

# IBM Storage Ceph Solutions Guide

Daniel Parkes

Franck Malterre

Jean-Charles (JC) Lopez

Jussi Lehtinen

Kyle Bader

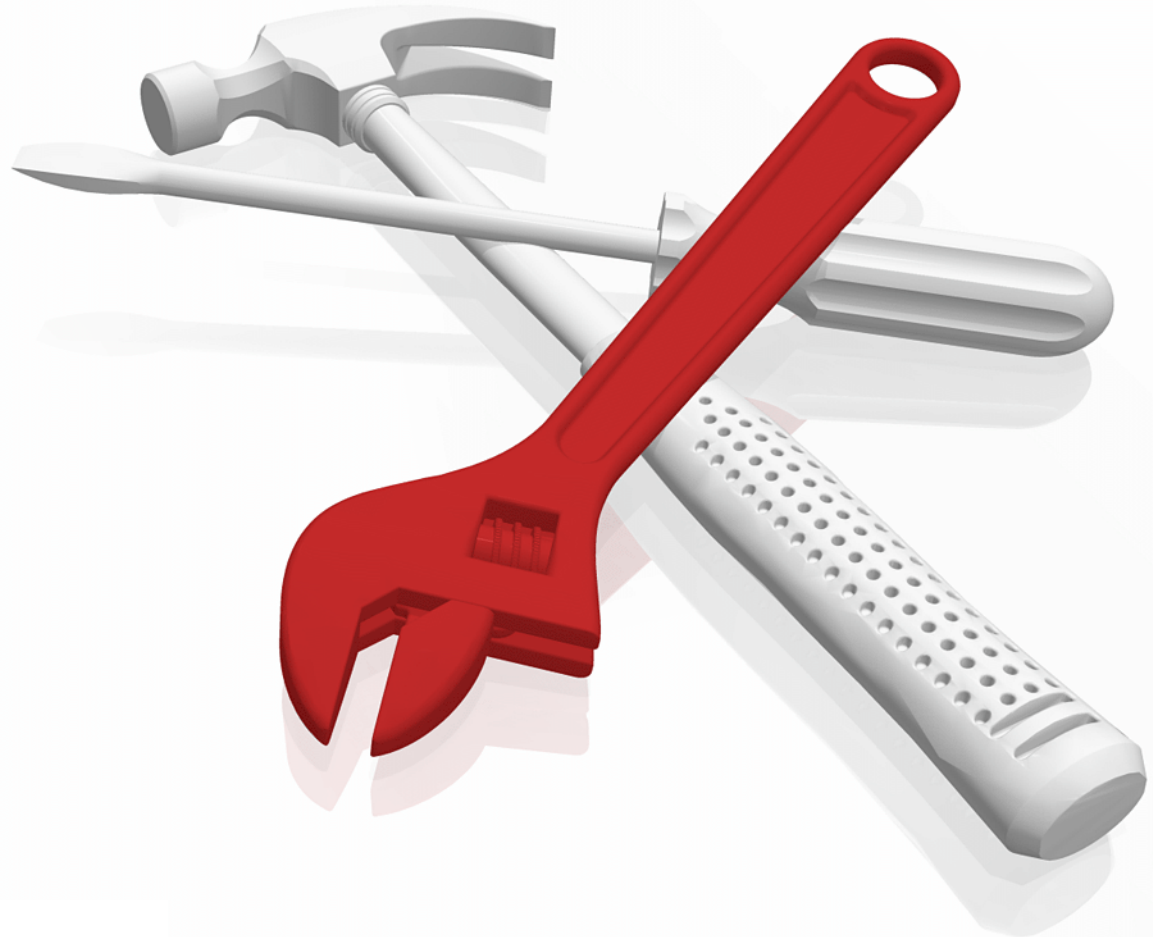
Poyraz Sagtekin

Reginald D'Souza

Suha Ondokuzmayis

Tan Long Siau

Vasfi Gucer







IBM Redbooks

**IBM Storage Ceph Solutions Guide**

August 2023

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

**First Edition (August 2023)**

This edition applies to IBM Storage Ceph Version 6.

This document was created or updated on November 28, 2023.

© Copyright International Business Machines Corporation 2023. 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 Connect to the cluster Dashboard .....	2
1.3 Expand the cluster .....	3
1.3.1 Adding hosts .....	3
1.3.2 Create OSDs .....	5
1.3.3 Create services .....	7
1.3.4 Expand the cluster .....	7
1.3.5 Checking configuration .....	8
1.4 Reduce the number of Monitors to 3 .....	9
1.4.1 Configure the web browser to access Grafana performance tabs .....	10
1.5 Configure RADOS Gateway (RGW) .....	13
1.5.1 Create rgw services and pools .....	13
1.5.2 Create ingress services .....	15
1.5.3 Create the data bucket pool using Erasure Coding .....	16
1.5.4 Create the user and its key .....	18
1.5.5 Put data in your bucket using AWS CLI .....	20
1.5.6 Checking pool status .....	21
1.6 Create RADOS Block Disk (RBD) .....	22
1.6.1 Create a RBD pool .....	22
1.7 Create an image .....	23
1.7.1 Map the created disk .....	24
1.7.2 Using snapshot .....	25
1.8 Create CephFS .....	28
1.9 Monitoring .....	32
1.9.1 Main Dashboard .....	32
1.9.2 Cluster menu .....	33
1.9.3 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 with the Ceph Dashboard .....	38
2.2.2 Creating user via the command line .....	41
2.3 Creating buckets .....	42
2.3.1 Creating bucket with the Ceph Dashboard .....	42
2.3.2 Creating bucket at the command line .....	43
2.4 Adding a bucket in watsonx.data .....	43
2.4.1 Associating a catalog .....	49
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	54
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>		57
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	60
3.3.1	Creating a user	60
3.3.2	Updating the quota	61
3.3.3	Usage and performance reporting	63
3.3.4	Documentation	64
<b>Chapter 4. Disaster recovery and backup and archive: IBM Storage Ceph as an S3 Backup Target</b>		67
4.1	Introduction	68
4.1.1	Benefits of IBM Storage Ceph for Veritas NetBackup	68
4.1.2	Scalability and flexibility	68
4.1.3	Data protection and disaster recovery	68
4.1.4	Efficiency and cost effectiveness	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	Configuration of Veritas NetBackup components	71
4.4	IBM Storage Ready Nodes for Ceph	75
4.4.1	IBM Storage Ceph Editions	77
<b>Chapter 5. S3 bucket notifications for event-driven architectures</b>		79
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, powered by the S3 bucket notification feature	83
5.4.1	Automated chest X-ray analysis for pneumonia detection	83
5.5	Step by step basic example of 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 OpenShift using the S3 RadosGW object storage provided by Fusion Data Foundation	89
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	93
<b>Chapter 6.</b>	<b>IBM Storage Fusion disaster recovery</b>	<b>97</b>
6.1	Introduction to Fusion Data Foundation	98
6.1.1	Introduction to Fusion Disaster Recovery (DR)	98
6.1.2	RegionalDR overview	100
6.1.3	MetroDR overview	101
6.1.4	OpenShift DR with stretched cluster	102
6.2	IBM Storage Ceph stretch mode	103
6.2.1	Stretched cluster pitfalls we avoid with Ceph stretched mode	104
6.2.2	Stretched mode architecture layout	105
6.2.3	Deployment example	107
6.2.4	Configuring Ceph in stretched mode	110
6.3	Fusion MetroDR for RPO Zero	114
6.3.1	Architecture - External versus internal Fusion Data Foundation deployments	114
6.3.2	MetroDR solution components	114
6.3.3	MetroDR architecture and layout	115
6.3.4	MetroDR application Failover and Failback workflow	117
6.3.5	MetroDR deployment example	117
<b>Chapter 7.</b>	<b>S3 enterprise user authentication (IDP) and authorization (RBAC)</b>	<b>119</b>
7.1	Introduction to RadosGW Auth methods	120
7.1.1	Introduction to RadosGW Secure Token Service	120
7.1.2	Introduction to STS with OIDC (SSO) provider	121
7.1.3	Introduction to IAM role policies	122
7.2	Step-by-step deployment guide: S3 enterprise user authentication (IDP) and authorization (RBAC)	124
7.2.1	Installation of RH SSO (Keycloak) on OpenShift	124
7.2.2	Installation of 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 Ceph Realm settings	140
7.2.10	Verify KC configuration: JWT token retrieval and validation	141
7.3	IBM Storage Ceph STS configuration	143
7.3.1	Registering OpenID Connect (OIDC) provider on IBM Storage Ceph	144
7.3.2	Configuring certificates	146
7.3.3	Configuring RGW in HA mode with load balancing and SSL	147
7.3.4	Register the OIDC provider with RadosGW	149
7.3.5	Role and role policy creation	151
7.3.6	Attaching policies to roles	154
7.3.7	Enabling STS on the RadosGW service	156
7.3.8	Importing Keycloak certificate to RGW node	157
7.4	Testing Assume Role With Web Identity for role based access control	158

7.4.1 Testing Assume role with Admin role .....	159
7.4.2 Testing Assume role with Writer role .....	160
7.4.3 Testing Assume role with Reader role .....	160
<b>Related publications</b> .....	<b>163</b>
IBM Redbooks .....	163
Other publications .....	163
Online resources .....	163
Help from IBM .....	163



# 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.

Redbooks (logo) ®  
IBM®

IBM Cloud®  
Redbooks®

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.

Veritas, the Veritas logo, Backup Exec and NetBackup are trademarks or registered trademarks of Veritas Technologies LLC or its affiliates in the U.S. and other countries. Other names may be trademarks of their respective owners.

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 operationalize AI with enterprise resiliency and 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 and includes storage analytics for critical insights into growing amounts of data. IBM Storage Ceph can be used as an easy and efficient way to build a data lakehouse for IBM watsonx.data and for next-generation AI workloads.

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

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 .



**Daniel Parkes** has been a die-hard Infrastructure enthusiast for many years with a huge passion for open-source technologies and a keen eye for innovation. Daniel is working in the IBM Storage Ceph Product Management team, focusing on the IBM Storage Ceph Object Storage offering. Always eager to get involved in Ceph technical discussions, helping our customers to succeed with Ceph on their journey to digital excellence.



**Franck Malterre** is an information technology professional with a background of over 25 years designing, implementing and maintaining large x86 physical and virtualized environments. Last 10 years, he specialized on IBM File and Object Storage Solution, developing IBM Cloud® Object Storage, IBM Storage Scale and IBM Storage Ceph live demonstrations and proof of concept for IBMers and Business Partners.



