) on Red Hat OpenShift

To install Red Hat SSO (Keycloak) on Red Hat OpenShift, complete the following steps:

1. Create a project on Red Hat OpenShift Platform with a name of your preference (Example 7-1).

*Example 7-1 Creating a project on Red Hat OpenShift Platform*

```
[root@ocpbastion ~]# oc new-project kcsso
Now using project "kcsso" on server "https://api.ocp.stg.local:6443".
You can add applications to this project with the 'new-app' command. For example,
try:
```

```
    oc new-app rails-postgresql-example
to build a new example application in Ruby. Or use kubectl to deploy a simple
Kubernetes application:
    kubectl create deployment hello-node
--image=k8s.gcr.io/e2e-test-images/agnhost:2.33 -- /agnhost serve-hostname
```

2. Browse to Red Hat OpenShift Operator Hub and search for Red Hat Single Sign-On Operator (Figure 7-5).

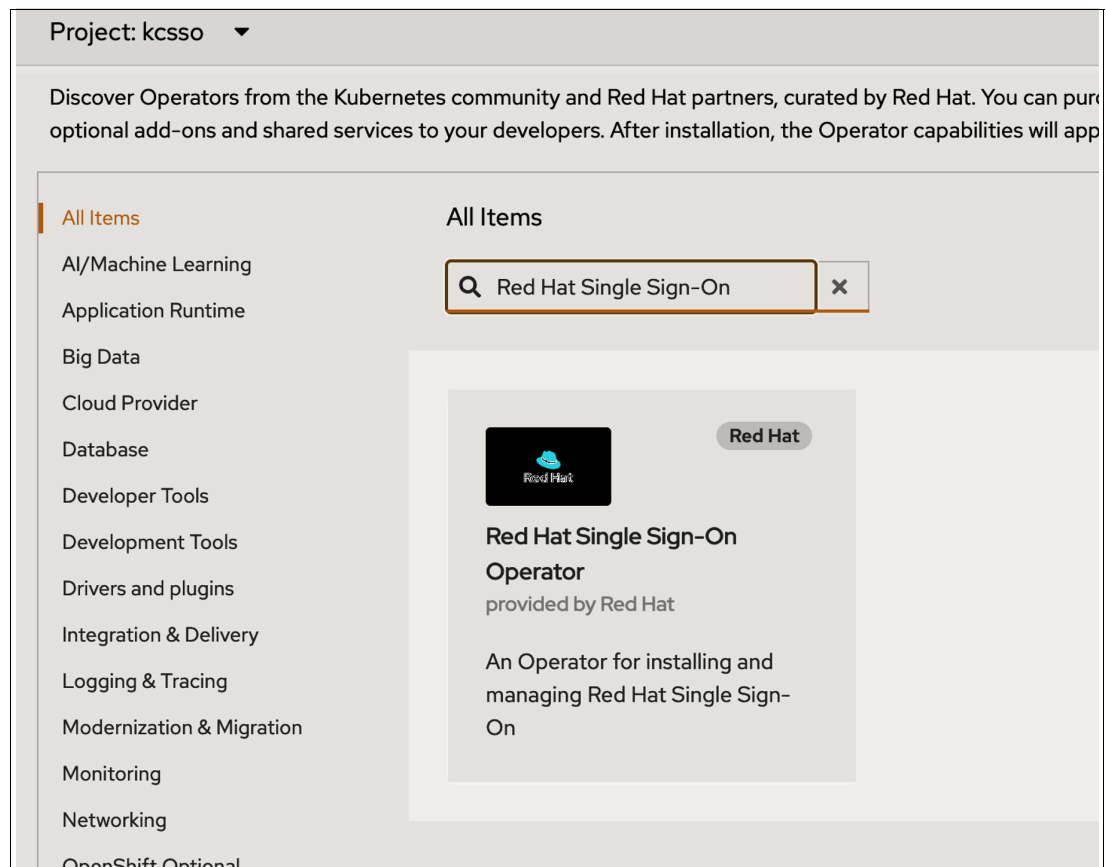


Figure 7-5 Searching for Red Hat Single Sign-On Operator

3. Install Red Hat Single Sign-On Operator (Figure 7-6).

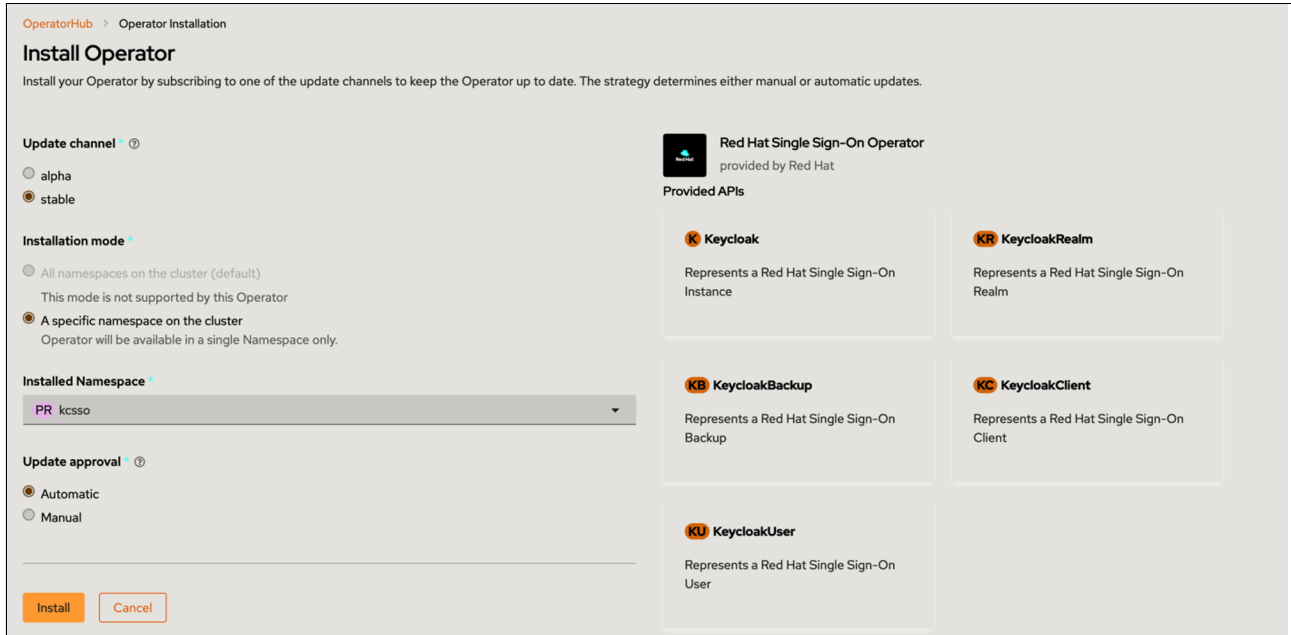


Figure 7-6 Install Operator

4. After the installation of the operator is complete, go to the Red Hat Single Sign-On Operator page and create a Keycloak instance (Figure 7-7).

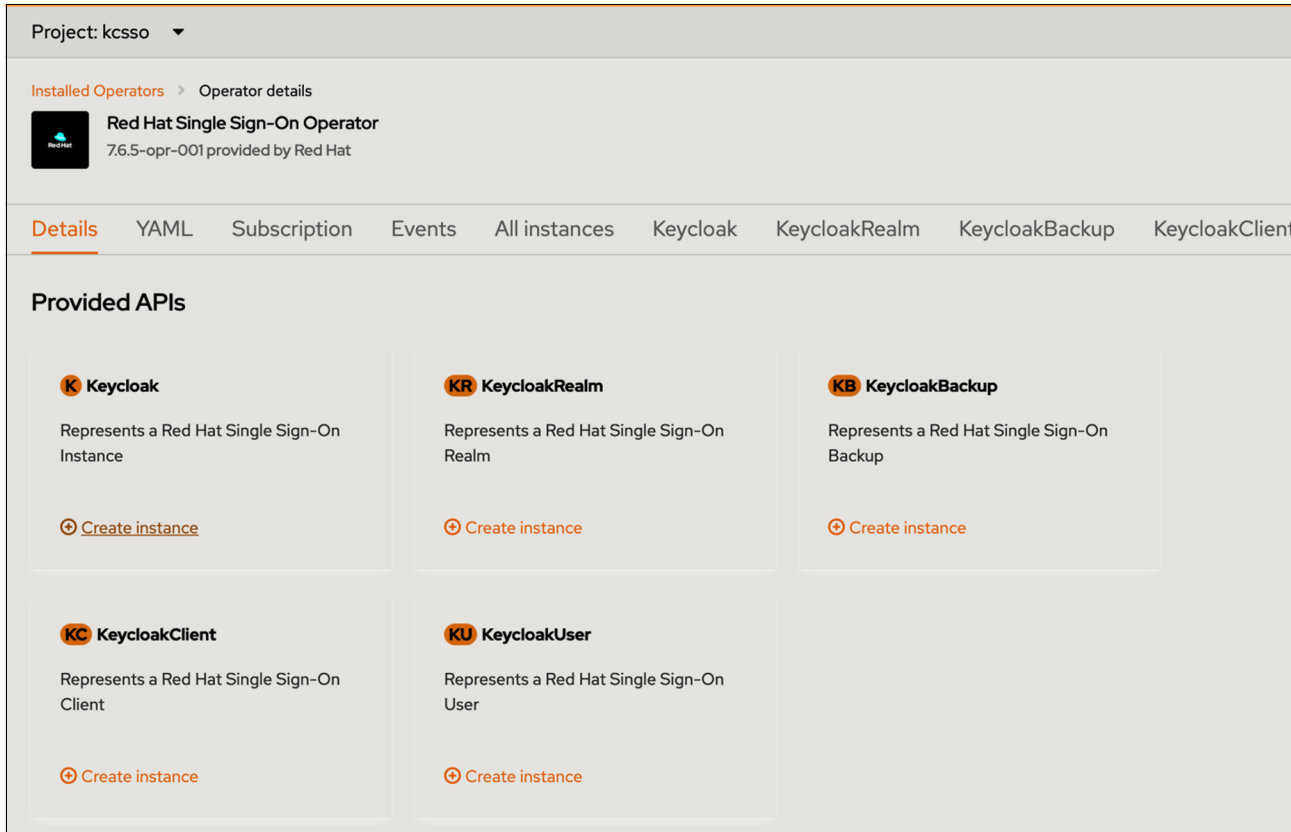
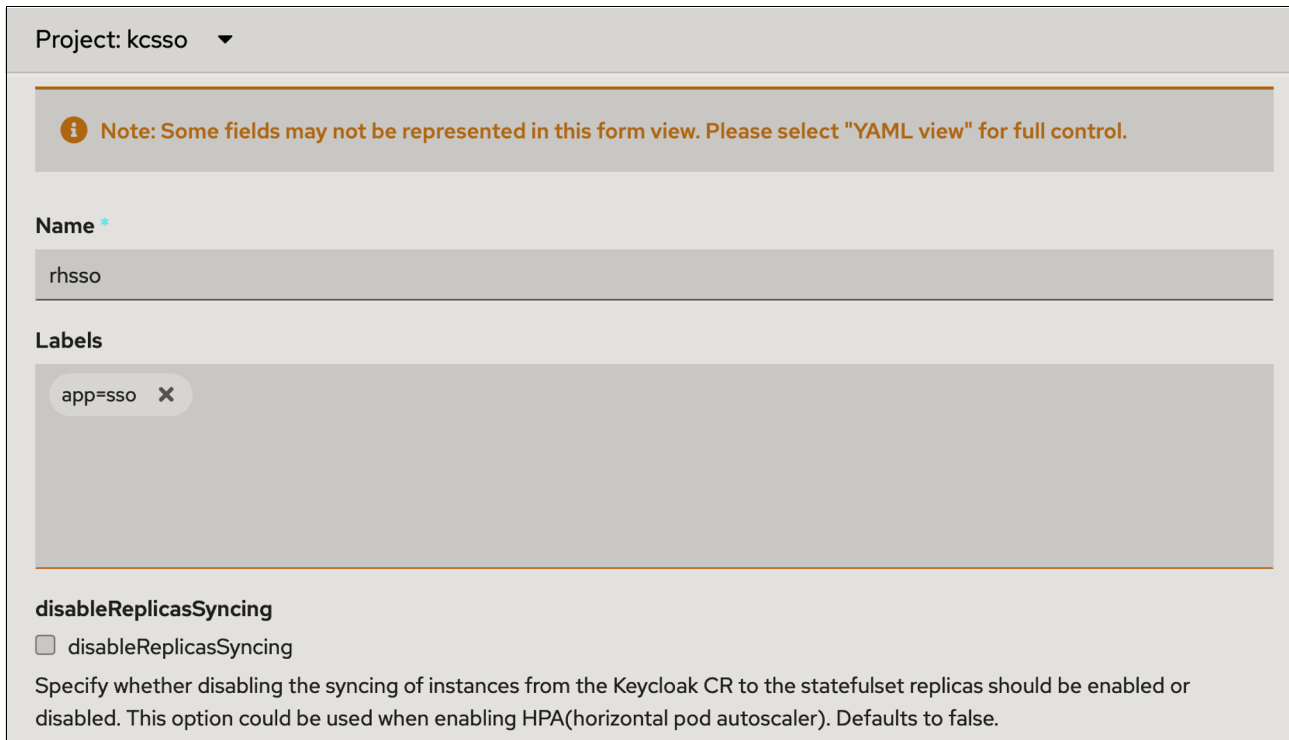