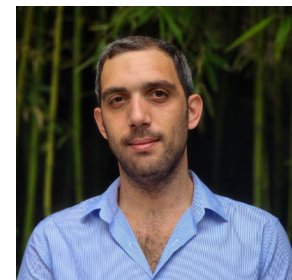
**Jean-Charles (JC) Lopez** has been in storage for 8/10th of his professional career and doing Software Defined Storage since 2013. JC really likes to think he can help people understand how they can move to a containerized environment when it comes to data persistence and protection. His go-to solutions are Fusion, Ceph and OpenShift. He has been practicing both the upstream community and downstream versions of the two latter ones.



**Jussi Lehtinen** is a Principal Storage SME for IBM Object Storage working for IBM Infrastructure in Nordics, 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.



**Kyle Bader** a Senior Technical Staff Member and Principal Portfolio Architect for Ceph based offerings at IBM. He has spent a decade architecting Ceph based storage clusters and solutions. His current interests lie at the intersection of distributed storage, containers, data intensive applications, and machine learning. Outside of tech he enjoys spending time with his wonderful wife and three children.



**Poyraz Sagtekin** joined IBM in 2016 after completion of his Computer Engineering degree from Middle East Technical University. Since then, he has developed expertise on diverse areas including IoT systems, IBM Cloud, Kubernetes, RedHat Enterprise Linux, and IBM Power Systems. Currently, he's working as a Hybrid Cloud Services Consultant in IBM Systems Expert Labs in Türkiye and he is an IBM recognized speaker and educator.



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



**Suha Ondokuzmayis** is a cloud-native and open source enthusiast, lifelong learner, and passionate technologist. He began his career in 2001 as an assembler developer intern at Turkish Airlines, followed by Turkcell, Turkish Telekom, IBM and Veritas Technologies within different departments, all with infrastructure related responsibilities. He joined İşbank in 2013 where he has led IT infrastructure, IaaS and PaaS platforms, and block, file, and object storage product offerings, as well as backup operations. He holds a Master of Science in Management Information Systems from the University of Istanbul.



**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, as well as container storage. He is a seasoned technologist with Big Data, AI, IoT, and HPC infrastructure architectural experience.



**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.

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

**Elias Luna, Kenneth David Hartsoe, William West, Henry Vo**  
IBM USA

Also thanks to **James Pool, James Eckersall** and **Ian Watson** from Red Hat Inc, as they started this journey, providing the initial insight and documentation for Chapter 7, “S3 enterprise user authentication (IDP) and authorization (RBAC)” on page 119 in this document.

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](https://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](https://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 graphical user interface called *Dashboard*. This dashboard can ease deployment, management, or monitoring. In this chapter we will have a look at these features.

This chapter has the following sections:

- ▶ “Introduction” on page 2
- ▶ “Connect to the cluster Dashboard” on page 2
- ▶ “Expand the cluster” on page 3
- ▶ “Reduce the number of Monitors to 3” on page 9
- ▶ “Configure RADOS Gateway (RGW)” on page 13
- ▶ “Create RADOS Block Disk (RBD)” on page 22
- ▶ “Create an image” on page 23
- ▶ “Create CephFS” on page 28
- ▶ “Monitoring” on page 32

## 1.1 Introduction

Before using the IBM Ceph Dashboard, we have to install some prerequisites that are already documented in [IBM Storage Ceph documentation](#).

We will start here just after the Ceph **bootstrap** command that 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 [video](#) that describes how to install IBM Storage Ceph using the command line.

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

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

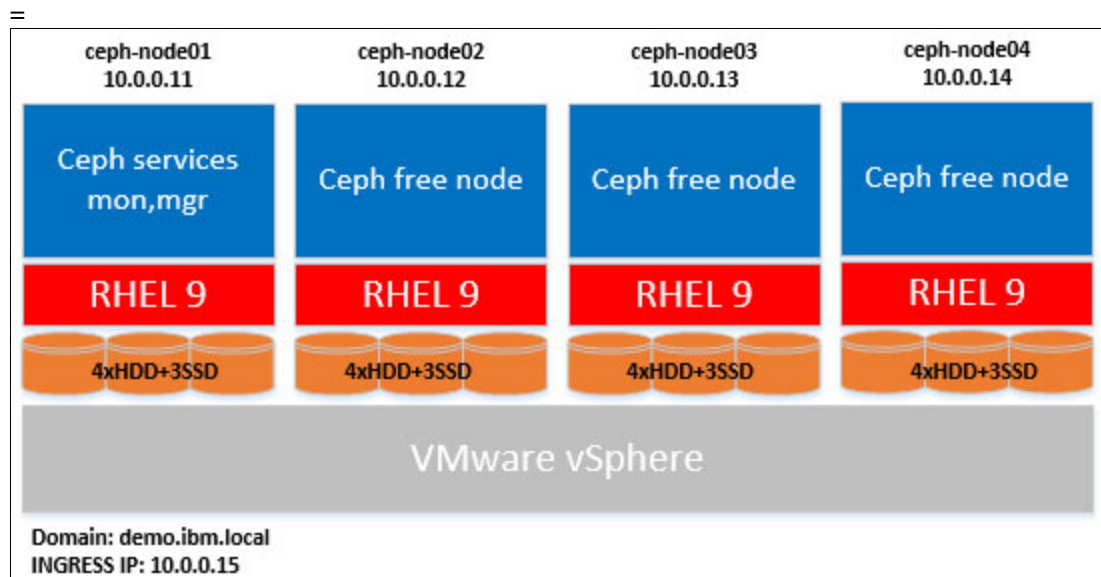


Figure 1-1 Cluster nodes

The bootstrap node is ceph-node01, each node is connected to 4 HDD disks and 3 SSD disks that will be used for OSD.

## 1.2 Connect to the cluster Dashboard

Connecting to the URL and password provided at the end of the bootstrap process, we will be able to login as admin on the Dashboard after setting a new password, you will be landed on the Expand Cluster page. See Figure 1-2 on page 3.



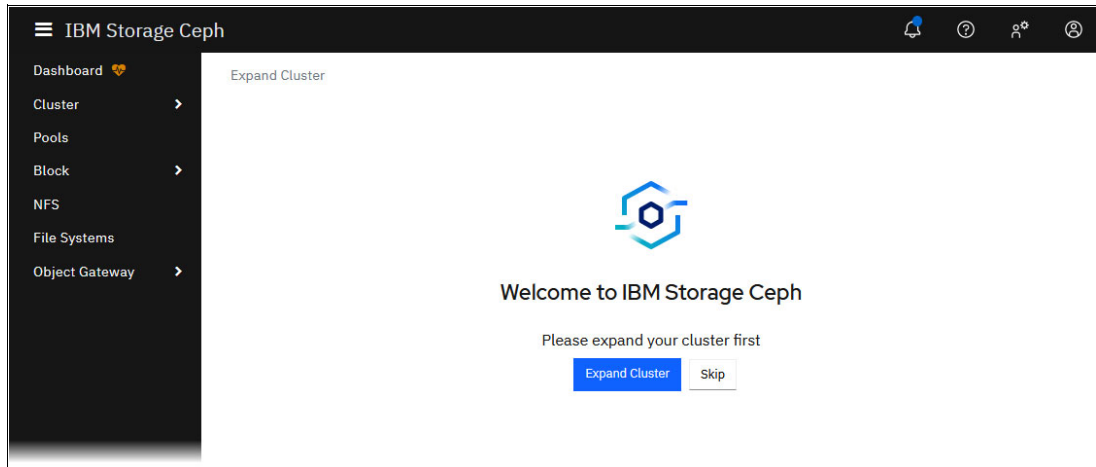


Figure 1-2 Dashboard landing page

## 1.3 Expand the cluster

In this chapter will walk you through the steps to expand the cluster.

### 1.3.1 Adding hosts

Perform the following steps to add hosts.

1. Click the **Expand Cluster** button to launch the assistant. See Figure 1-3

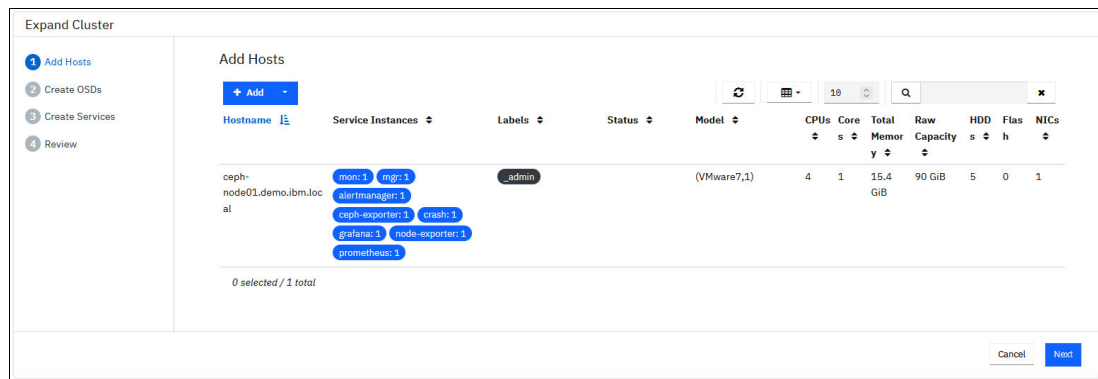


Figure 1-3 Add hosts step

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

We can choose to add one host at a time or multiple hosts at the same time. In our configuration hosts from 02 to 04 gives `ceph-node[02-04].demo.ibm.local`. When adding hosts, it is also convenient to select labels to apply. Labels are used to automatically deploy the corresponding daemon on the host. See Figure 1-4 on page 4.

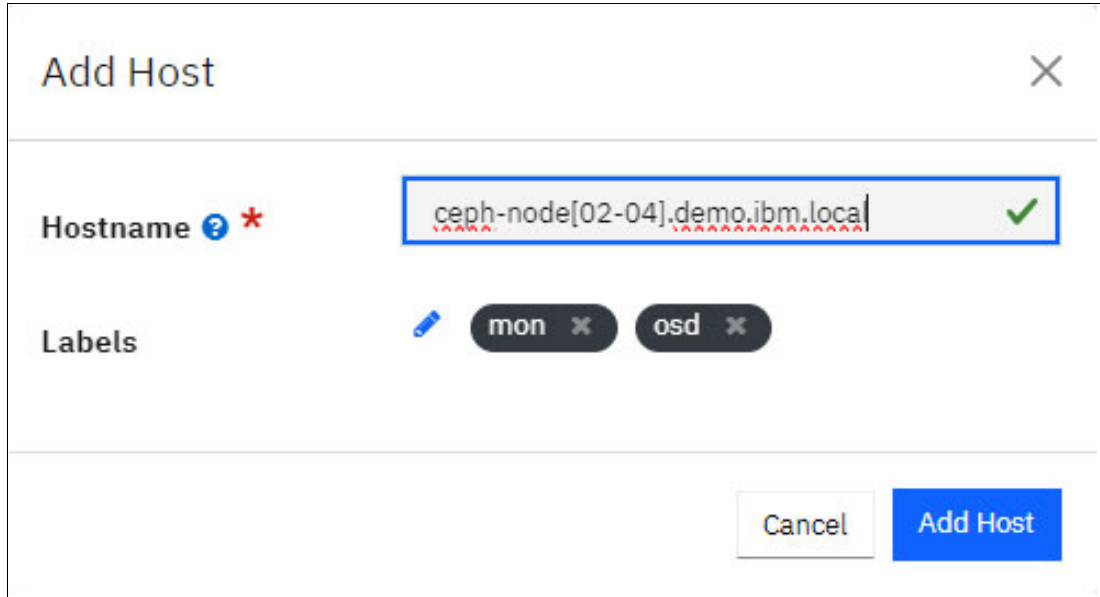


Figure 1-4 Adding multiple hosts at once with labels

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

Adding multiple hosts at once is faster but does not allow to specify different labels on each host. On the other hand, adding one host at a time allows to select different labels but can become cumbersome on large configuration. Here, every host runs the monitor and the osd daemon so we can select these labels.

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

Table 1-1 Daemon dispatch

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

3. We have to edit ceph-node01 to add osd label, ceph-node02 to add mgr label, ceph-node03 and ceph-node04 to add rgw label. We also need to move grafana daemon from ceph-node01 to ceph-node04. See Figure 1-5 on page 5.

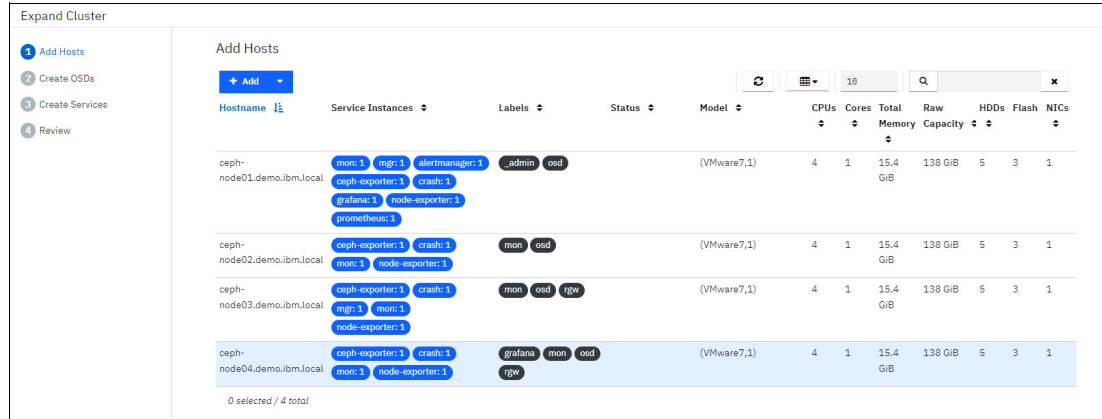


Figure 1-5 Hosts Labels

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

### 1.3.2 Create OSDs

Depending on the disks available, we can have the choice between different deployment options.

The expand cluster assistant allows to either select 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 WAL and DB.
- ▶ IOPS-optimized which uses NVME drive to store data.

In our example, we will use HDD drives as "Primary devices" and SSD drives as "DB to support BlueStore to increase performance.

1. We first select the **Advanced** mode. See Figure 1-6.

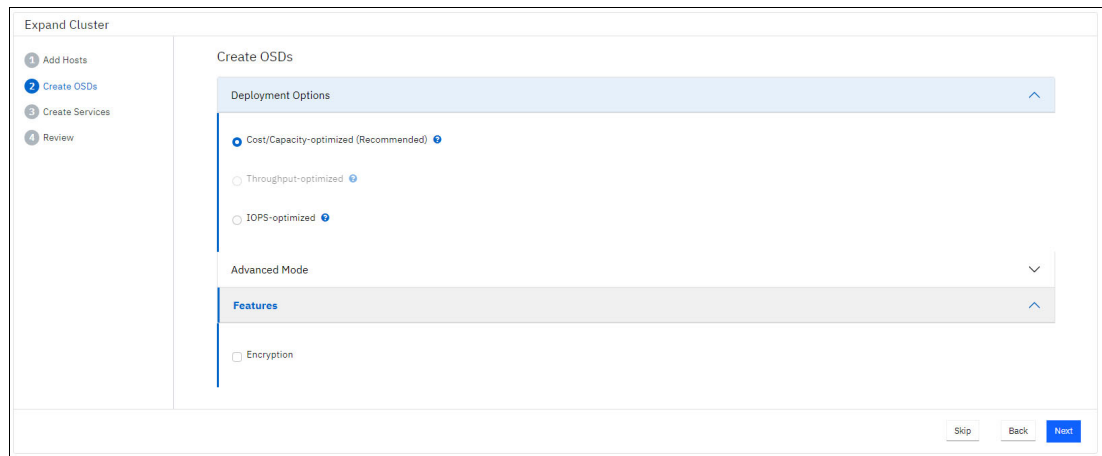


Figure 1-6 Create OSDs

- Then, we click **Add Primary devices**. Using the filters on top right, we can choose to only display hdd type. Once HDD devices are selected, we can click **Add** button. See Figure 1-7 on page 6.

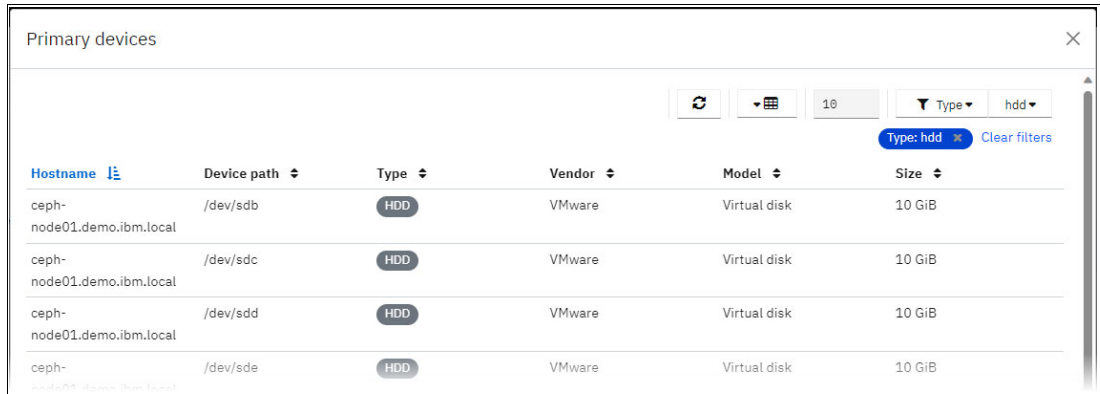


Figure 1-7 Add Primary devices

- Once Primary devices are selected, we can add WAL devices and DB devices. Here we choose to use ssd devices for DB devices only. See Figure 1-8 on page 6.