Figure 7-7 Red Hat Single Sign-On Operator

5. Name your Keycloak instance (Figure 7-8).



Project: kcsso ▾

**Note:** Some fields may not be represented in this form view. Please select "YAML view" for full control.

**Name \***

rhsso

**Labels**

app=sso ✕

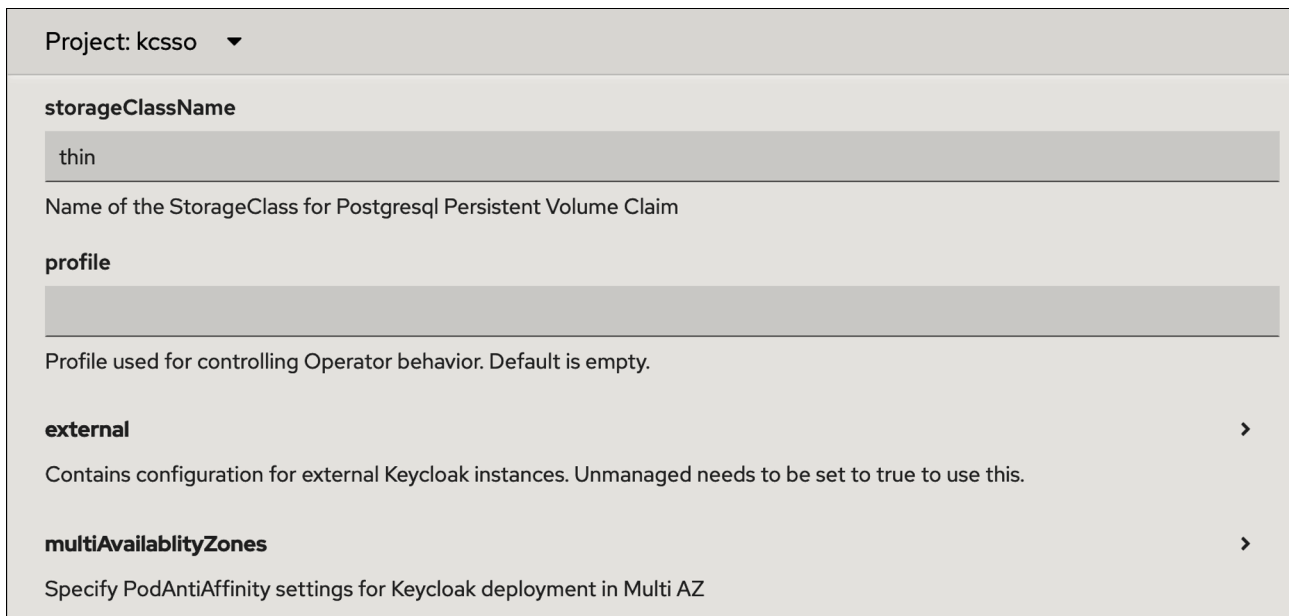
**disableReplicasSyncing**

disableReplicasSyncing

Specify whether disabling the syncing of instances from the Keycloak CR to the statefulset replicas should be enabled or disabled. This option could be used when enabling HPA(horizontal pod autoscaler). Defaults to false.

Figure 7-8 Naming your Keycloak instance

6. Because in this example we are not doing an advanced installation of Keycloak, we specify only the storage class name that we want to use and leave the other settings empty (Figure 7-9).



Project: kcsso ▾

**storageClassName**

thin

Name of the StorageClass for Postgresql Persistent Volume Claim

**profile**

Profile used for controlling Operator behavior. Default is empty.

**external** >

Contains configuration for external Keycloak instances. Unmanaged needs to be set to true to use this.

**multiAvailabilityZones** >

Specify PodAntiAffinity settings for Keycloak deployment in Multi AZ

Figure 7-9 Adding the Storage Class name

The operator now installs a Keycloak instance and a PostgreSQL to keep the records. The storage requirement for the total installation is two 2 GB. The total installation time might take up to 1 hour, depending on the resources of your cluster.

After the installation, you can reach the Red Hat SSO admin credentials from the Secrets page of the project. The values are stored under the `credential-keycloak` secret.

## 7.2.2 Installing Red Hat Identity Management Server

Red Hat Identity Management Server (IdM) can be installed with various options:

- ▶ Using integrated DNS with integrated CA as the root CA
- ▶ Using integrated DNS with external CA as the root CA
- ▶ Using external DNS with integrated CA as the root CA
- ▶ Using external DNS with external CA as the root CA

If you want to use an external DNS server with your IdM installation, a list of records that must be added to the DNS server is provided by IdM at the end of the installation. The installation is complete without adding these records.

**Note:** The DNS server within IdM is not intended for general-purpose external use.

Before the installation, configure your Red Hat Enterprise Linux (RHEL) system for IdM.

## 7.2.3 Hostname and DNS requirements

Update your `/etc/hosts` file to include your server IP address with an FQDN. Your hostname should not be `localhost` or `localhost6` (Example 7-2).

*Example 7-2 Updating your `/etc/hosts` file*

---

```
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1      localhost localhost.localdomain localhost6 localhost6.localdomain6
172.16.29.32 idm.stg.local
```

---

Make sure to set your hostname by using the `nmtui` tool, for example. The installer may change the hostname to `localhost` and fail to start. The installer fails too.

## 7.2.4 Port requirements

Table 7-1 shows the ports that should be open for IdM.

*Table 7-1 Ports that need to be open for IdM:*

Service	Ports	Protocol
HTTP/HTTPS	80 and 443	TCP
LDAP/LDAPS	389 and 636	TCP
Kerberos	88 and 464	TCP and UDP
DNS	53	TCP and UDP (optional)

## 7.2.5 Installing packages

IdM installers are in the BaseOS and AppStream repositories. Enable these repositories by running the following commands:

```
[root@idm ~]# subscription-manager repos --enable=rhel-9-for-x86_64-baseos-rpms
[root@idm ~]# subscription-manager repos --enable=rhel-9-for-x86_64-appstream-rpms
```

Download the packages that are required to install IdM with integrated DNS by running the following command:

```
[root@idm ~]# dnf install ipa-server ipa-server-dns
```

The IdM installer requires that the `umask` value to be set to 0022 for the root account to allow other users to read files that are created during installation. If this value is not set to 0022, the installer sends a warning, and some functions of the IdM will not be working correctly. You can change this value to its original setting after the installation (Example 7-3).

*Example 7-3 Setting the umask value to 0022*

---

```
#Display current umask value
[root@idm ~]# umask
0027
#Set the new umask value
[root@idm ~]# umask 0022
#Verify the umask value:
[root@idm ~]# umask
0022
```

---

## 7.2.6 Installing IdM

Now, start the IdM installation. Because you will not perform any configurations during the installation, the default values will be used. You can find an example of answers to installer questions in Example 7-4.

*Example 7-4 Starting the IdM installation*

---

```
[root@idm ~]# ipa-server-install
The log file for this installation can be found in /var/log/ipaserver-install.log
=====
This program sets up the IPA Server.
Version 4.10.1
This includes:
* Configure a stand-alone CA (dogtag) for certificate management
* Configure the NTP client (chronyd)
* Create and configure an instance of Directory Server
* Create and configure a Kerberos Key Distribution Center (KDC)
* Configure Apache (HTTPd)
* Configure SID generation
* Configure the KDC to enable PKINIT
To accept the default shown in brackets, press the Enter key.
Do you want to configure integrated DNS (BIND)? [no]:
Enter the fully qualified domain name of the computer
on which you're setting up server software. Using the form
<hostname>.<domainname>
Example: master.example.com
Server hostname [idm.stg.local]:
```

The domain name has been determined based on the hostname.  
Confirm the domain name [stg.local]:  
The Kerberos protocol requires a Realm name to be defined.  
This is typically the domain name converted to uppercase.  
Provide a realm name [STG.LOCAL]:  
Certain directory server operations require an administrative user.  
This user is referred to as the Directory Manager and has full access  
to the Directory for system management tasks and will be added to the  
instance of directory server created for IPA.  
The password must be at least 8 characters long.  
Directory Manager password: <<ENTER DIRECTORY MANAGER PASSWORD>>  
Password (confirm): <<VERIFY DIRECTORY MANAGER PASSWORD>>  
The IPA server requires an administrative user, who is named 'admin'.  
This user is a regular system account that is used for IPA server administration.  
IPA admin password: <<ENTER ADMIN PASSWORD>>  
Password (confirm): <<VERIFY ADMIN PASSWORD>>  
Trust is configured but no NetBIOS domain name is found, setting it now.  
Enter the NetBIOS name for the IPA domain.  
Only up to 15 uppercase ASCII letters, digits and dashes are allowed.  
Example: EXAMPLE.

NetBIOS domain name [STG]:  
Do you want to configure chrony with NTP server or pool address? [no]:

The IPA Master Server will be configured with:  
Hostname: idm.stg.local  
IP addresses: 172.16.28.185  
Domain name: stg.local  
Realm name: STG.LOCAL  
The CA will be configured with:  
Subject DN: CN=Certificate Authority,O=STG.LOCAL  
Subject base: O=STG.LOCAL  
Chaining: self-signed  
Continue to configure the system with these values? [no]: yes

---

After a successful installation, the IdM installation provides a list of records to add to your DNS server under the /tmp directory. Add records in this file to your DNS system /tmp/ipa.system.records.hazugazn.db. Example 7-5 shows the contents of this list.

*Example 7-5 Adding records to /tmp/ipa.system.records.hazugazn.db*

---

```
[root@idm ~]# cat /tmp/ipa.system.records.hazugazn.db
_kerberos-master._tcp.stg.local. 3600 IN SRV 0 100 88 idm.stg.local.
_kerberos-master._udp.stg.local. 3600 IN SRV 0 100 88 idm.stg.local.
_kerberos._tcp.stg.local. 3600 IN SRV 0 100 88 idm.stg.local.
_kerberos._udp.stg.local. 3600 IN SRV 0 100 88 idm.stg.local.
_kerberos.stg.local. 3600 IN TXT "stg.LOCAL"
_kerberos.stg.local. 3600 IN URI 0 100 "krb5srv:m:tcp:idm.stg.local."
_kerberos.stg.local. 3600 IN URI 0 100 "krb5srv:m:udp:idm.stg.local."
_kpasswd._tcp.stg.local. 3600 IN SRV 0 100 464 idm.stg.local.
_kpasswd._udp.stg.local. 3600 IN SRV 0 100 464 idm.stg.local.
_kpasswd.stg.local. 3600 IN URI 0 100 "krb5srv:m:tcp:idm.stg.local."
_kpasswd.stg.local. 3600 IN URI 0 100 "krb5srv:m:udp:idm.stg.local."
_ldap._tcp.stg.local. 3600 IN SRV 0 100 389 idm.stg.local.
```

---



These records are required DNS records so that your IdM can respond to requests properly. Therefore, the installation is not complete without adding these records.

If you are using a DNS server that does not support URI record registrations, you can install IdM by using the `--allow-zone-overlap` option. With this option, ensure that your DNS server is forwarding to IdM properly. You can find example installation configuration answers to the `--allow-zone-overlap` option as shown in Example 7-6.

*Example 7-6 Installing IdM with --allow-zone-overlap option enabled*

---

```
[root@idm ~]# ipa-server-install --allow-zone-overlap
```

```
The log file for this installation can be found in /var/log/ipaserver-install.log
=====
```

```
This program sets up the IPA Server.
Version 4.10.1
```

This includes:

- \* Configure a stand-alone CA (dogtag) for certificate management
- \* Configure the NTP client (chronyd)
- \* Create and configure an instance of Directory Server
- \* Create and configure a Kerberos Key Distribution Center (KDC)
- \* Configure Apache (HTTPd)
- \* Configure SID generation
- \* Configure the KDC to enable PKINIT

To accept the default shown in brackets, press the Enter key.

**Do you want to configure integrated DNS (BIND)? [no]: yes**

Enter the fully qualified domain name of the computer on which you're setting up server software. Using the form `<hostname>.<domainname>`

Example: master.example.com

Server hostname [idm.stg.local]:

Warning: skipping DNS resolution of host idm.stg.local  
The domain name has been determined based on the hostname.

Confirm the domain name [stg.local]:

The Kerberos protocol requires a Realm name to be defined. This is typically the domain name that is converted to uppercase. Provide a realm name [STG.LOCAL]:  
Certain directory server operations require an administrative user. This user is referred to as the Directory Manager and has full access to the Directory for system management tasks and will be added to the instance of directory server created for IPA.  
The password must be at least 8 characters long.

Directory Manager password: <<Enter directory manager password>>

Password (confirm): <<Verify directory manager password>>

The IPA server requires an administrative user, who is named 'admin'. This user is a regular system account that is used for IPA server administration.

IPA admin password: <<Enter admin password>>  
Password (confirm): <<Verify admin password>>

Checking DNS domain stg.local, wait ...  
DNS zone stg.local. already exists in DNS and is handled by servers:  
dc001.stg.local. Make sure that the domain is properly delegated to this IPA  
server.

**Do you want to configure DNS forwarders? [yes]:**

The following DNS servers are configured in /etc/resolv.conf: 172.16.28.100  
Do you want to configure these servers as DNS forwarders? [yes]:  
All detected DNS servers were added. You can enter additional addresses now:  
Enter an IP address for a DNS forwarder, or press Enter to skip:  
DNS forwarders: 172.16.28.100

Checking DNS forwarders, wait ...

Do you want to search for missing reverse zones? [yes]:

A reverse record for IP address 172.16.29.32 already exists  
Trust is configured but no NetBIOS domain name is found, setting it now.  
Enter the NetBIOS name for the IPA domain.  
Only up to 15 uppercase ASCII letters, digits and dashes are allowed.  
Example: EXAMPLE.

NetBIOS domain name [STG]:

Do you want to configure chrony with NTP server or pool address? [no]:

The IPA Master Server will be configured with:

Hostname: idm.stg.local  
IP addresses: 172.16.29.32  
Domain name: stg.local  
Realm name: STG.LOCAL

The CA will be configured with:

Subject DN: CN=Certificate Authority,0=STG.LOCAL  
Subject base: O=STG.LOCAL  
Chaining: self-signed

**BIND DNS server will be configured to serve IPA domain with:**

**Forwarders: 172.16.28.100**

**Forward policy: only**

**Reverse zones: No reverse zone**

Continue to configure the system with these values? [no]: yes

---

## Configuring IdM users and groups

In this example, we have three user groups, with one user in each group. IdM is the LDAP server for Red Hat SSO (Keycloak) integration. Red Hat SSO is synchronized from IdM with the users and their respective groups.

Complete the following steps:

1. Log in to the IdM interface from your browser by using the admin user and the password that you defined during the installation. You see the user's page. Go to the **Groups** section under the **Identity** tab and add the following groups, as shown in Figure 7-10 on page 133:
  - rgwadmins
  - rgwreaders
  - rgwwriters

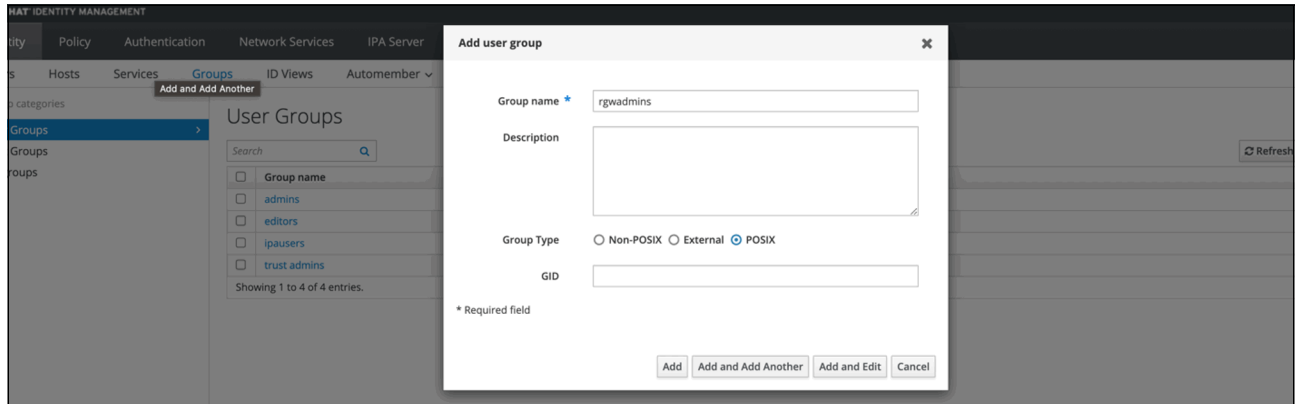


Figure 7-10 Add user group

2. After creating the groups, go to the **Users** tab and add the users. The following users should be added:

- s3admin
- s3reader
- s3writer

You can add users to their respective groups while creating them, as shown in Figure 7-11.

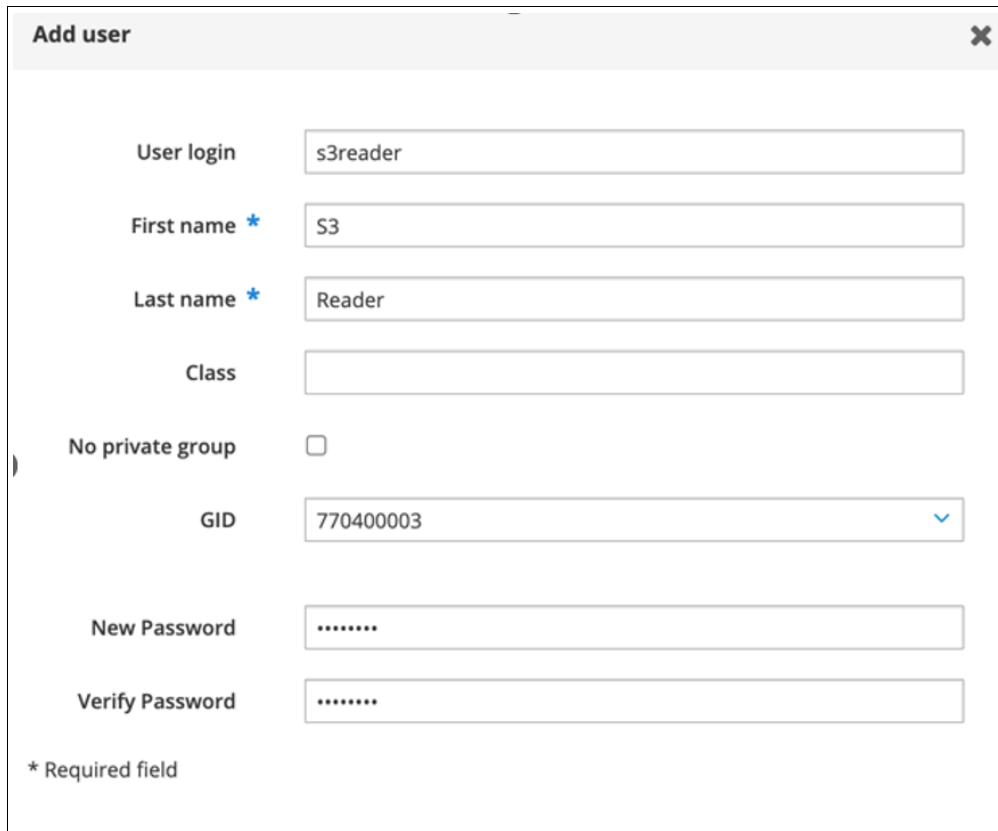


Figure 7-11 Add user

- After creating the users successfully (s3admin, s3writer, and s3reader) with their groups, they should look like the Figure 7-12 (The picture shows three images that are combined into one).

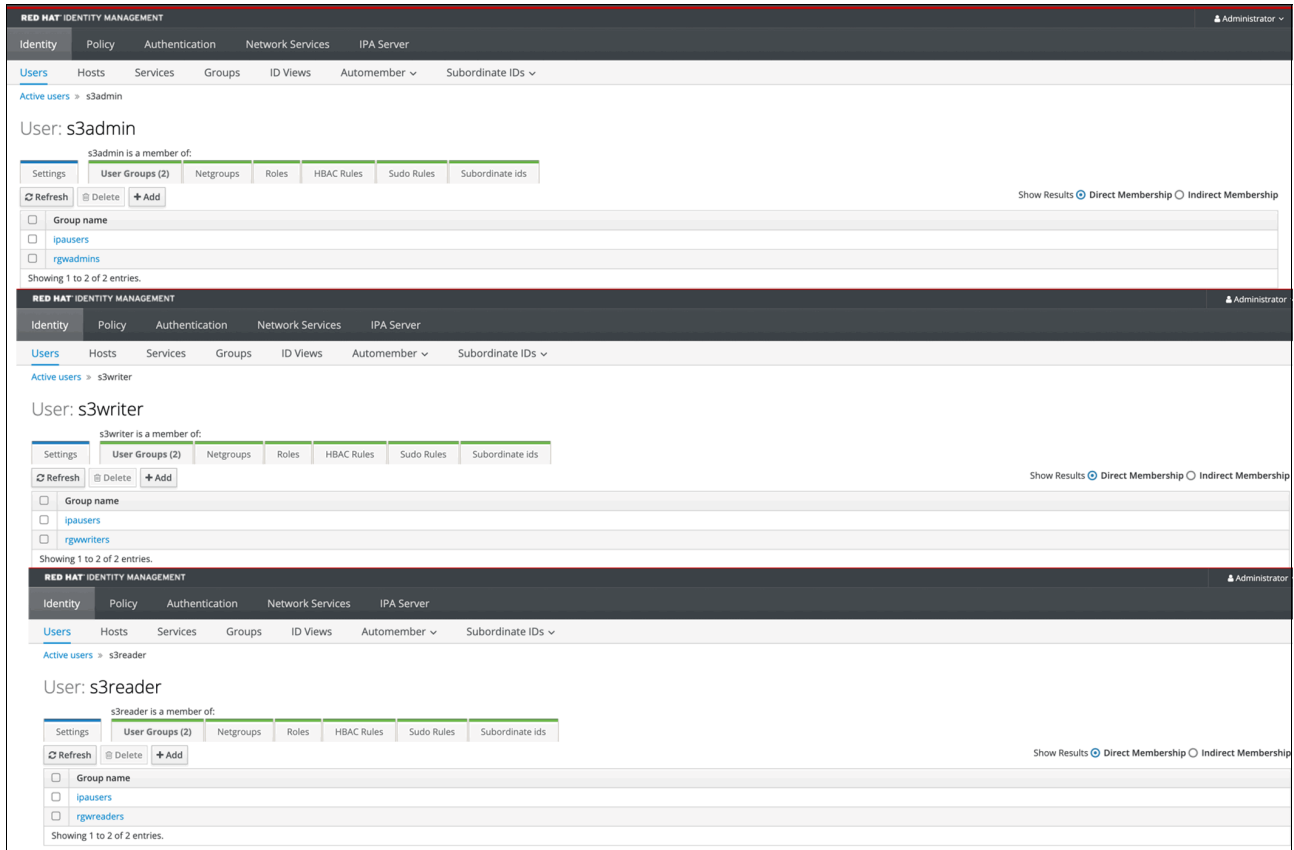


Figure 7-12 Successful creation of users

## 7.2.7 Configuring Red Hat Single Sign-On (Keycloak)

To configure Red Hat Single Sign-On (Keycloak), complete the following steps.

- Sign in to your Red Hat SSO instance as an admin user and create a realm with the name Ceph (Figure 7-13).

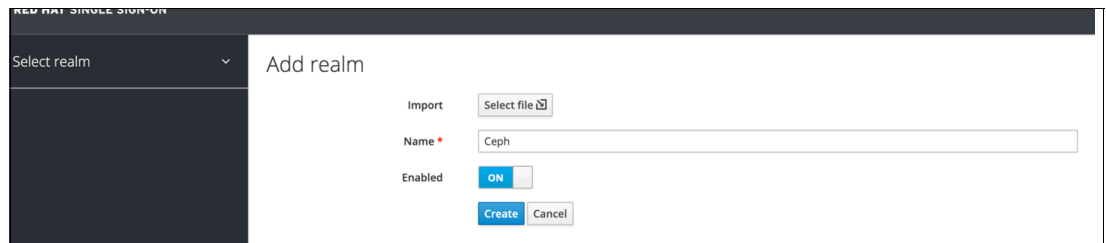


Figure 7-13 Add realm

- Go to **User Federation** from the left menu and add an LDAP provider. Example settings are shown in Figure 7-14.

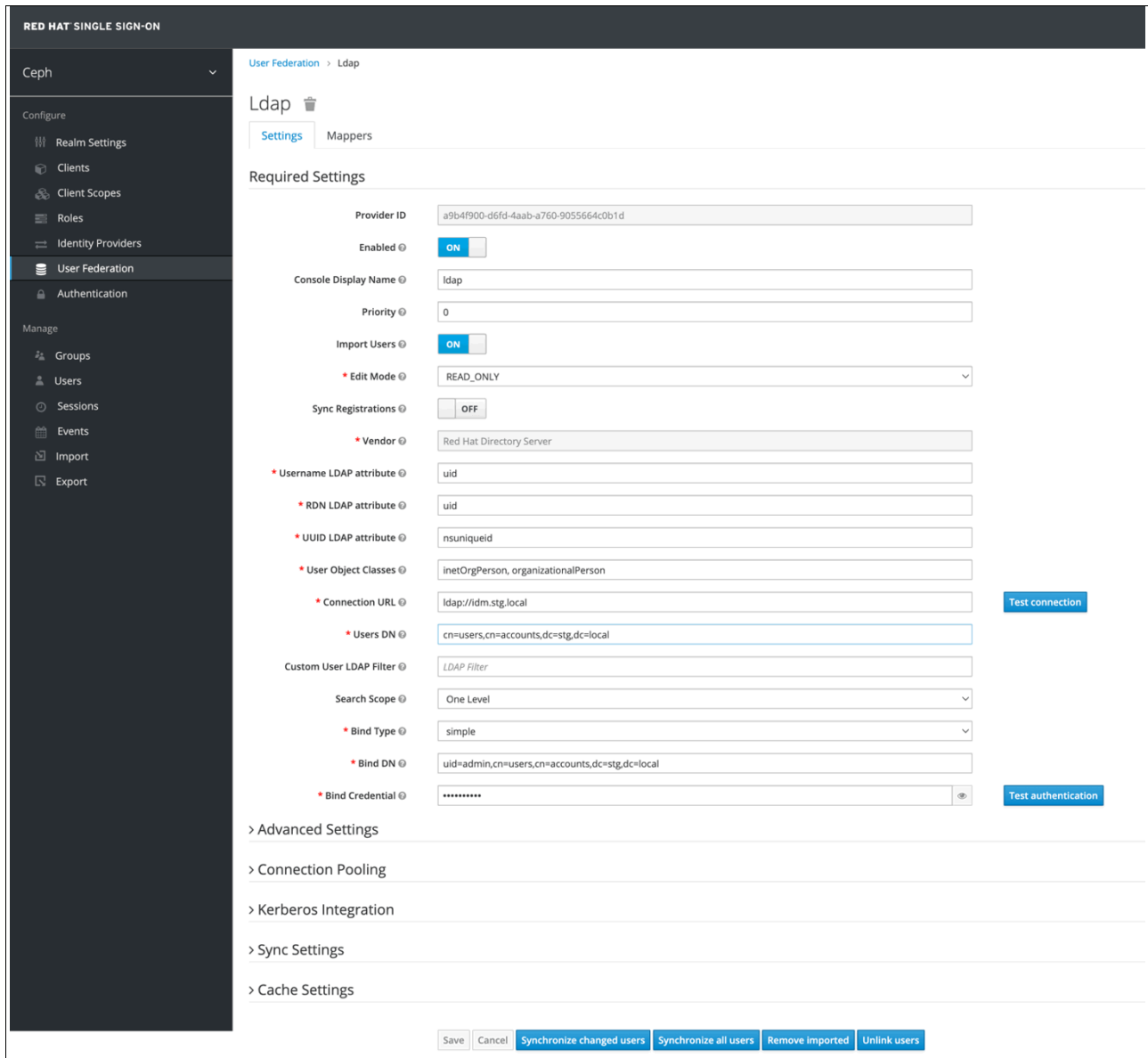


Figure 7-14 Adding an LDAP provider

- Enter the admin password into the Bind Credential field and test both connection and authentication separately. You do not need to change other settings below the Required Settings field.
- After successfully creating an LDAP provider, save the settings, synchronize the users, and go to the **Mappers** section. Create an LDAP Group mapper to map the users to their respective groups during the import process to Red Hat SSO.