**Note:** Since 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.

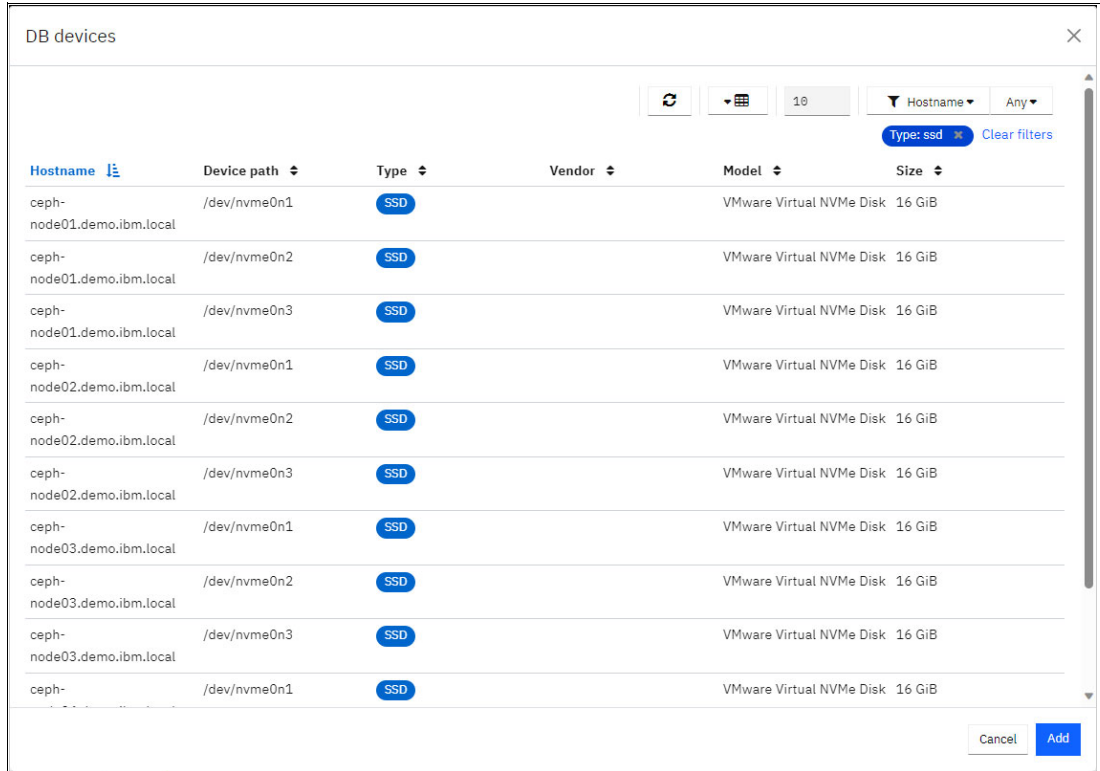


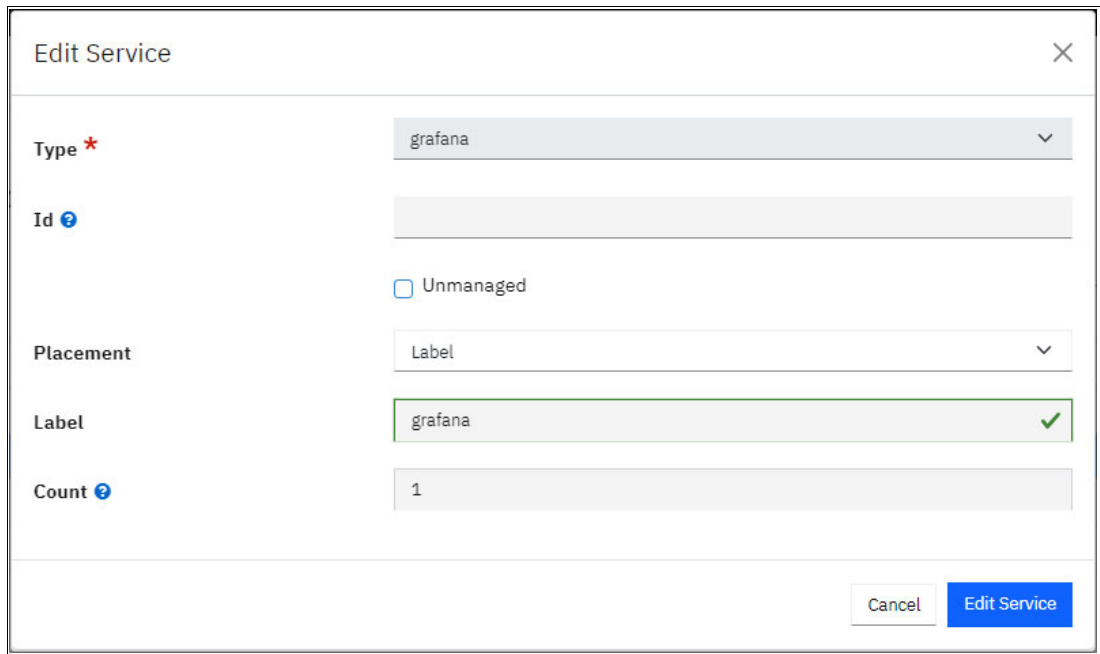
Figure 1-8 Add DB devices

### 1.3.3 Create services

This step allows to create or edit services, in this document, for more clarity, we will create the services in next sections.

But as an example, we can edit the Grafana service to move it to ceph-node04 to free up some resources from ceph-node01.

1. We select **grafana** and select **Edit** in the drop-down list. See 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:

- Type \***: A dropdown menu with "grafana" selected.
- Id**: An empty text input field.
- Unmanaged**: A checkbox that is currently unchecked.
- Placement**: A dropdown menu with "Label" selected.
- Label**: A dropdown menu with "grafana" selected, which is highlighted with a green border and a green checkmark on the right.
- 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 grafana

2. Once done, we can review the configuration and then launch the configuration of the cluster.

### 1.3.4 Expand the cluster

These steps will launch the deployment and configuration of daemons on the cluster hosts. We can follow this process, by clicking on the **bell icon** on the upper right of the page. See Figure 1-10 on page 8.

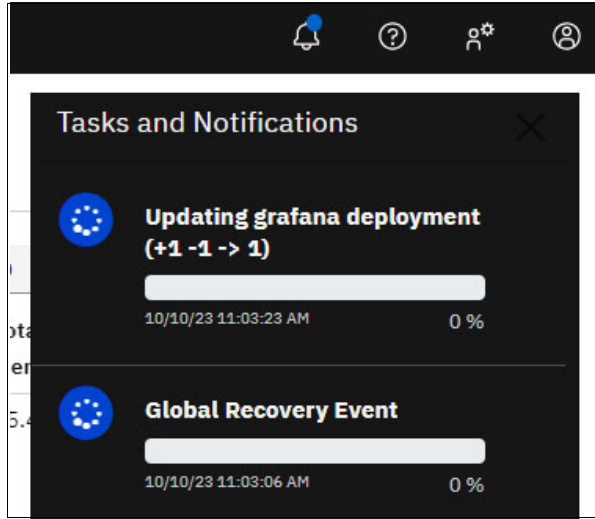


Figure 1-10 Follow the deployment

### 1.3.5 Checking configuration

Services deployment and OSDs creation takes a few minutes. We can check their status from the Cluster menu.

Hosts should display the four nodes and their instanced services. See Figure 1-11.

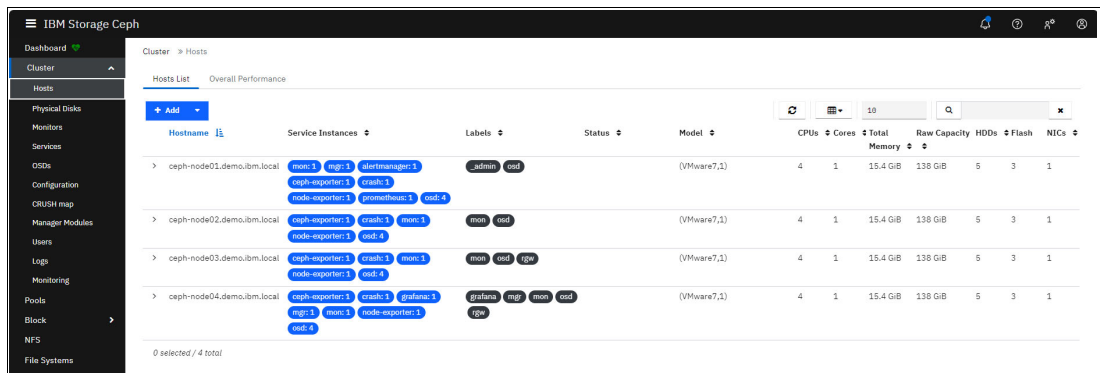


Figure 1-11 Checking hosts services

OSDs should show all the OSDs in and up, meaning they are available to be used. See Figure 1-12 on page 9.

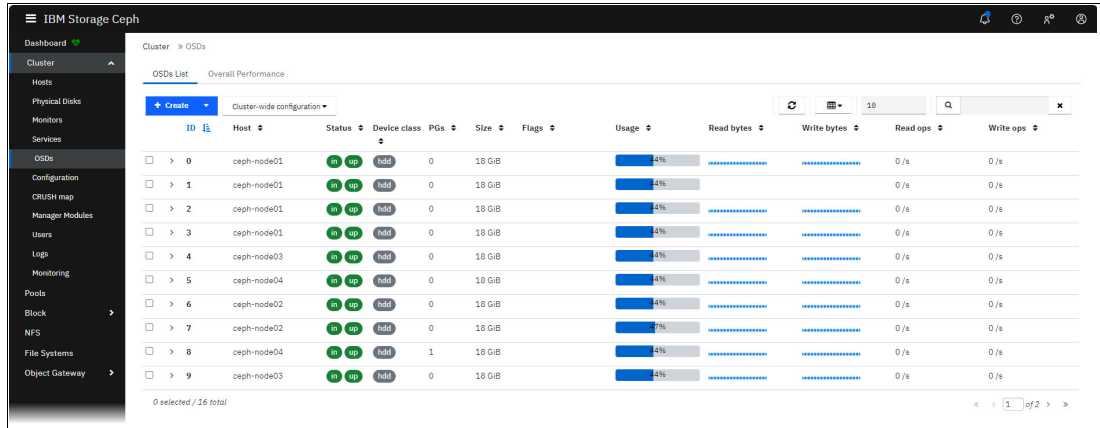


Figure 1-12 Checking the OSDs

## 1.4 Reduce the number of Monitors to 3

By default, a typical IBM Storage Ceph cluster has three or five monitor daemons deployed on different hosts. The usual recommendation 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, in our small cluster configuration of four nodes only, this results in the deployment of one monitor per node. That said, an odd number of monitors is usually recommended for quorum, so we are going to remove the monitor deployed on ceph-node04. This will remove some workload on ceph-node04 which is now the Grafana node.

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

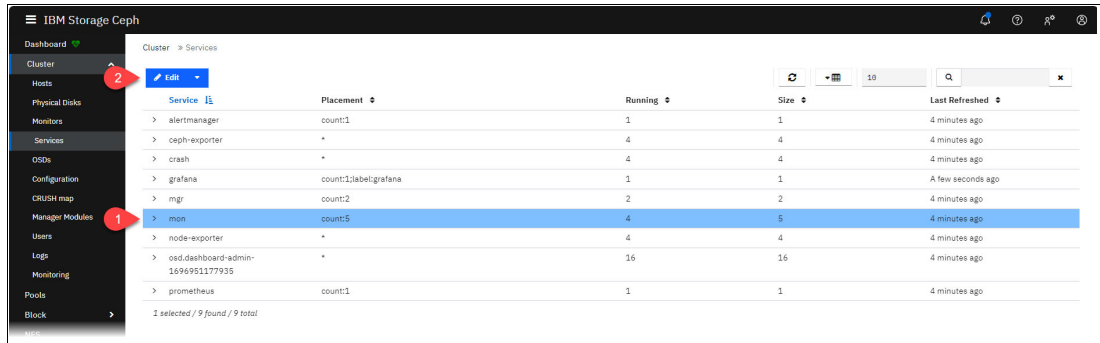


Figure 1-13 Edit mon service

2. We can edit the **Hosts list** to select first 3 hosts and reduce Count to 3. See Figure 1-14 on page 10.

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

- Type \***: A dropdown menu currently showing "mon".
- Id ?**: An empty text input field.
- Unmanaged**: A checkbox that is currently unchecked.
- Placement**: A dropdown menu currently showing "Hosts".
- Hosts**: A list of three hostnames: "ceph-node01.demo.ibm.local", "ceph-node02.demo.ibm.local", and "ceph-node03.demo.ibm.local". Each hostname has a small blue pencil icon to its left and a grey 'x' icon to its right.
- Count ?**: A text input field containing the number "3", with a green checkmark to its right.

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

Figure 1-14 Select mon hosts

3. Checking Services page will rapidly show the new placement of the mon service.

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

As we moved the Grafana service to hosts `ceph-node04.demo.ibm.local`, we need to accept its self-signed certificate to be able to display the content of the Grafana tabs.

In this example we are using Firefox, but the process should be quite similar on other browsers.