5. Click **Create** to start adding your LDAP Group Mapper. You can find an example configuration in Figure 7-15.

The screenshot shows the configuration page for an LDAP Group Mapper in Red Hat Single Sign-On. The page is titled "Ldapgroups" and is part of the "User Federation > Ldap > LDAP Mappers > Ldapgroups" navigation path. The configuration fields are as follows:

- ID:** 54d3f58d-c697-4488-81a7-a8cbf585d474
- Name:** ldapgroups
- Mapper Type:** group-ldap-mapper
- LDAP Groups DN:** cn=groups,cn=accounts,dc=stg,dc=local
- Group Name LDAP Attribute:** cn
- Group Object Classes:** groupOfNames
- Preserve Group Inheritance:** ON
- Ignore Missing Groups:** OFF
- Membership LDAP Attribute:** member
- Membership Attribute Type:** DN
- Membership User LDAP Attribute:** uid
- LDAP Filter:** (empty)
- Mode:** READ\_ONLY
- User Groups Retrieve Strategy:** LOAD\_GROUPS\_BY\_MEMBER\_ATTRIBUTE
- Member-Of LDAP Attribute:**memberOf
- Mapped Group Attributes:** (empty)
- Drop non-existing groups during sync:** OFF
- Groups Path:** /

At the bottom of the form, there are four buttons: "Save", "Cancel", "Sync LDAP Groups To Keycloak", and "Sync Keycloak Groups To LDAP".

Figure 7-15 Adding the LDAP Group Mapper

6. Save your settings and click **Sync LDAP Groups to Keycloak**. In this step, if the top message says "0 imported groups", you can change the Mode from READ\_ONLY to LDAP\_ONLY and revert this setting to READ\_ONLY after the import is complete.

You can validate the imported users from the Users section on the left menu (Figure 7-16).

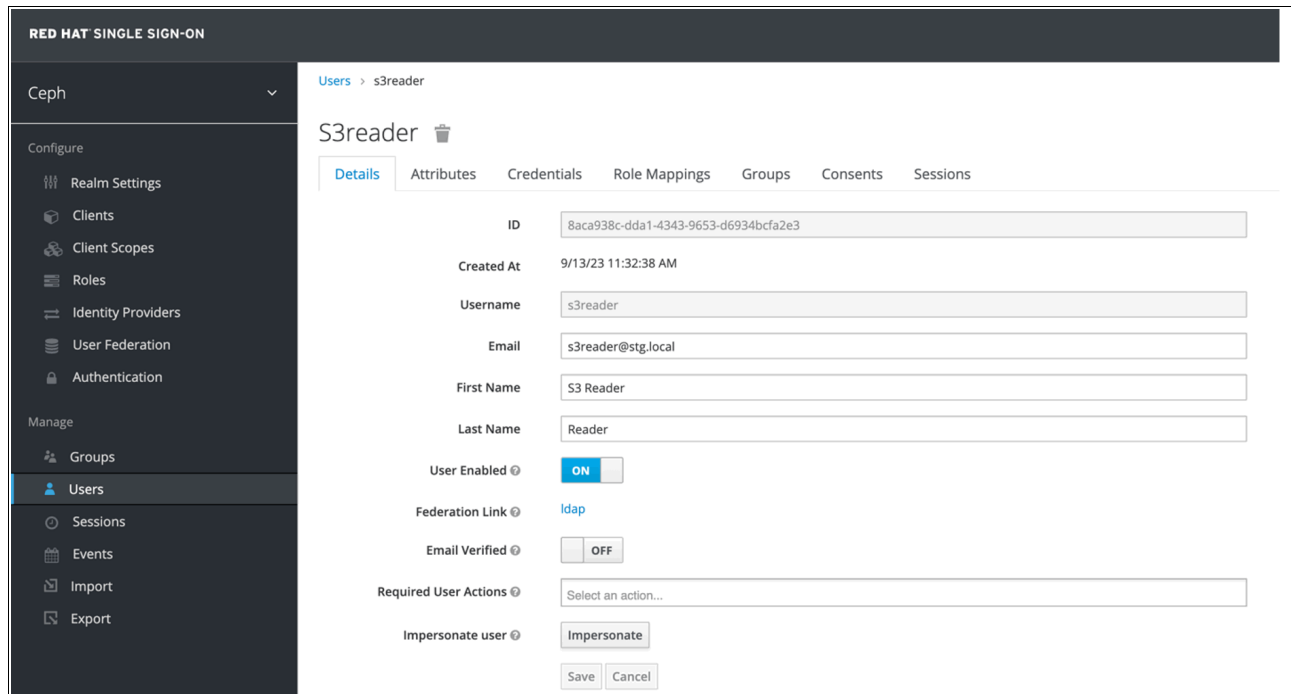


Figure 7-16 Validating the imported users

## 7.2.8 Client configuration

Add a client to the Red Hat SSO (Keycloak) to allow connections from the IBM Storage Ceph instance. **Clients** are entities that can make requests to Keycloak for user authentication. As a best practice, use separate clients for each service or application that uses Keycloak for authentication. Add the client name to the RGW by completing the following steps.

1. To create a client on Red Hat SSO (Keycloak), go to the Clients section from the left menu and click **Create**. In this example, we use `openid-connect` as the Client Protocol (Figure 7-17).

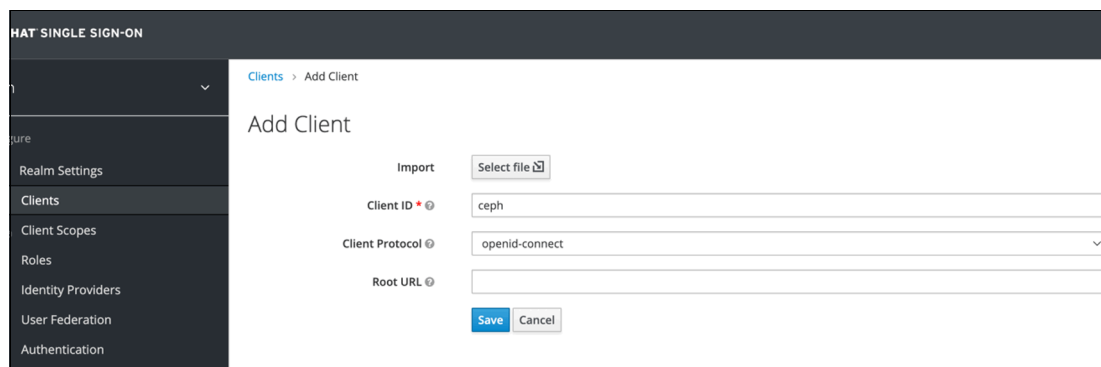


Figure 7-17 Add Client

- An example Ceph client configuration is shown in Figure 7-18. Save the settings after you enter the required values and go to the **Credentials** tab. Note the client's secret because this key is used to authenticate the Ceph client.

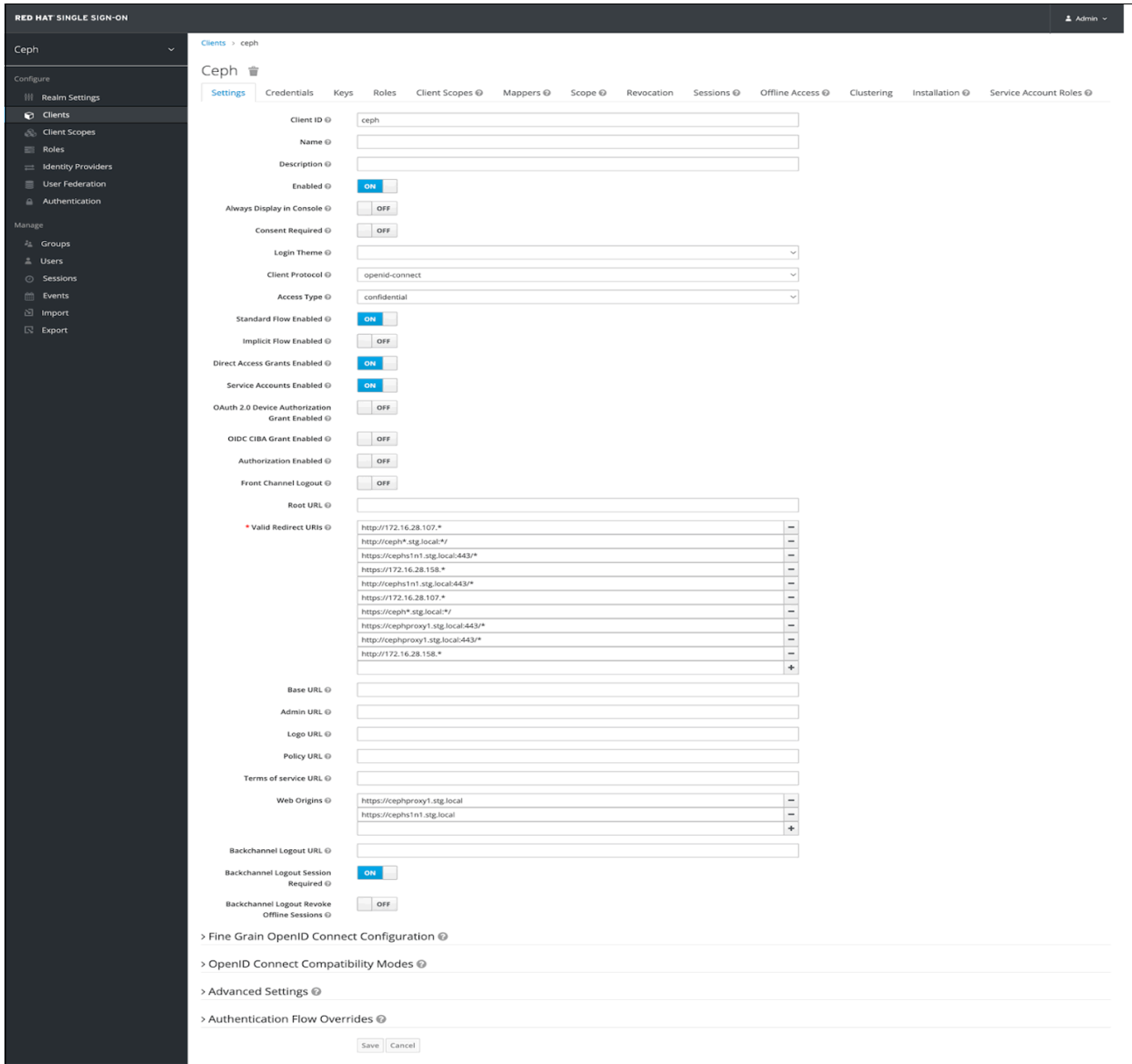


Figure 7-18 An example Ceph client configuration



You can always retrieve this value from the window that is shown in Figure 7-19.

The screenshot shows the 'Ceph' client configuration page in the AWS IAM console. The breadcrumb trail is 'Clients > ceph'. The page title is 'Ceph' with a trash icon. There are several tabs: 'Settings', 'Credentials', 'Keys', 'Roles', 'Client Scopes', 'Mappers', 'Scope', 'Revocation', and 'Sessions'. The 'Credentials' tab is active. Under 'Client Authenticator', a dropdown menu is set to 'Client Id and Secret'. Below it is a 'Secret' field with a 'Regenerate Secret' button. Under 'Registration access token', there is a text input field and a 'Regenerate registration access token' button.

Figure 7-19 Ceph client secret

Create Ceph client mappers of the types User Property, Group Membership, and Audience. These values are included in the token that is received from Red Hat SSO (Keycloak) during authentication and verification.

3. Create a User Property type of mapper that is named username. An example configuration is shown in Figure 7-20.

The screenshot shows the 'Username' mapper configuration page in the AWS IAM console. The breadcrumb trail is 'Clients > ceph > Mappers > username'. The page title is 'Username' with a trash icon. The configuration fields are: 'Protocol' (openid-connect), 'ID' (1818af08-231f-4cc1-a785-f31d6e37a947), 'Name' (username), 'Mapper Type' (User Property), 'Property' (username), 'Token Claim Name' (sub), 'Claim JSON Type' (String), 'Add to ID token' (ON), 'Add to access token' (ON), and 'Add to userinfo' (ON). There are 'Save' and 'Cancel' buttons at the bottom.

Figure 7-20 An example configuration: Username

4. Create a Group Membership type of mapper that is named ceph-groups. An example configuration is shown in Figure 7-21.

The screenshot shows the configuration page for a mapper named 'Ceph-groups'. The page has a title 'Ceph-groups' with a trash icon. Below the title are several configuration fields:

- Protocol: openid-connect
- ID: aeecb85d-2c9d-4ef9-b15a-092d98ddb36f
- Name: ceph-groups
- Mapper Type: Group Membership
- Token Claim Name: groups
- Full group path: OFF
- Add to ID token: ON
- Add to access token: ON
- Add to userinfo: ON

At the bottom of the form are 'Save' and 'Cancel' buttons.

Figure 7-21 An example configuration: Ceph-groups

5. Create an Audience type of mapper that is named ceph-mapper. An example configuration is shown in Figure 7-22.

The screenshot shows the configuration page for a mapper named 'Ceph-mapper'. The breadcrumb navigation at the top reads 'Clients > ceph > Mappers > ceph-mapper'. The page has a title 'Ceph-mapper' with a trash icon. Below the title are several configuration fields:

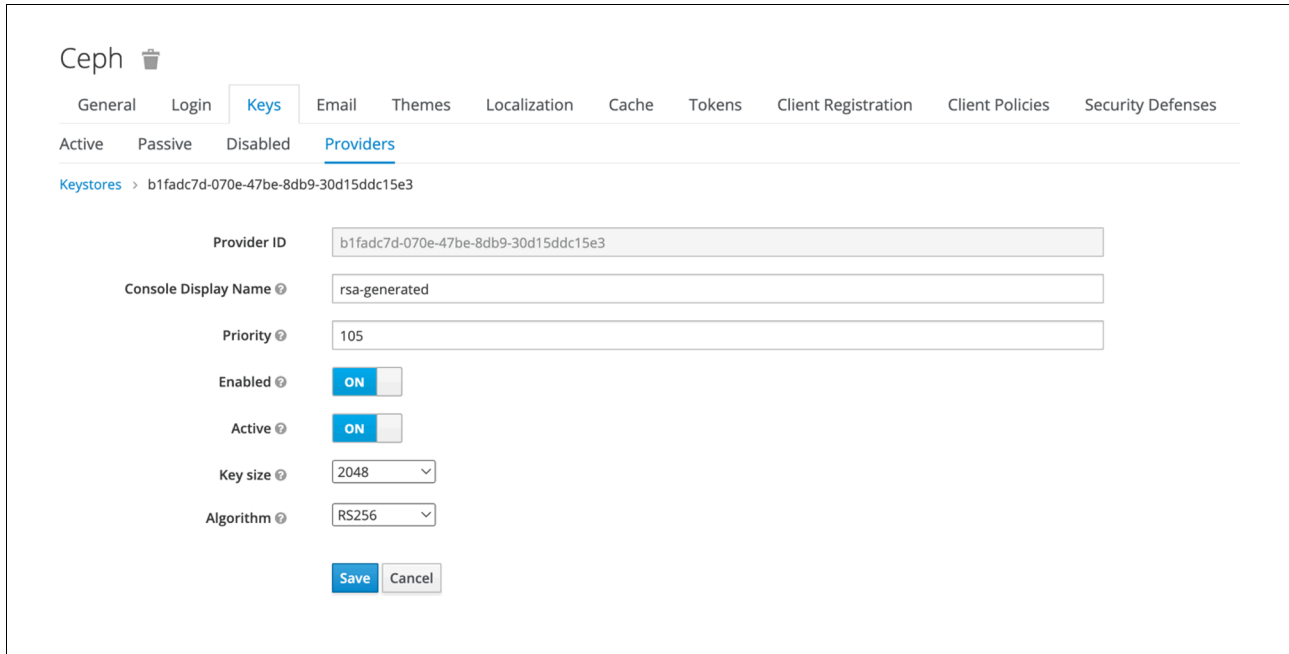
- Protocol: openid-connect
- ID: e8403170-a16d-4473-9580-3c3ed0730120
- Name: ceph-mapper
- Mapper Type: Audience
- Included Client Audience: ceph (with a dropdown arrow and an 'x' icon)
- Included Custom Audience: (empty text box)
- Add to ID token: OFF
- Add to access token: ON

At the bottom of the form are 'Save' and 'Cancel' buttons.

Figure 7-22 An example configuration: Ceph-mapper

## 7.2.9 Configuring the Ceph Realm settings

After successfully importing users, mapper configurations, and the Ceph client, update the Ceph realm settings for the RSA-generated provider to a higher priority so that the signing key for the JWT is presented earlier than others. To do this task, go to Realm Settings on the Keycloak interface and click **rsa-generated** on the provider column. Update the Priority value to 105 and click **Save** (Figure 7-23).



The screenshot shows the Keycloak console interface for configuring a provider. The breadcrumb path is "Keystores > b1fad7d-070e-47be-8db9-30d15ddc15e3". The provider name is "rsa-generated". The configuration fields are:

- Provider ID: b1fad7d-070e-47be-8db9-30d15ddc15e3
- Console Display Name: rsa-generated
- Priority: 105
- Enabled: ON
- Active: ON
- Key size: 2048
- Algorithm: RS256

Buttons for "Save" and "Cancel" are visible at the bottom.

Figure 7-23 Configuring the Ceph Realm settings

## 7.2.10 Verifying the Keycloak configuration: JWT token retrieval and validation

After successfully implementing Red Hat SSO and IdM, you can retrieve a JWT token from Keycloak and verify it by using the user IDs that you created on IdM.

Complete the following steps:

1. Retrieve the token from Keycloak. Example 7-7 shows an example script for retrieving a token with a specific user.

*Example 7-7 Retrieving the token from Keycloak*

```
[root@cephs1n1 ~]# cat get_web_token.sh
curl -k -q -L -X POST
"https://keycloak-ss0.apps.ocp.stg.local/auth/realm/ceph/protocol/openid-connect/
token" \
-H 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'client_id=ceph' \
--data-urlencode 'grant_type=password' \
--data-urlencode 'client_secret=7sQXqyMSzHIeMcSALoKa1jB6sNIBDRjU' \
--data-urlencode 'scope=openid' \
--data-urlencode "username=$1" \
--data-urlencode "password=$2"
```



4. An example output of running this script by using one of the users is shown in Example 7-10.

*Example 7-10 Example output of running the script*

---

```
[root@cephs1n1 ~]# ./check_token.sh s3reader passw0rd
{
  "exp": 1695110521,
  "iat": 1695110221,
  "jti": "abe1688c-bd0b-4c5f-af45-36b8490fdcb1",
  "iss": "https://keycloak-sso.apps.ocp.stg.local/auth/realms/ceph",
  "aud": [
    "ceph",
    "account"
  ],
  "sub": "s3reader",
  "typ": "Bearer",
  "azp": "ceph",
  "session_state": "5bb81c10-3b3a-47dc-9790-6a2acc5565e3",
  "name": "S3 Reader Reader",
  "given_name": "S3 Reader",
  "family_name": "Reader",
  "preferred_username": "s3reader",
  "email": "s3reader@stg.local",
  "email_verified": false,
  "acr": "1",
  "allowed-origins": [
    "https://cephproxy1.stg.local",
    "https://cephs1n1.stg.local"
  ],
  "realm_access": {
    "roles": [
      "default-roles-ceph",
      "offline_access",
      "uma_authorization"
    ]
  },
  "resource_access": {
    "account": {
      "roles": [
        "manage-account",
        "manage-account-links",
        "view-profile"
      ]
    }
  },
  "scope": "openid email profile",
  "sid": "5bb81c10-3b3a-47dc-9790-6a2acc5565e3",
  "groups": [
    "ipausers",
    "rgwreaders"
  ],
  "client_id": "ceph",
  "username": "s3reader",
  "active": true
}
```

---

You should be able to see both the user (s3reader) and the groups (rgwreader) that the user belongs to in the output.

## 7.3 IBM Storage Ceph STS configuration

This section covers the IBM Storage Ceph Object Storage (RGW) STS configuration. We integrate the SSO provider (Keycloak) by using the OIDC open authentication protocol with the RGW STS feature, so RGW S3 users can authenticate against the configured SSO to access S3 resources.

### 7.3.1 Registering an OpenID Connect provider on IBM Storage Ceph

Because you completed the Red Hat SSO (Keycloak) integration with IdM, you must register your OIDC provider to IBM Storage Ceph.

Complete the following steps:

1. Create a local user on IBM Storage Ceph RGW and assign the correct capabilities to it with the **caps** option.

To create a local user on RGW, run the following command:

```
[root@cephs1n1 ~]# radosgw-admin user create --uid oidc
--display-name "Open ID Connect Provider" --access-key oidc --secret oidc
```

**Note:** The user's access and secret keys should be changed for production environments.

2. Assign the admin capabilities to this user by using the following command:  

```
[root@cephs1n1 ~]# radosgw-admin caps add --uid="oidc" --caps="oidc-provider=*
```
3. Obtain the thumbprint for the x5c certificate. Use this thumbprint to register the OIDC client on RGW, as described in 7.3.4, "Registering the OIDC provider with RGW" on page 150.

An example script for retrieving the x5c thumbprint is shown in Example 7-11.

*Example 7-11 Retrieving the x5c thumbprint*

---

```
[root@cephs1n1 ~]# cat x5c.sh
#!/bin/bash
RHSSO_REALM="https://keycloak-ss0.apps.ocp.stg.local/auth/realms/ceph"
set -o pipefail
CERT_FILE=$(mktemp)
cat << EOF > ${CERT_FILE}
----BEGIN CERTIFICATE-----
$(curl -k ${RHSSO_REALM}/protocol/openid-connect/certs | jq -r .keys[0].x5c[])
----END CERTIFICATE-----
EOF
openssl x509 -in ${CERT_FILE} -fingerprint -noout | awk -F = '{ print $NF }' | sed
's://g'
rm -f ${CERT_FILE}
[root@cephs1n1 ~]# ./x5c.sh
A616EE704C3DA89B6187FCF430153334C3A11D75
```

---

**Note:** The script that is shown in Example 7-11 captures the first key in the array:

```
".keys[0].x5c[]"
```

Depending on your Keycloak configuration, you might have more than one key in the output of the `curl certs` command:

```
curl -k ${RHSSO_REALM}/protocol/openid-connect/certs
```

If you have multiple keys, use the key with the `use: "sig"` parameter, which is the key/certificate that is used to sign JWT tokens, and the one that you must get the thumbprint from for configuring RGW. Example 7-12 shows an example output. (This example is not an actual step to take; it is just an example of the warning.)

#### Example 7-12 Script output

```
# curl -k ${RHSSO_REALM}/protocol/openid-connect/certs | jq .
{
  "keys": [
    {
      "kid": "LeuppLq90y1gfQ1GHdgG9B7iTQ51fD4DGCA-58hrKns",
      "kty": "RSA",
      "alg": "RSA-OAEP",
      "use": "enc", <----- Not this one!
      "n":
"Xhwk5ordbkSNRftiSAD-JYEq-g7KFykJMsn40PyWqWgPtINxVhLGzrAohYi5Sk2VTkohpgCkyWE63eWqP
GqnttNxVSKub00j_IQ4PvzxDQb1yPiBc4cnsQ_b5m07ih0MfG_-nM_qU_kaiVz48ifzNRaCi0fK6H3Yi6u
5-vK70tLRT0Ycxwz4dw416QrkNTF8PKK98Qu_-5_-0yh5HAyGS6mMXhB0Eseye1wUF2KCXAwwQ5HMi-QAU1
xiT1bLD7uPToMAOnqvkoZHG8ZzRJS-X5QAGgXKMJ39mIBBkLjo27U0UtKzPK0QYeZxytQiN1PRrFI0PhsI
M1mm6ivb4vQ9U9w",
      "e": "AQAB",
      "x5c": [
"MIIC1zCCAX8CBgGKdRY7ozANBgqhkiG9w0BAQsFADAPMQ0wCwYDVQQDDARjZXBoMB4XDTIzMDkwODEz
TYONVoXDTMzMDkwODEzNTgyNVowDzENMAsGA1UEAwEY2VwaDCCASiWdQYJKoZIhvcNAQEBBQADggEPAD
CAQoCggEBAMycJ0aK3W5EjUX7YkgA/iWBKvo0yhpcCTLJ+ND8lq1oD7SDcVYSxs6wKIWIuUpN1U5KIaYAp
M1h0t3lqjxqp7bTcVUirmztI/yEOD788Q0G9cj4gX0HJ7EP2+Zt04odDHxv/pzP6lP5Go1c+PIn8zUWgot
Hyuh92IurufryuzrS0UzmHMcM+HcONekK5DUxfDyivfELv/uf/tMoeRwMhkupjF4QdBLMntcFBdig1wMME
ORzIvkaFJcYk5Wyw+7j06DANJ6r5KGRxvGc0SUv1+UABoFyjCd/ZiAQZC46Nu1NFE5GaStEGHmccrUIjZT
0axSND4bCDNZpuor2+LOPVPcCAwEAATANBgqhkiG9w0BAQsFAAOCQAQAwK2iM7r30IYjjC7XMHteeg1Ef
1Dsu+f4zCRVR6YNlqDmU+UzWZAsEGu+cVxydvfNA3jEzGbs8USOnfw09grX6tDUA5dWaqmIQbQX1hLBjz7
AjlWUv0gh4Dy7r/JSq1ZmDV8kqDxotwJbmcGDI8Hm97U9tN3D2aj0Ympc3i1b1F7qYK/1jkIOxQ4iHZfon
ij85/975LosUf/7ScceNkZnfrTr8sbw9UDuUP/fXmTBvWf27/tjgd00K8VBdVmq9rNjYwtXUHuYC1nX7q/
OX/GyENne08rj6W+1n18DKZm0wccvnrjcwH/tkeW309Pckop+NE/2rR4w2GjXFj9tCESIg=="
      ],
      "x5t": "A0nP1pfgsW3RPIaw_9wplX5dJN8",
      "x5t#S256": "LWJy0RSrq_VCKsY1tMZsvUmPFWUuxZFC4a11ULZ2X8k"
    },
    {
      "kid": "KQuLnre0heQKsc_BkAbW1aUvesGbMcpZw1mDHT4kSVQ",
      "kty": "RSA",
      "alg": "RS256",
      "use": "sig", <-- This is the correct x5c key/cert!
      "n":
"uJZsgG51iWUm_rLtkYgjDAIr8cew_7-aU1m7-XBd3vtNt6DGRPSHQ59BBd1cpt4mg4zAan7x4RUL8ed-
nvNb0cph8DS5VLG10N6CXm7N6FCpPx-eB_ssCjGFiyzLR9cxE3QuDUK3r0_AeEarn9mDw_fkv2edXxFLXC
```

```

x01iAI0bSFrAb69iq33LiZh73smhQDC8zHnmL1Gs2x2UhjB2_Vfa79-rGHZLSVMoLXBn-hFLTU3Td-auZX
rJHF5rFVKTrgsv1s1a5Ek_hY8Yvhh1D9BzK0gvnp1otIeU42W-gApS4CFG96cJEN4AGYJJW01ScrBnUQ8
6e9HVb0K_eB3Nfw",
  "e": "AQAB",
  "x5c": [

"MIIC1zCCAX8CBgGkDRY6YjANBgkqhkiG9w0BAQsFADAPMQ0wCwYDVQQDDARjZXBoMB4XDTIzMDkwODEzN
TYONVoXDTMzMDkwODEzNTgyNVowDzENMA5GA1UEAwEY2VwaDCCASiWdQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBALiWbIBudY11Jv6y7ZGIiwCK/HS+P+/m1JZu/1wXd77TbegxkT0oak+fQQXdXKbeJo0MwGp+
8eEVC/Hnfp7zW9HKYfA0uVSxtTjeg15uzehQqT8fngf7LAoxhSmsy0fXMRNOLg1Ct6zvwHhGq5/Zg8P35L
9nnV8RS1wsTtYgCNG0hawG+vYqt9y4mYe97JoUAWmX55i5RrNsd1IYwdv1X2u/fqxh2S7FTK1wZ/oRS0
1N03fmrMv6yRxeaxVSk64LL5ebNWuRJP4WPGL4YZQ/QcytIL56daLSH1ON1voAKUuAhRvenCRDeABmCSVj
pUnKwZ1EP0nvR1W9Cv3gdzX8CAwEAATANBgkqhkiG9w0BAQsFAA0CAQEAAAD2FAPmVBTVI4S6Eo0gH8ktbe
AXu9odebGG14gAWZ00Qdu0eC4pdSemxENXN+0doNFpJJLs2VTXasHORI04Lx8Nw2P58so1GZec2uFS/Fpt
wk964eDwRcCt3bYbKKEmomsCPdjCCQDs/V3c1WRz3Z1pE+eqKZXn0s0r/9JEU8wcNDUeM2baXDACmvCtWH
tIxyFcyooLzBkEmHxuvKoa92C8VmuIJAIZbN8Qjym6TBSotCz9Ybzb99Q/EnrOccuXrEMI/Z1nwURp1miA
LgBu/BZ84bmU2Kz07vcc0zdIoMi1NYfw4QVECIzk9ohzzwo2LWa9cfLxYf06qtj3pNYUkjQ=="
  ],
  "x5t": "phbucEw9qJthh_zOMBuzNMOhHXU",
  "x5t#S256": "L1z7h1B3aWyuSnyi_wpq2GGH917u552a_FD3sW-Y99s"
}
]
}