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



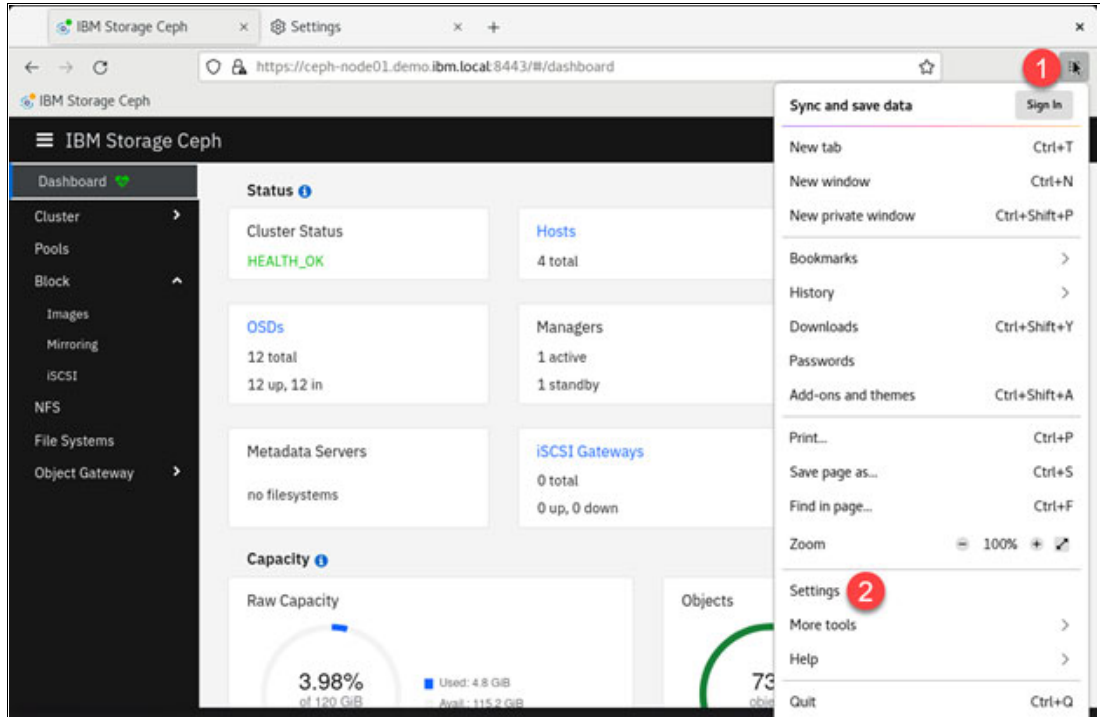


Figure 1-15 Browser certificates settings

2. Select **Privacy & Security** (1), go down on the right pane to find Certificates, then click **View Certificates...**(2). See Figure 1-16.

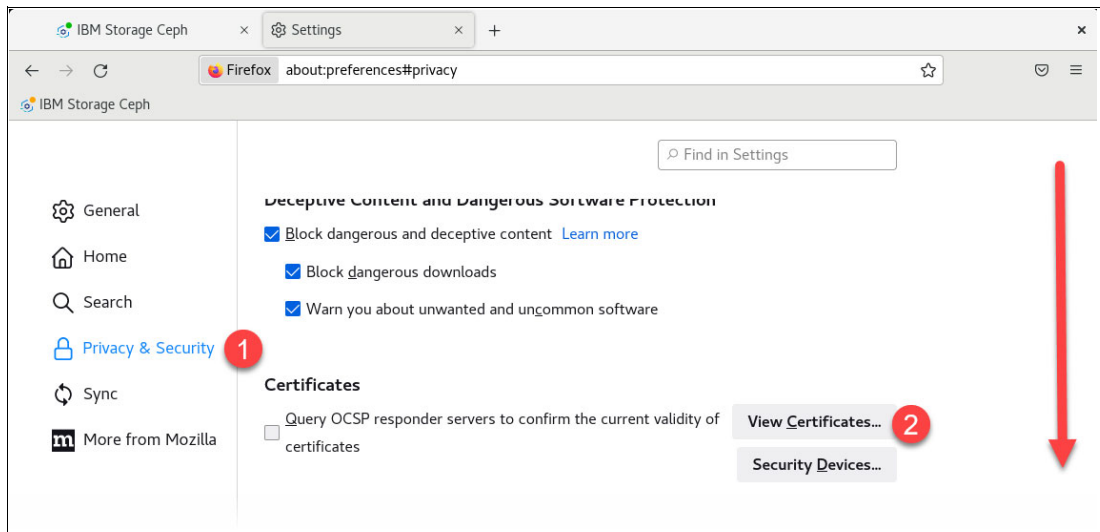


Figure 1-16 View certificates

3. On the Servers tab of Certificate Manager, click **Add Exception...** See Figure 1-17 on page 12.

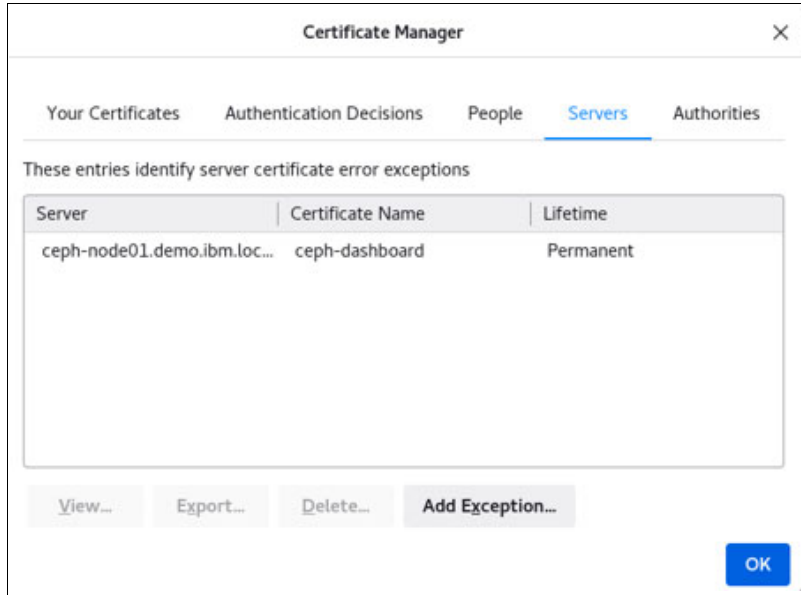


Figure 1-17 Certificates Servers tab

4. Add the ceph-node04 Grafana URL and click **Get Certificate**, then click **Confirm Security Exception**. See Figure 1-18.



Figure 1-18 Server certificate URL

5. We can click **OK** to validate the configuration and close the **Settings** tab.

## 1.5 Configure RADOS Gateway (RGW)

Ceph Object Storage uses the Ceph Object Gateway daemon (radosgw), an HTTP server 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 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, and the RADOS Gateway 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. In a production environment, it's recommended that you run more than one instance of the RGW masked by a Load Balancer.

### 1.5.1 Create rgw services and pools

Perform the following steps to create rgw services and pools.

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

We complete the Create Service dialog box as in the capture screen below and click **Create Service**. Remember we previously added the rgw label to ceph-node03 and ceph-node04. So we can now use this label to deploy the service on these hosts. See Figure 1-19 on page 14.

The 'Create Service' dialog box contains the following fields and options:

- Type \***: rgw (with a green checkmark)
- Id \***: default (with a green checkmark)
- Placement**: Label (with a green checkmark)
- Label**: rgw (with a green checkmark)
- Count ?**: (empty dropdown)
- Port**: (empty dropdown)
- Unmanaged
- SSL

Buttons: Cancel, Create Service

Figure 1-19 Create rgw service

2. We can verify the deployment of the service on the two selected hosts

Service	Count	Label	Status	Last Refreshed	CPU Usage	Memory Usage	Daemon Events
crash	4			8 minutes ago			
grafana	1	count1:labelgrafana		8 minutes ago			
mgr	2	count2		8 minutes ago			
mon	3	ceph-node01.demo.ibm.local;ceph-node02.demo.ibm.local;ceph-node03.demo.ibm.local;count3		8 minutes ago			
node-exporter	4	*		8 minutes ago			
osd.dashboard-admin-1696995177935	16	*		8 minutes ago			
prometheus	1	count1		4 minutes ago			
rgw.default	2	labelrgw		8 minutes ago			

Hostname	Daemon name	Version	Status	Last Refreshed	CPU Usage	Memory Usage	Daemon Events
ceph-node03.demo.ibm.local	rgw.default.ceph-node03.bsqzsg	17.2.6-100.el9cp	running	8 minutes ago	0%	72.1 MiB	29 minutes ago - Deployed rgw.default.ceph-node03.bsqzsg on host 'ceph-node03.demo.ibm.local'
ceph-node04.demo.ibm.local	rgw.default.ceph-node04.pjtqjd	17.2.6-100.el9cp	running	8 minutes ago	0%	72.3 MiB	29 minutes ago - Deployed rgw.default.ceph-node04.pjtqjd on host 'ceph-node04.demo.ibm.local'

Figure 1-20 RGW services details

3. The Pools tab (**Cluster** → **Pools**) will now show the creation of the 3 pools required by the rgw service to store all its storage needs.

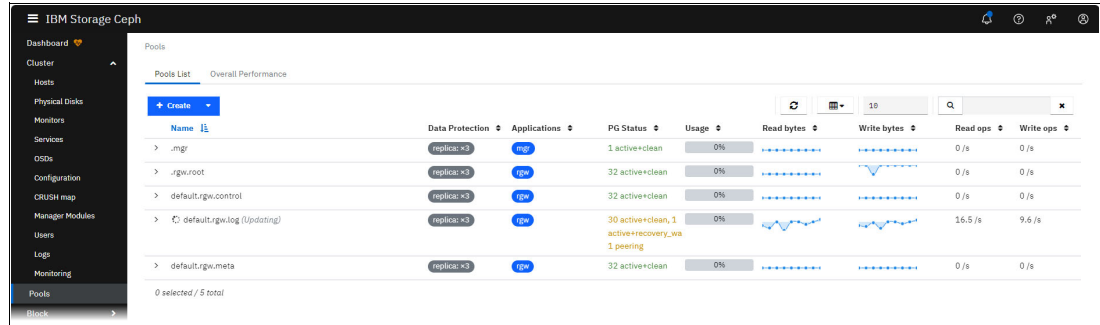


Figure 1-21 RGW service pools

## 1.5.2 Create ingress services

To avoid a single point of failure in the Ceph RGW deployment, we need a highly available (HA) and scalable S3/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 provided by cephadm, which provides an HA and load-balancing stack based on keepalived and haproxy.

1. Select **Cluster** → **Services** then **Create** and complete the Create Service dialog box as shown in Figure 1-22 on page 16.