```

### 7.3.2 Configuring certificates

Configure IBM Storage Ceph Object Storage (RGW) to use HTTPS/SSL. Terminate SSL at the load balancer/Ingress service. With this configuration, the communications between the S3 clients and the load balancer are encrypted, but the communications from the load balancer to the RGWs are in the clear. In this example, we use self-signed certificates, but you can also use trusted CA-signed SSL certificates.

Create an SSL certificate on the RGW host by completing the following steps:

1. Create a `certs` directory under `/root` and create the configuration file for the certificate with Subject Alternate Names. You can use the certificate with the domains that are defined in the Subject Alternate Name area. An example of a configuration file is shown in Example 7-13.

*Example 7-13 Creating a `certs` directory under `/root` and creating the configuration file*

```

[root@cephproxy1 certs]# cat cert.cnf
[req]
distinguished_name = req_distinguished_name
x509_extensions = x509
prompt = no

[req_distinguished_name]
countryName = TR
stateOrProvinceName = Istanbul
organizationName = IBM
organizationalUnitName = STG
commonName = cephproxy1.stg.local

[x509]

```



```
keyUsage = critical, digitalSignature, keyAgreement
extendedKeyUsage = serverAuth
subjectAltName = @sans
```

```
[sans]
DNS.1 = *.stg.local
DNS.2 = cephproxy1.stg.local
```

---

2. Create a key for the certificate (Example 7-14).

*Example 7-14 Creating a key for the certificate*

---

```
[root@cephproxy1 certs]# openssl genrsa 2048 > cert.key
[root@cephproxy1 certs]# chmod 400 cert.key
```

---

3. Create your certificate file by running the command that is shown in Example 7-15.

*Example 7-15 Creating the certificate file*

---

```
[root@cephproxy1 certs]# openssl req -x509 -nodes -days 730 -newkey rsa:2048
-keyout cert.key -out cert.pem -config cert.cnf -sha256
```

---

4. Verify the certificate SANs by running the command that is shown in Example 7-16.

*Example 7-16 Verifying the certificate SANs*

---

```
[root@cephproxy1 certs]# openssl x509 -in /root/certs/cert.pem -noout -text | grep
-A1 'Subject Alternative Name'
      X509v3 Subject Alternative Name:
        DNS:*stg.local, DNS:cephproxy1.stg.local
```

---

5. After successful creation, the certificate file should be added to the trusted certificates on the workstation node, which is the first ceph node.

To do this task, transfer both the `cert.pem` and `cert.key` files to `/etc/pki/ca-trust/source/anchors` directory under `cephs1n1.stg.local` host by using `scp` (Example 7-17).

*Example 7-17 Transferring the cert.pem and cert.key files*

---

```
[root@cephproxy1 certs]# scp cert.pem cert.key
root@cephs1n1.stg.local:/etc/pki/ca-trust/source/anchors
```

---

6. After the transfer, import the certificate into the system. The commands that are shown in Example 7-18 can be used to import the certificate to the system and update the `ca-bundle`.

*Example 7-18 Importing the certificate into the system*

---

```
[root@cephs1n1 ~]# update-ca-trust
[root@cephs1n1 ~]# update-ca-trust enable
[root@cephs1n1 ~]# update-ca-trust extract
```

---

### 7.3.3 Configuring RGW in highly available mode with load balancing and SSL

With the ingress service, you can create a highly available (HA) endpoint for RGW with minimal configuration. The orchestrator deploys and manages a combination of haproxy and keepalived to balance the load on a floating virtual IP address. The ingress service is deployed on N hosts, each of which has a haproxy daemon and a keepalived daemon. A virtual IP address is automatically configured on only one of these hosts at a time.

Every few seconds, each keepalived daemon checks whether the haproxy daemon on the same host is responding. Keepalived also checks that the master keepalived daemon is running without problems. If the master keepalived daemon or the active haproxy daemon is not responding, one of the remaining keepalived daemons running in backup mode is elected as master, and the virtual IP will be moved to that node.

The active haproxy acts like a load balancer, distributing all RGW requests between all available RGW daemons.

Figure 7-24 shows the ingress service.

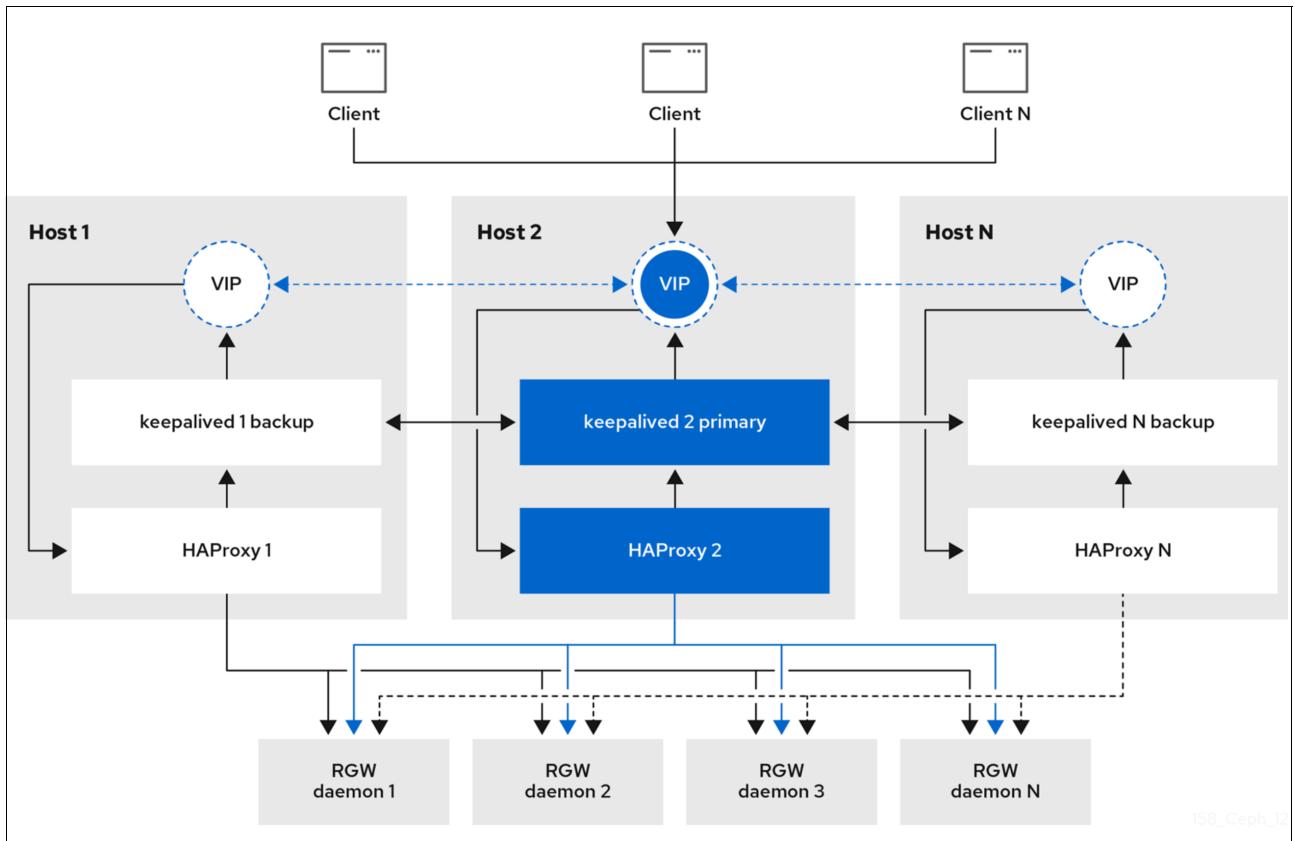


Figure 7-24 The ingress service

After a successful import of the certificate file, configure the RGW with SSL ingress by completing the following steps:

1. Label the RGW host with `rgws` to use it with the RGW configuration file. The command in Example 7-19 can label the `cephproxy1` node.

*Example 7-19 Labeling the cephproxy1 node*

---

```
[root@cephs1n1 ~]# ceph orch host label add cephproxy1.stg.local rgws
Added label rgws to host cephproxy1.stg.local
```

---

2. Verify the host label by running the command in Example 7-20.

*Example 7-20 Verifying the host label*

---

```
[root@cephs1n1 ~]# ceph orch host ls
HOST                ADDR                LABELS              STATUS
cephproxy1.stg.local 172.16.28.158      rgws
cephs1n1.stg.local  172.16.28.107      _admin osd mon
cephs1n2.stg.local  172.16.28.108      osd
cephs1n3.stg.local  172.16.28.109      mgr mon osd
cephs1n4.stg.local  172.16.28.110      osd
cephs1n5.stg.local  172.16.28.111      osd mon
6 hosts in cluster
```

---

3. After successfully labeling the host, configure an RGW service with SSL by using port 8443 (Example 7-21). To do this task, create a YAML file that includes the certificate and key file for the RGW. An example configuration file can be used to implement an RGW service with SSL ingress. This YAML file should be created on the RGW node and be deployed from there.

*Example 7-21 Configuring an RGW service with an SSL by using port 8443*

---

```
[root@cephproxy1 certs]# cat <<EOF >> /root/rgw-ssl.yaml
service_type: rgw
service_id: objectgw
service_name: rgw.objectgw
placement:
  count: 1
  label: rgws
spec:
  rgw_frontend_port: 8443
  rgw_frontend_type: beast
  rgw_frontend_ssl_certificate: |
    -----BEGIN CERTIFICATE-----
    $( cat /root/certs/certificate.pem | grep -v CERTIFICATE | awk '{ $1="    "$1 }' )
    -----END CERTIFICATE-----
    -----BEGIN RSA PRIVATE KEY-----
    $( cat /root/certs/certificate.key | grep -v PRIVATE | awk '{ $1="    "$1 }' )
    -----END RSA PRIVATE KEY-----
    ,
rgw_realm: multisite
rgw_zone: zone1
ssl: true
extra_container_args:
  - "-v"
  - "/etc/pki:/etc/pki/:z"
EOF
# ceph orch apply -i /root/rgw-ssl.yaml
```

---

The `/etc/pki` directory is bind-mounted into the container that the RGW service will be running in. This setup ensures that the RGW service always uses the latest updates of the certificates that are imported into the RGW host.

### 7.3.4 Registering the OIDC provider with RGW

Register the OIDC provider with RGW by completing the following steps:

1. Use the Python code in Example 7-22 to register the OIDC provider to RGW.

*Example 7-22 Python code to register the OIDC provider to RGW*

---

```
import boto3
import argparse
import re
parser = argparse.ArgumentParser(prog = 'OIDC provide registration',description =
"Adds or deletes OIDC providers in CEPH")
parser.add_argument("-o","--op",action="store",dest="op",default="add",choices=["a
dd","del"])
# Admin user with: radosgw-admin caps add --uid="admin" --caps="oidc-provider="
parser.add_argument("-k","--access-key",action="store",dest="AWS_ACCESS_KEY_ID",re
quired=True)
parser.add_argument("-s","--secret-key",action="store",dest="AWS_SECRET_ACCESS_KEY
",required=True)
###
parser.add_argument("-r","--rgw-endpoint",action="store",dest="RGW_ENDPOINT",requi
red=True)
parser.add_argument("-t","--oidc-thumbprint",action="store",dest="OIDC_THUMBPRINT"
,required=True)
parser.add_argument("-u","--oidc-url",action="store",dest="OIDC_URL",required=True
)
args = parser.parse_args()
OIDC_ARN = re.compile(r"https?://")
OIDC_ARN = "arn:aws:iam::oidc-provider/" +
OIDC_ARN.sub('',args.OIDC_URL).strip().strip('/')
iam_client = boto3.client(
    'iam',
    aws_access_key_id=args.AWS_ACCESS_KEY_ID,
    aws_secret_access_key=args.AWS_SECRET_ACCESS_KEY,
    endpoint_url=args.RGW_ENDPOINT,
    region_name='',
    verify='/etc/pki/tls/certs/ca-bundle.crt'
)
if args.op == "del":
    try:
        oidc_delete =
iam_client.delete_open_id_connect_provider(OpenIDConnectProviderArn=OIDC_ARN)
        print("Deleted OIDC provider")
    except:
        print("Provider already absent")
else:
    try:
        oidc_get = iam_client.list_open_id_connect_providers()
        print("Provider already registered.")
    except:
        oidc_create = iam_client.create_open_id_connect_provider(
```

```

        Url=args.OIDC_URL,
        ClientIDList=[
            "ceph",
        ],
        ThumbprintList=[
            args.OIDC_THUMBPRINT,
        ]
    )
    print("Provider Created")

```

---

2. Run the Python script with the arguments that are shown in Example 7-23.

*Example 7-23 Running the Python script*

---

```

[root@cephs1n1 ~]# python3 oidc_register.py -k oidc -s oidc \
-r https://cephproxy1.stg.local:8443 -t 00E9CFD697E0B16DD13C86B0FFDC29957E5D24DF \
-u https://keycloak-sso.apps.ocp.stg.local/auth/realms/ceph

```

---

3. After running the code successfully, verify that the OIDC provider was added to RGW by running the command that is shown in Example 7-24.

*Example 7-24 Verifying that the OIDC provider is added to the RGW*

---

```

[root@cephs1n1 ~]# aws --endpoint https://cephproxy1.stg.local:8443 iam
list-open-id-connect-providers
{
    "OpenIDConnectProviderList": [
        {
            "Arn":
"arn:aws:iam::oidc-provider/keycloak-sso.apps.ocp.stg.local/auth/realms/ceph"
        }
    ]
}

```

---

### 7.3.5 Role and role policy creation

Now, you assign roles and policies to these roles according to the groups. First, you must create the roles `rgwadmins`, `rgwwriters`, and `rgwreaders`.

#### **RGW Admins role**

Create a role for `rgwadmins` by completing the following steps:

1. Create a JSON file defining the role properties. This JSON policy document grants access to the `rgwadmins` role to any user who is authenticated by the SSO (OIDC provider) and part of the IDM/LDAP `rgwadmins` group.

The condition checks for a match on the JWT and enables the user to assume the role if it finds a StringLike rgwadmins value in the JWT groups section of the token (Example 7-25).

*Example 7-25 Creating a JSON file defining the role properties*

---

```
[root@cephs1n1 ~]# cat role-rgwadmins.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": [
          "arn:aws:iam:::oidc-provider/keycloak-sso.apps.ocp.stg.local/auth/realms/ceph"
        ]
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity"
      ],
      "Condition": {
        "StringLike": {
          "keycloak-sso.apps.ocp.stg.local/auth/realms/ceph:groups": [
            "/rgwadmins"
          ]
        }
      }
    }
  ]
}
```

---

2. Create an RGW role by using the JSON file (Example 7-26).

*Example 7-26 Creating an RGW role by using the JSON file*

---

```
[root@cephs1n1 ~]# radosgw-admin role create --role-name rgwadmins \
--assume-role-policy-doc=$(jq -rc . /root/role-rgwadmins.json)
{
  "RoleId": "668021ca-16a4-41ad-950d-85e8cc526811",
  "RoleName": "rgwadmins",
  "Path": "/",
  "Arn": "arn:aws:iam:::role/rgwadmins",
  "CreateDate": "2023-09-22T11:15:21.863Z",
  "MaxSessionDuration": 3600,
  "AssumeRolePolicyDocument":
  "{\n\"Version\": \"2012-10-17\", \"Statement\": [\n{\n\"Effect\": \"Allow\", \"Principal\": {\n\"Federated\": [\n\"arn:aws:iam:::oidc-provider/keycloak-sso.apps.ocp.stg.local/auth/realms/ceph\"]\n}, \"Action\": [\n\"sts:AssumeRoleWithWebIdentity\"], \"Condition\": {\n\"StringLike\": {\n\"keycloak-sso.apps.ocp.stg.local/auth/realms/ceph:groups\": [\n\"rgwadmins\"]\n}}\n}}\n}"
}
```

---

## RGW Writers role

Create a role for rgwwriters by completing the following steps:

1. Create a JSON file defining the role properties (Example 7-27).

### Example 7-27 Creating a role for rgwwriters

---

```
[root@cephas1n1 ~]# cat role-rgwwriters.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": [
          "arn:aws:iam:::oidc-provider/keycloak-sso.apps.ocp.stg.local/auth/realms/ceph"
        ]
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity"
      ],
      "Condition": {
        "StringLike": {
          "keycloak-sso.apps.ocp.stg.local/auth/realms/ceph:groups": "rgwwriters"
        }
      }
    }
  ]
}
```

---

2. Create a role for rgwwriters by using the JSON file (Example 7-28).

### Example 7-28 Creating a role for rgwwriters by using the JSON file

---

```
[root@cephas1n1 ~]# radosgw-admin role create --role-name rgwwriters \
> --assume-role-policy-doc=$(jq -rc . /root/role-rgwwriters.json)
{
  "RoleId": "f5bd0b47-516b-4a31-8c17-ea5928a84b6e",
  "RoleName": "rgwwriters",
  "Path": "/",
  "Arn": "arn:aws:iam:::role/rgwwriters",
  "CreateDate": "2023-10-11T19:01:56.995Z",
  "MaxSessionDuration": 3600,
  "AssumeRolePolicyDocument":
  "{\n\"Version\": \"2012-10-17\", \"Statement\": [\n{\n\"Effect\": \"Allow\", \"Principal\": {\n\"Federated\": [\n\"arn:aws:iam:::oidc-provider/keycloak-sso.apps.ocp.stg.local/auth/realms/ceph\"]},\n\"Action\": [\n\"sts:AssumeRoleWithWebIdentity\"],\n\"Condition\": {\n\"StringLike\": {\n\"keycloak-sso.apps.ocp.stg.local/auth/realms/ceph:groups\": \"rgwwriters\"}}}}]"
}
```

---

## RGW Readers role

Create a role for rgwreaders by completing the following steps:

1. Create a role for rgwreaders. This JSON policy document grants access to the rgwreaders role to any user who is authenticated by the SSO (OIDC provider) and part of the IDM/LDAP rgwreaders group. The condition checks for a match on the JWT and enables the user to assume the role if it finds a StringLike rgwreaders value in the JWT groups section of the token (Example 7-29).

### Example 7-29 Creating a role for rgwreaders

---

```
[root@cephas1n1 ~]# cat role-rgwreaders.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": [
          "arn:aws:iam::oidc-provider/keycloak-sso.apps.ocp.stg.local/auth/realms/ceph"
        ]
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity"
      ],
      "Condition": {
        "StringLike": {
          "keycloak-sso.apps.ocp.stg.local/auth/realms/ceph:groups": "rgwreaders"
        }
      }
    }
  ]
}
```

---

2. Create a role for rgwreaders by using the JSON file (Example 7-30).

### Example 7-30 Create a role for rgwreaders by using the JSON file

---

```
[root@cephas1n1 ~]# radosgw-admin role create --role-name rgwreaders \
--assume-role-policy-doc=$(jq -rc . /root/role-rgwreaders.json)
{
  "RoleId": "b088171c-ae9c-469b-9061-6ea943daffa0",
  "RoleName": "rgwreaders",
  "Path": "/",
  "Arn": "arn:aws:iam::role/rgwreaders",
  "CreateDate": "2023-10-11T14:43:58.483Z",
  "MaxSessionDuration": 3600,
  "AssumeRolePolicyDocument":
  "{\n\"Version\": \"2012-10-17\", \"Statement\": [\n{\n\"Effect\": \"Allow\", \"Principal\": {\n\"Federated\": [\n\"arn:aws:iam::oidc-provider/keycloak-sso.apps.ocp.stg.local/auth/r
ealms/ceph\" ]}, \"Action\": [\n\"sts:AssumeRoleWithWebIdentity\" ], \"Condition\": {\n\"Str
ingLike\": {\n\"keycloak-sso.apps.ocp.stg.local/auth/realms/ceph:groups\": \"rgwreader
s\" } } ] } ] }"
}
```

---



## 7.3.6 Attaching policies to roles

After successfully creating roles on RGW, attach policies to define the capabilities of users within those groups. To do so, complete the following steps:

1. Create a role policy for `rgwadmins`. In this policy, you enable the user that assumed the `rgwadmins` role to do any S3 action on any S3 resource (Example 7-31).

*Example 7-31 Creating a role policy for `rgwadmins`*

---

```
[root@cephs1n1 ~]# cat policy-rgwadmin.json
{
  "Version": "2012-10-17",
  "Statement": [
    { "Effect": "Allow",
      "Action": [ "s3:*" ],
      "Resource": "arn:aws:s3:::*"
    }
  ]
}
```

---

2. Apply the role policy for the `rgwadmins` role (Example 7-33).

*Example 7-32 Applying the role policy for the `rgwadmins` role*

---

```
[root@cephs1n1 ~]# radosgw-admin role policy put --role-name=rgwadmins
--policy-name=admin --policy-doc=$(jq -rc . /root/policy-rgwadmin.json)
Permission policy attached successfully
```

---

3. Create a role policy for `rgwwriters` (Example 7-33).

*Example 7-33 Creating a role policy for `rgwwriters`*

---

```
[root@cephs1n1 ~]# cat policy-rgwwriter.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:PutObjectAcl",
        "s3:PutObjectTagging"
      ],
      "Resource": "arn:aws:s3:::*"
    }
  ]
}
```

---

4. Attach the role policy for `rgwwriters` (Example 7-34).

*Example 7-34 Attaching the role policy for `rgwwriters`*

---

```
[root@cephs1n1 ~]# radosgw-admin role policy put --role-name=rgwwriters
--policy-name=writer --policy-doc=$(jq -rc . /root/policy-rgwwriter.json)
Permission policy attached successfully
```

---

5. Create a policy for rgwreaders (Example 7-35).

*Example 7-35 Creating a policy for rgwreaders*

---

```
[root@cephs1n1 ~]# cat policy-rgwreader.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:ListBucketMultipartUploads",
        "s3:GetObject"
      ],
      "Resource": "arn:aws:s3:::*"
    }
  ]
}
```

---

6. Apply the role policy for rgwreaders (Example 7-36).

*Example 7-36 Applying the role policy for rgwreaders*

---

```
[root@cephs1n1 ~]# radosgw-admin role policy put --role-name=rgwreaders \
--policy-name=reader --policy-doc=$(jq -rc . /root/policy-rgwreader.json)
Permission policy attached successfully
```

---

You can define multiple policies concurrently for a specific group. An example of a multiple-policy definition is shown in Example 7-37. For more information, see the [AWS documentation](#).

*Example 7-37 Multiple-policy definition*

---

```
[root@cephs1n1 ~]# cat policy-multiple.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:ListBucketMultipartUploads",
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::atestbucket"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListMultipartUploadParts"
      ],
      "Resource": [
        "arn:aws:s3:::stsbucket/*",
        "arn:aws:s3:::stsbucket"
      ]
    }
  ]
}
```

```

    ]
  },
  {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject",
      "s3:AbortMultipartUpload",
      "s3:ListMultipartUploadParts"
    ],
    "Resource": [
      "arn:aws:s3:::stsbucket/uploads",
      "arn:aws:s3:::stsbucket/uploads/*"
    ]
  },
  {
    "Effect": "Deny",
    "Action": "s3:*",
    "NotResource": [
      "arn:aws:s3:::stsbucket/*",
      "arn:aws:s3:::stsbucket"
    ]
  }
]
}
}

```

---

### 7.3.7 Enabling STS on the RGW service

Enable STS on the RGW service that you are going to use. To do so, use the commands that are shown in Example 7-38.

*Example 7-38 Enabling STS on the RGW service*

```

[root@cephs1n1 ~]# ceph config set client.rgw.objectrgw rgw_s3_auth_use_sts true
[root@cephs1n1 ~]# ceph config set client.rgw.objectrgw rgw_sts_key
passw0rd12345678

```

---

**Note:** The `rgw_sts_key` should be a 16-character alphanumeric value.

### 7.3.8 Importing a Keycloak certificate to an RGW node

Because Keycloak is using an SSL connection for requests and running on Red Hat OpenShift, you must import the certificate files of Red Hat OpenShift Ingress Service into your RGW node. There are many ways to retrieve the Keycloak certificate chain; the following steps describe one such way:

1. To import the Red Hat OpenShift certificate, open a Firefox browser and go to the Keycloak URL, which is `https://keycloak-sso.apps.ocp.stg.local/` in this case.
2. Click the **Advanced** option on the security warning and click **View Certificate** when the detailed description is shown, as shown in Figure 7-25.