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:

Figure 1-22 Ingress service create

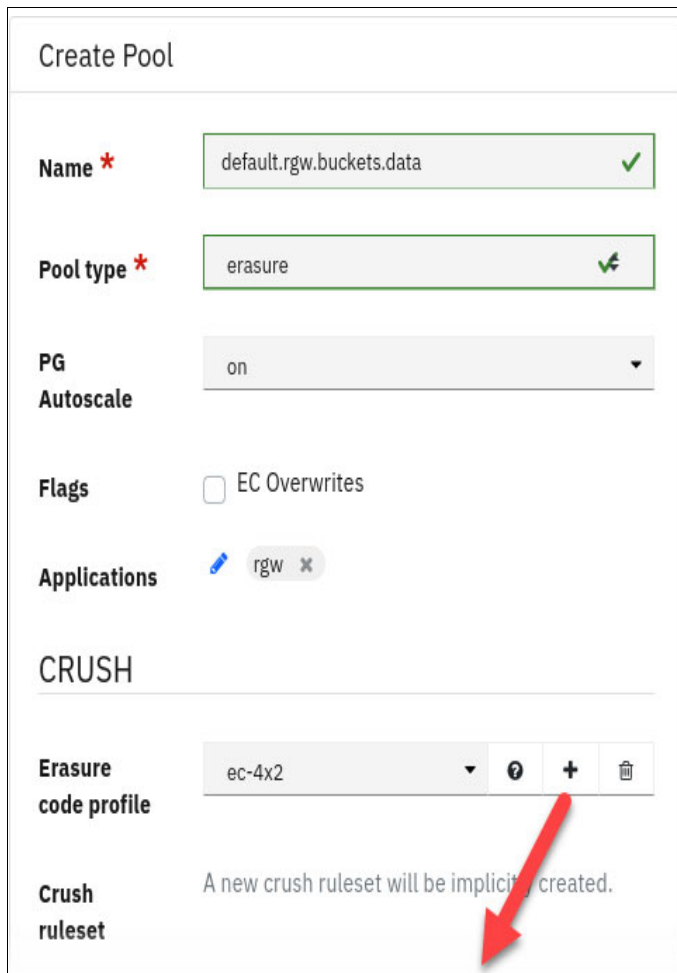
The type is of course ingress, it will use the rgw default backend service created, be deployed on the hosts with the rgw label, with a virtual IP address a frontend port and a monitor port.

### 1.5.3 Create the data bucket pool using Erasure Coding

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

1. From the left navigation panel, select **Pools** → **Create** then **Create**.

2. We 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 on the + to be able to create our own Erasure Code profile. See Figure 1-23.



The screenshot shows the 'Create Pool' dialog box. It has 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 icon)
- Erasure code profile**: `ec-4x2` (dropdown menu with a plus icon and a trash icon)
- Crush ruleset**: `A new crush ruleset will be implicitly created.`

A red arrow points to the plus icon next to the Erasure code profile field.

Figure 1-23 Create rgw data bucket

3. Let us create a  $k=4, m=2$  profile, that will allow the loss of 2 OSDs ( $m=2$ ) by distributing an object on 6 ( $k+m=6$ ) OSDs, to do so complete the dialog box as shown in Figure 1-24 on page 18. Then click **Create EC Profile**, then **Create Pool**.

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

- Name: ec-4x2
- Plugin: jerasure
- Data chunks (k): 4
- Coding chunks (m): 2
- Crush failure domain: osd ( 16 )
- Technique: reed\_sol\_van
- Packetsize: 2048
- Crush root: default
- Crush device class: hdd (Available OSDs: 16)
- Directory: /usr/lib64/ceph/erasure-code

Figure 1-24 Create EC profile

**Note:** These settings are for demonstration purpose only. For a supported production environment, 7 (4+2+1) Ceph nodes would be required for such Erasure Coding settings.

### 1.5.4 Create the user and its key

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

1. Select **Object Gateway** → **Users** then **Create**. See Figure 1-25 on page 19.



The screenshot shows a 'Create User' form 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 right, there are two buttons: 'Cancel' and 'Create User'.

Figure 1-25 Create RGW Use

Here we only need to specify the user ID and the Full name. Let us use demouser in both cases. The user s3 key will be created automatically.

We have created a user, so we can now create a bucket and assign it to them.

2. Select **Object Gateway** → **Buckets** then Create and complete the dialog box as follow. Placement target corresponds to the EC pool we created previously, meaning the bucket will be protected using erasure coding. See Figure 1-26 on page 20.

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

**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 Put data in your bucket using AWS CLI

As an example we are using AWS CLI which we previously installed on a RHEL workstation. This command line tool allows to manipulate S3 buckets. We need to configure it with the demouser's key to use it. These keys can be retrieved from the Ceph Dashboard.

1. In order to get the demouser access key and secret key select **Object Gateway** → **Users** → **demouser** → **keys** → **show**.

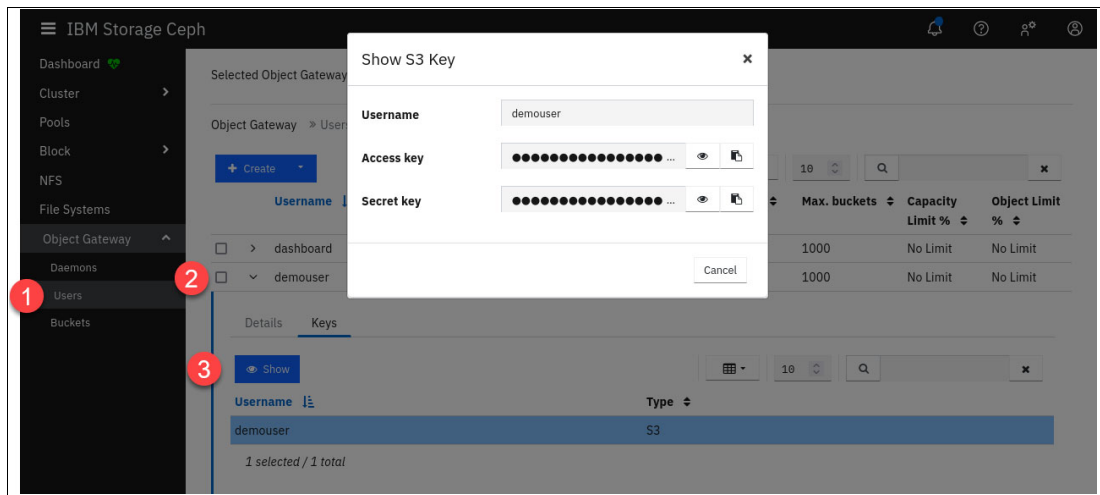


Figure 1-27 Locate user's S3 keys

2. Using the RHEL workstation, open a terminal and enter `aws configure`, then copy and paste the keys from the Dashboard. See Figure 1-28 on page 21.

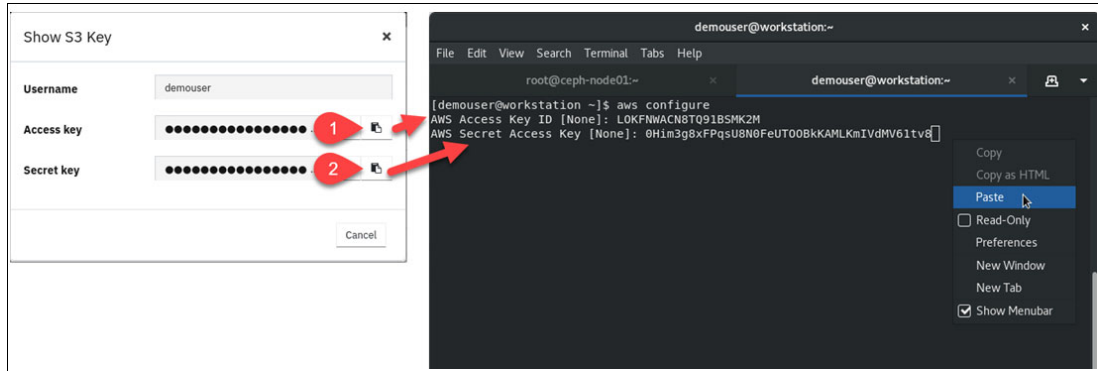


Figure 1-28 AWS CLI configuration

- Then, type Enter twice to keep Default region name and Default output format to None. AWS CLI is now configured to use your bucket. Let us 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 pool status

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

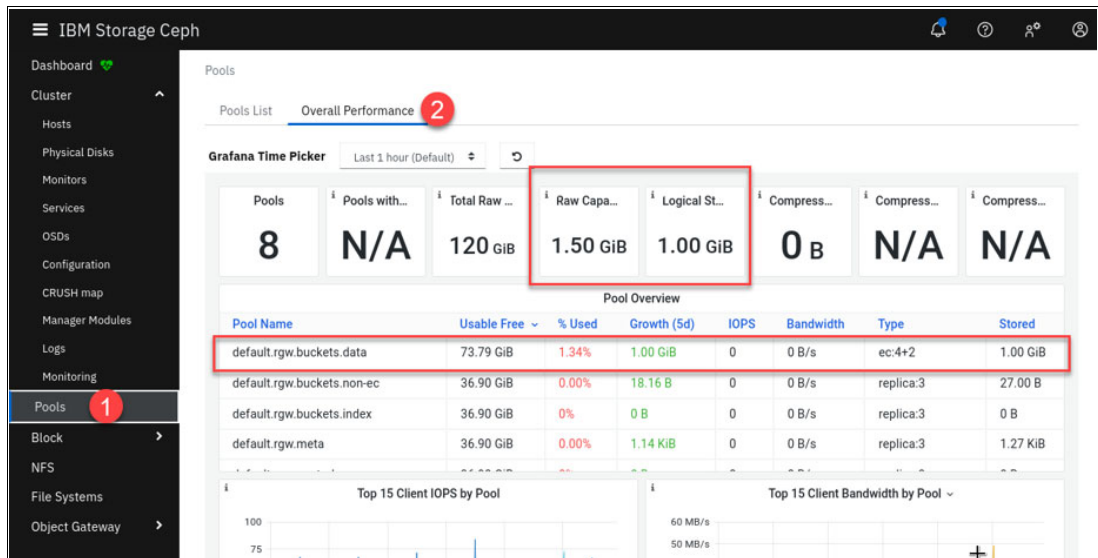


Figure 1-29 Visualizing Grafana data

This will show the raw and logical space used on OSDs and also some statistics on 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 Create RADOS Block Disk (RBD)

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, Kubernetes) or through the librbd API (OpenStack, Proxmox). In the Kubernetes world, RBDs are designed to address the need for RWO PVCs.