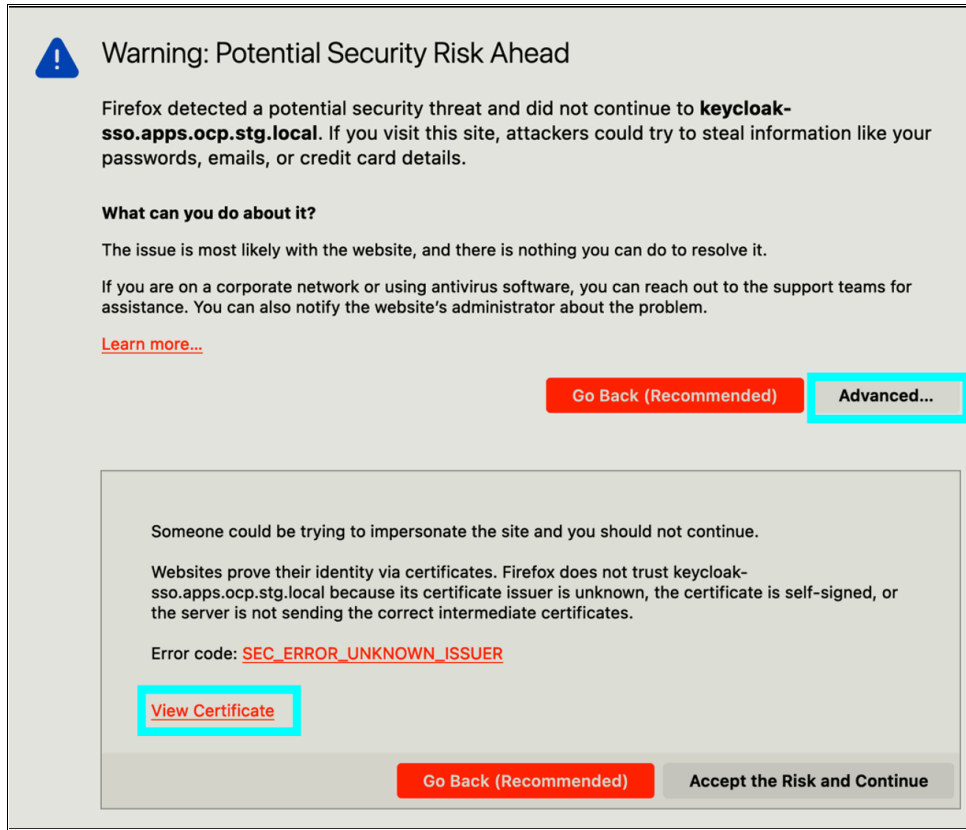


Figure 7-25 View Certificate

3. On the `*apps.ocp.stg.local` tab, go to the Miscellaneous section and download the chain certificate file (Figure 7-26).



Figure 7-26 Downloading the chain certificate file

4. Transfer the downloaded file into your RGW node's `/etc/pki/ca-trust/source/anchors` directory.

5. After the transfer, update your system's certificate bundle and restart the RGW service by running the commands that are shown in Example 7-39.

*Example 7-39 Updating your system's certificate bundle and restarting the RGW service*

---

```
[root@cephproxy1 anchors]# update-ca-trust
[root@cephproxy1 anchors]# update-ca-trust enable
[root@cephproxy1 anchors]# update-ca-trust extract
[root@cephproxy1 anchors]# ceph orch restart rgw.objectgw
```

---

6. Verify the certificate installation by doing a simple `curl` operation on the Keycloak main page (Example 7-40).

*Example 7-40 Verifying the certificate installation*

---

```
[root@cephproxy1 anchors]# curl https://keycloak-sso.apps.ocp.stg.local/
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">!DOCTYPE html
PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <meta http-equiv="refresh" content="0;url=/auth">
</head>
</html>
```

---

## 7.4 Testing Assume Role With Web Identity for role-based access control

After successfully importing the certificates into the system, you can test the Assume Role With Web Identity function for RBAC of your IBM Storage Ceph cluster.

An example script for the test is shown in Example 7-41. The `AWS_CA_BUNDLE` argument is pointing to the certificate that was created on the RGW node.

*Example 7-41 Example script for the test*

---

```
[root@cephs1n1 ~]# cat test-assume-role.sh
#!/bin/bash
export AWS_CA_BUNDLE="/etc/pki/ca-trust/source/anchors/cert.pem"
unset AWS_ACCESS_KEY_ID
unset AWS_SECRET_ACCESS_KEY
unset AWS_SESSION_TOKEN
KC_ACCESS_TOKEN=$(curl -k -q -L -X POST
"https://keycloak-sso.apps.ocp.stg.local/auth/realms/ceph/protocol/openid-connect/
token" \
-H 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'client_id=ceph' \
--data-urlencode 'grant_type=password' \
--data-urlencode 'client_secret=7sQXqyMSzHIeMcSALoKa1jB6sNIBDRjU' \
--data-urlencode 'scope=openid' \
--data-urlencode "username=$1" \
--data-urlencode "password=$2" | jq -r .access_token)
echo ${KC_ACCESS_TOKEN}
IDM_ASSUME_ROLE_CREDS=$(aws sts assume-role-with-web-identity --role-arn
"arn:aws:iam:::role/$3" --role-session-name testbr
```

```

--endpoint=https://cephproxy1.stg.local:8443
--web-identity-token="$KC_ACCESS_TOKEN")
echo "aws sts assume-role-with-web-identity --role-arn "arn:aws:iam::role/$3"
--role-session-name testb --endpoint=https://cephproxy1.stg.local:8443
--web-identity-token="$KC_ACCESS_TOKEN"
echo $IDM_ASSUME_ROLE_CREDS
export AWS_ACCESS_KEY_ID=$(echo $IDM_ASSUME_ROLE_CREDS | jq -r
.Credentials.AccessKeyId)
export AWS_SECRET_ACCESS_KEY=$(echo $IDM_ASSUME_ROLE_CREDS | jq -r
.Credentials.SecretAccessKey)
export AWS_SESSION_TOKEN=$(echo $IDM_ASSUME_ROLE_CREDS | jq -r
.Credentials.SessionToken)

```

---

## 7.4.1 Testing the Assume role with the Admin role

Running the example script in Example 7-42 with the admin role should give you the Admin role on buckets.

*Example 7-42 Running the example script with the admin role*

---

```

[root@cephs1n1 ~]# source ./test-assume-role.sh s3admin passwd rgwadmins
...Output omitted...
"SubjectFromWebIdentityToken": "s3admin", "AssumedRoleUser": { "Arn":
"arn:aws:sts::assumed-role/rgwadmins/testbr" }, "PackedPolicySize": 0,
"Provider": "https://keycloak-sso.apps.ocp.stg.local/auth/realms/ceph",
"Audience": "account" }

```

---

The **test-assume-role.sh** script that you ran in Example 7-41 on page 159 exports the three environment variables `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY`, and `AWS_SESSION_TOKEN`. These variables are used by the AWS CLI S3 client to authenticate against the S3 endpoint that is provided by the RGW.

Running the script with the `s3admin` user grants you admin privileges. You have full control over the buckets and can perform any actions. You can test the function by running the commands that are shown in Example 7-43.

*Example 7-43 Testing the function*

---

```

[root@cephs1n1 ~]# aws --endpoint https://cephproxy1.stg.local:8443 s3api
create-bucket --bucket=stsbucket

[root@cephs1n1 ~]# xfs_mkfile 5m 5mfile

[root@cephs1n1 ~]# aws --endpoint https://cephproxy1.stg.local:8443 s3 cp 5mfile
s3://stsbucket
upload: ./5mfile to s3://stsbucket/5mfile

[root@cephs1n1 ~]# aws --endpoint https://cephproxy1.stg.local:8443 s3 ls
s3://stsbucket
2023-10-05 16:53:01    5242880 5mfile
2023-10-05 10:56:32     3332 index.html

```

---

## 7.4.2 Testing the Assume role with the Writer role

If you run the script with the s3writer role, the writer role is assigned to you on buckets (Example 7-44).

*Example 7-44 Running the example script with the s3writer role*

---

```
[root@cephs1n1 ~]# source ./test-assume-role.sh s3writer passwd rgwwriters
...Output omitted...
"Expiration": "2023-10-12T15:10:36.135445+00:00" }, "SubjectFromWebIdentityToken":
"s3writer", "AssumedRoleUser": { "Arn":
"arn:aws:sts::assumed-role/rgwwriters/testbr" }, "PackedPolicySize": 0,
"Provider": "https://keycloak-sso.apps.ocp.stg.local/auth/realms/ceph",
"Audience": "account" }
```

---

With the writer role, you only write to the bucket. You can test the function by running the commands that are shown in Example 7-45.

*Example 7-45 Testing the function*

---

```
[root@cephs1n1 ~]# aws --endpoint https://cephproxy1.stg.local:8443 s3 mb
s3://testbucket
make_bucket failed: s3://testbucket An error occurred (AccessDenied) when calling
the CreateBucket operation: Unknown

[root@cephs1n1 ~]# xfs_mkfile 3m 3mfile

[root@cephs1n1 ~]# aws --endpoint https://cephproxy1.stg.local:8443 s3 cp 3mfile
s3://stsbucket/uploads
upload: ./3mfile to s3://stsbucket/uploads

[root@cephs1n1 ~]# aws --endpoint https://cephproxy1.stg.local:8443 s3 ls
s3://stsbucket
An error occurred (AccessDenied) when calling the ListObjectsV2 operation: Unknown
```

---

## 7.4.3 Testing the Assume role with the Reader role

If you run the script with the s3reader role, the reader role is assigned to you on buckets (Example 7-46).

*Example 7-46 Running the example script with the s3reader role*

---

```
[root@cephs1n1 ~]# source ./test-assume-role.sh s3reader passwd rgwreaders
...Output omitted...
"SubjectFromWebIdentityToken": "s3reader", "AssumedRoleUser": { "Arn":
"arn:aws:sts::assumed-role/rgwreaders/testbr" }, "PackedPolicySize": 0,
"Provider": "https://keycloak-sso.apps.ocp.stg.local/auth/realms/ceph",
"Audience": "account" }
```

---

With the reader role, you can only list the bucket contents and retrieve objects from it. You can test the function by running the commands that are shown in Example 7-47.

*Example 7-47 Testing the function*

---

```
[root@cephs1n1 ~]# aws --endpoint https://cephproxy1.stg.local:8443 s3 ls
s3://stsbucket
```

```
                PRE uploads/
2023-10-12 17:15:15 3145728 3mfile
2023-10-05 16:53:01 5242880 5mfile
2023-10-05 10:56:32   3332 index.html
```

```
[root@cephs1n1 ~]# aws --endpoint https://cephproxy1.stg.local:8443 s3 cp readfile
s3://stsbucket
```

```
upload failed: ./readfile to s3://stsbucket/readfile An error occurred
(AccessDenied) when calling the PutObject operation: Unknown
```

```
[root@cephs1n1 ~]# mkdir dwnld
```

```
[root@cephs1n1 ~]# aws --endpoint https://cephproxy1.stg.local:8443 s3 cp
s3://stsbucket/5mfile ./dwnld
```

```
download: s3://stsbucket/5mfile to dwnld/5mfile
```

---



# Abbreviations and acronyms

<b>ABAC</b>	attribute-based access control	<b>RPO</b>	recovery point objective
<b>ACK</b>	acknowledged	<b>RTO</b>	recovery time objective
<b>AMQP</b>	Advanced Message Queuing Protocol	<b>RTT</b>	round-trip time
<b>ARN</b>	Amazon Resource Name	<b>SRE</b>	Site Reliability Engineering
<b>CephFS</b>	Ceph File System	<b>STS</b>	Secure Token Service
<b>CIDR</b>	Classless Inter-Domain Routing	<b>VM</b>	virtual machine
<b>CR</b>	custom resource	<b>VMDK</b>	virtual machine disk
<b>DB</b>	database	<b>WAL</b>	write-ahead
<b>DR</b>	disaster recovery		
<b>FDF</b>	Fusion Data Foundation		
<b>GLB</b>	global load balancer		
<b>HA</b>	high availability or highly available		
<b>HADR</b>	high availability and disaster recovery		
<b>HMAC</b>	Hash-based Message Authentication Code		
<b>IAM</b>	Identity and Access Management		
<b>IBM</b>	International Business Machines Corporation		
<b>IdM</b>	Identity Management Server		
<b>IDP</b>	Identity Provider		
<b>IOPS</b>	input/output operations per second		
<b>ITIL</b>	IT Infrastructure Library		
<b>JWT</b>	JSON Web Token		
<b>KDC</b>	Kerberos Key Distribution Center		
<b>MCG</b>	Multicloud Object Gateway		
<b>ML</b>	machine learning		
<b>MDS</b>	MetaData Server		
<b>OBC</b>	Object Bucket Claim		
<b>OIDC</b>	OpenID Connect		
<b>OSaaS</b>	Object Storage as a Service		
<b>OSD</b>	object storage daemon		
<b>PG</b>	placement group		
<b>PMP</b>	Project Management Professional		
<b>PVC</b>	persistent volume claim		
<b>RBAC</b>	role-backed access control		
<b>RBD</b>	RADOS Block Device		
<b>RGW</b>	RADOS Gateway		
<b>RHACM</b>	Red Hat Advanced Cluster Management		
<b>RHEL</b>	Red Hat Enterprise Linux		



# Related publications

The publications that are listed in this section are considered suitable for a more detailed description of the topics that are covered in this paper.

## IBM Redbooks

The following IBM Redbooks publication provides more information about the topics in this document. It might be available in softcopy only.

*IBM Storage Ceph Concepts and Architecture Guide*, REDP-5721

You can search for, view, download, or order this document and other Redbooks, Redpapers, web docs, drafts, and additional materials at the following website:

[ibm.com/redbooks](https://ibm.com/redbooks)

## Other publications

These websites are also relevant as further information sources:

- ▶ IBM Storage Ceph Documentation:  
<https://www.ibm.com/docs/en/storage-ceph/6?topic=dashboard-monitoring-cluster>
- ▶ Community Ceph Documentation  
<https://docs.ceph.com/en/latest/monitoring/>

## Help from IBM

IBM Support and downloads

[ibm.com/support](https://ibm.com/support)

IBM Global Services

[ibm.com/services](https://ibm.com/services)







REDP-5715-00

ISBN 0738458279

Printed in U.S.A.

Get connected