### 1.6.1 Create a RBD pool

To create block disk we of course need to create a pool to allow some storage.

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

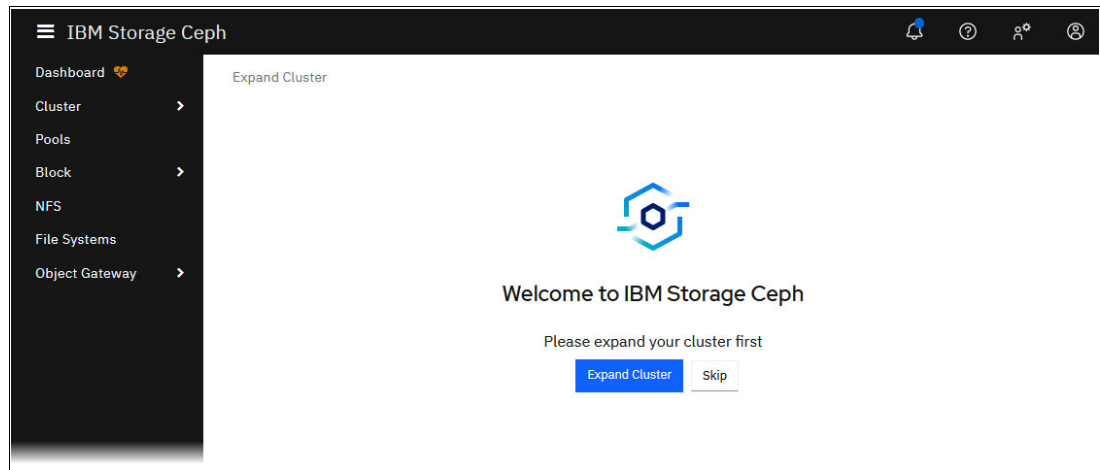


Figure 1-30 Block Image

2. We give a name to the pool (rbd\_pool1) and we associate Applications with the rbd label. See Figure 1-31 on page 23.

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

- Name \***: rdp\_pool ✓
- Pool type \***: replicated ✓
- PG Autoscale**: on
- Replicated size \***: 3
- Applications**: rdp x
- CRUSH**: replicated\_rule
- Compression**: Mode: none
- Quotas**: Max bytes: e.g., 10GiB; Max objects: 0
- RBD Configuration**: Quality of Service: [empty]

Buttons at the bottom: Cancel, Create Pool

Figure 1-31 Create rbd pool

## 1.7 Create an image

In this step we create the disk that will be made accessible. We select **Block** → **Images** then select **Create**. Complete the dialog box to set the name (disk1) the pool to use (rdp\_pool) and the size of the disk. See Figure 1-32 on page 24.

Figure 1-32 Create disk

### 1.7.1 Map the created disk

The image is now available on the Ceph cluster, but it still must be mounted on the client filesystem. Since the `ceph-common` package has been installed and configured on the workstation, we can use the RBD Kernel module to mount this device. See 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 filesystem with 524288 4k blocks and 131072 inodes
Filesystem 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 journal (16384 blocks): done
Writing superblocks and filesystem 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/
```

---

We can now use the `rbd_mnt` folder like any other folder in the filesystem, such as by moving a file into the new mounted folder. We will use it to create snapshots. See 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
```

## 1.7.2 Using snapshot

Going back to the Images tab of the Dashboard. We can see the space used on the disk. We are now going to create a snapshot.

1. First let use check the disk usage. See Figure 1-33.

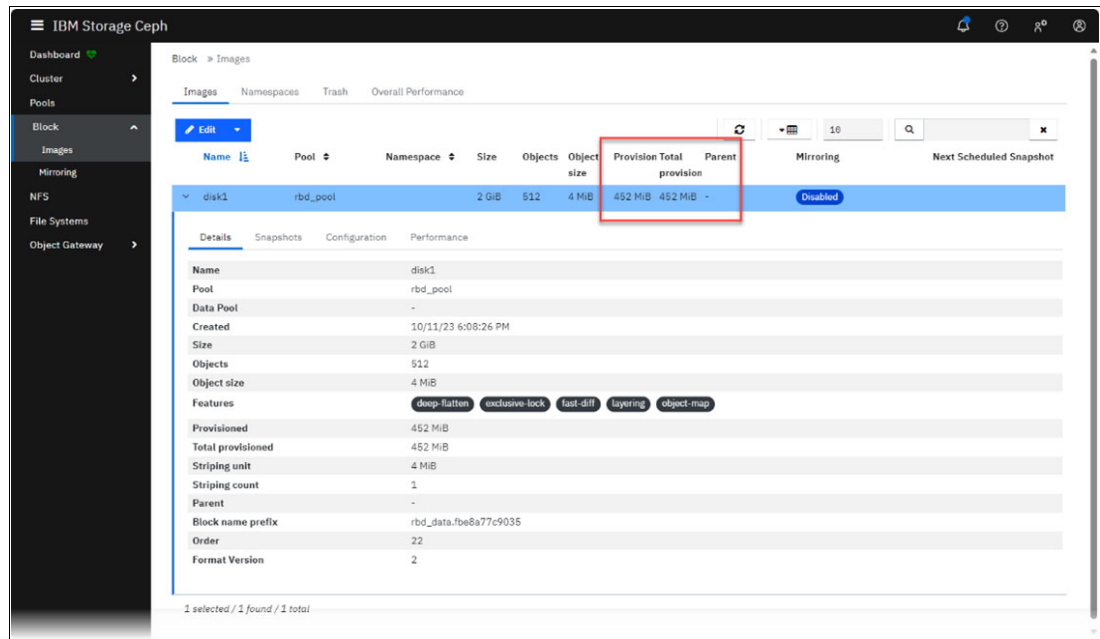


Figure 1-33 Disk Usage

2. Create a snapshot. Select the **Snapshots** tab, then **+ Create**. A name is automatically assigned. See Figure 1-34.

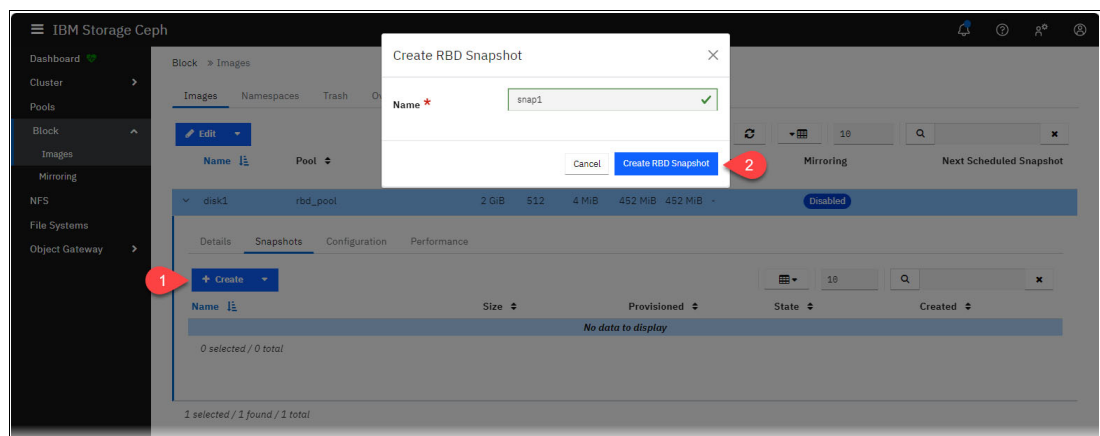


Figure 1-34 Snapshot creation

3. Going back to the command console, we can now delete the file. See Example 1-3 on page 26.

Example 1-3 Delete the time

```
[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, as it can be time consuming it is recommended to use snapshot clone instead. To do so we must first change the snapshot from unprotected to protected using the dropdown menu. See Figure 1-35.

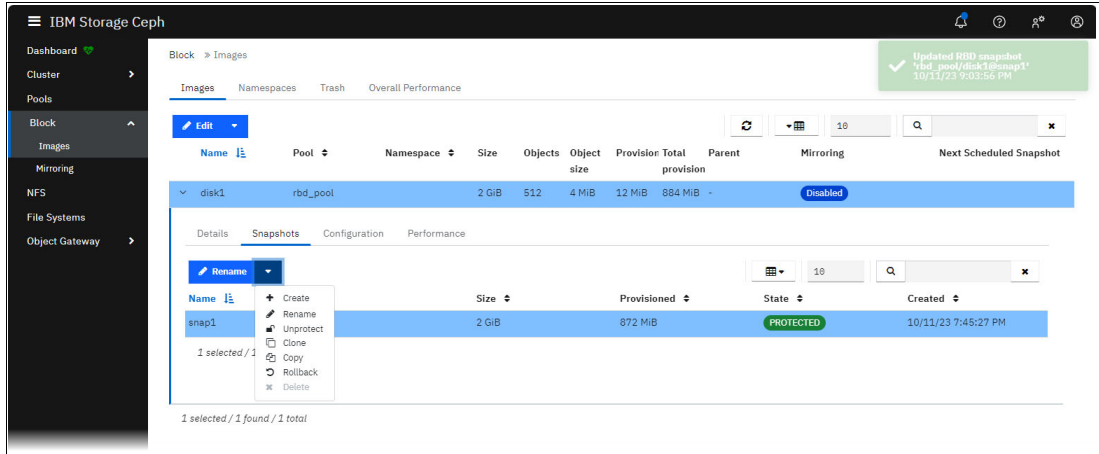


Figure 1-35 Clone snapshot

- 5. Next we will create a clone of the snapshot. In the dropdown menu, we select **Clone** and enter the name of the clone. See Figure 1-36 on page 27.



### Clone RBD

**Clone from**

---

**Name \***

**Pool \***  ▼

Use a dedicated data pool

**Size \***

**Features**

- Deep flatten
- Layering
- Exclusive lock
- Object map (requires exclusive-lock)
- Fast diff (interlocked with object-map)
- Mirroring

[Advanced...](#)

Figure 1-36 Clone disk1 snapshot

Figure 1-37 shows the Snapshot list.

Block > Images

Images   Namespaces   Trash   Overall Performance

Edit
↻
⌵
10
Q
✕

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

1 selected / 2 found / 2 total

Figure 1-37 Snapshot list

6. In order to restore the deleted file, we create a new directory to mount the clone, then we map the cloned disk before mounting it in the new directory. We can now copy or move the deleted file in the `rnd_mnt` directory. See Example 1-4 on page 28.

*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
```

---

## 1.8 Create CephFS

One of the features available in a Ceph cluster is the Ceph File System, also known as CephFS. It is a POSIX compliant file system that stores its data and metadata using Ceph pools.

As POSIX uses inodes to uniquely reference its content, CephFS offers a dedicated component that allows CephFS client machines to locate the actual RADOS objects for a given inode. This component is known as the *MetaData Server* or *MDS*.

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

The MDS will also keep track of the metadata for the directory or the file such as ACLs.

To configure a Ceph File System you must deploy one or more MDSs (at least 2 to provide a highly available architecture). Once the MDS components are up and running, you can then create your shared file system.

This chapter describes how to configure a MetaData Server and create a file system through the Ceph dashboard. The configuration will create two Ceph pools:

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

Perform the following steps:

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

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

- Type \***: mds
- Id \***: cephfs
- Unmanaged
- Placement**: Hosts
- Hosts**: ceph-node03.demo.ibm.local, ceph-node04.demo.ibm.local
- Count**: (empty)

Buttons: Cancel, Create Service

Figure 1-38 Create mds service

2. Create the cephFS pools: Select **Pools** and click **+ Create** and complete the dialog box as shown in Figure 1-39.

**Create Pool**

**Name \*** cephfs\_data ✓

**Pool type \*** replicated ✓

**PG Autoscale** on

**Replicated size \*** 3

**Applications** cephfs ✕

**CRUSH**

**Crush ruleset** replicated\_rule ⓘ + 🗑️

**Compression**

**Mode** none

**Quotas**

**Max bytes** ⓘ e.g., 10GiB

**Max objects** ⓘ 0

Cancel Create Pool

Figure 1-39 Cephfs pools

3. Repeat the same to create the cephfs\_metadata pool with the following values:

- Name: cephfs\_metadata
- Pool type: replicated
- Applications: cephfs

Once completed, you can verify your configuration. See Figure 1-40.

Name	Data Protection	Applications	PG Status	Usage	Read bytes	Write bytes	Read ops	Write ops
> .mgr	replicas: x3	mgr	1 active+clean	0%	0 / s	0 / s	0 / s	0 / s
> .rgw.root	replicas: x3	rgw	32 active+clean	0%	0 / s	0 / s	0 / s	0 / s
> cephfs_data	replicas: x3	cephfs	1 active+clean	0%	0 / s	0 / s	0 / s	0 / s
> cephfs_metadata	replicas: x3	cephfs	1 active+clean	0%	0 / s	0 / s	0 / s	0 / s
> default:rgw.buckets.data	EC: 4+2	rgw	1 active+clean	0%	0 / s	0 / s	0 / s	0 / s
> default:rgw.control	replicas: x3	rgw	32 active+clean	0%	0 / s	0 / s	0 / s	0 / s
> default:rgw.log	replicas: x3	rgw	32 active+clean	0%	0 / s	0 / s	0 / s	0 / s
> default:rgw.meta	replicas: x3	rgw	32 active+clean	0%	0 / s	0 / s	0 / s	0 / s
> rbd_pool	replicas: x3	rbd	1 active+clean	2.89%	0 / s	0 / s	0 / s	0 / s

Figure 1-40 Cephfs pools

- Next we need to create a filesystem. To do so we open a terminal session using `root@ceph-node01` and launch the following commands. See Example 1-5.

*Example 1-5 Create a new filesystem*

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

- Next, we create a new user and allow him to access read/write the filesystem. See Example 1-6.

*Example 1-6*

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

- Once the key is generated, it can be used to mount the filesystem on the client using `fstab`, for example. For this demonstration, we will use `ceph-fuse`, a userspace client for CephFS filesystems, because it is easier to use.

To use `ceph-fuse` on the workstation we need to export the key to a keyring file and copy it to the workstation. See Example 1-7.

*Example 1-7 Export the key to a keyring file and copy it to the workstation*

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

- We will mount the filesystem on the workstation. In a terminal console, opened as `demouser@workstation`, we mount the filesystem. See Example 1-8 on page 32.

*Example 1-8 Mount the filesystem 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. The filesystem can now be used. From the Dashboard we can see the File Systems properties. See Figure 1-41.

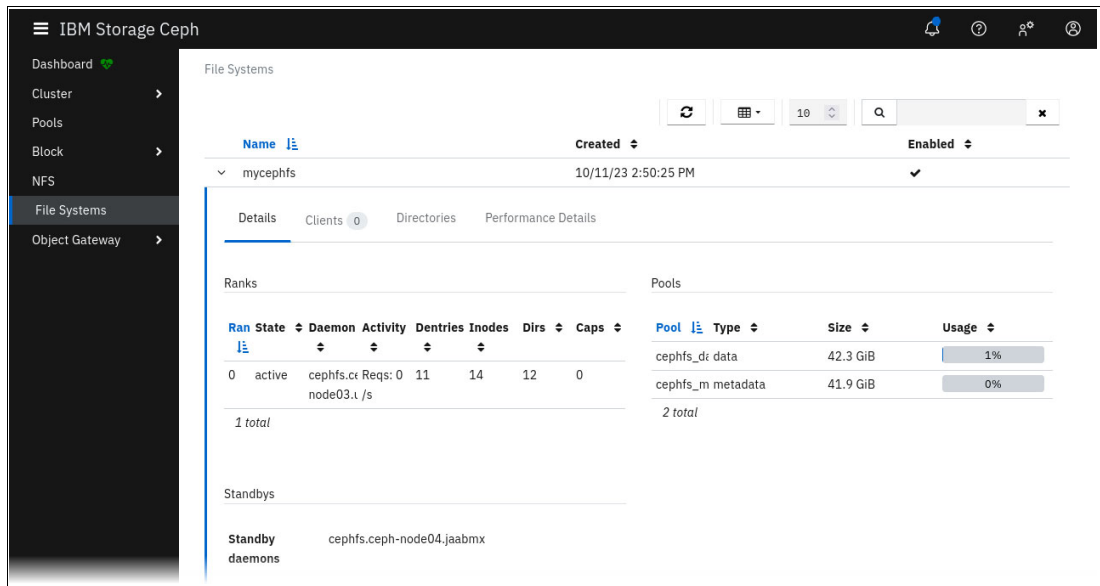


Figure 1-41 Filesystem properties

## 1.9 Monitoring

Ceph Dashboard also provides the storage administrator with monitoring information.

### 1.9.1 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). See Figure 1-42 on page 33.

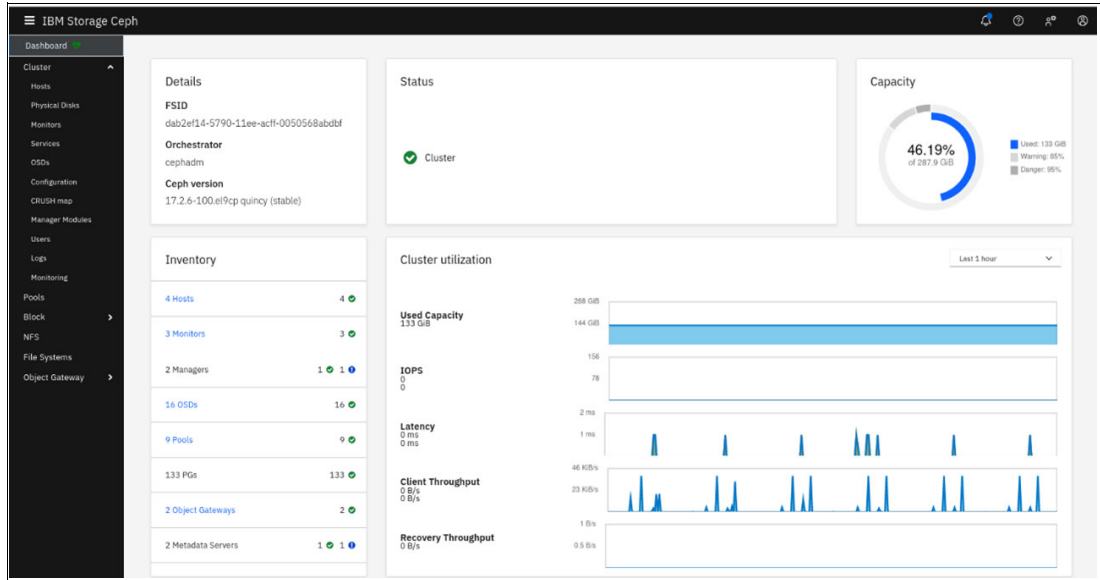


Figure 1-42 Dashboard monitoring

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

### 1.9.2 Cluster menu

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

Figure 1-43 shows the Hosts performance data.

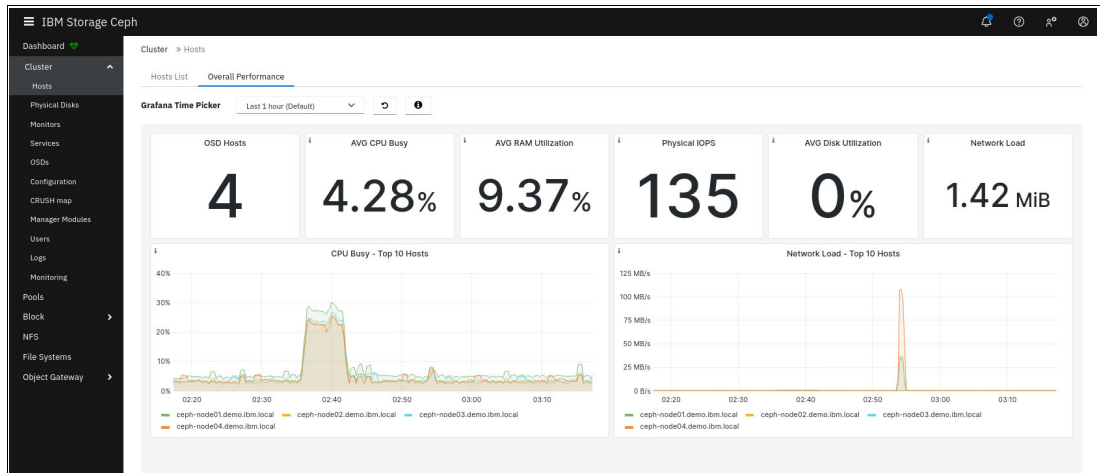


Figure 1-43 Hosts overall monitoring

Figure 1-44 on page 34 shows the OSDs performance data.

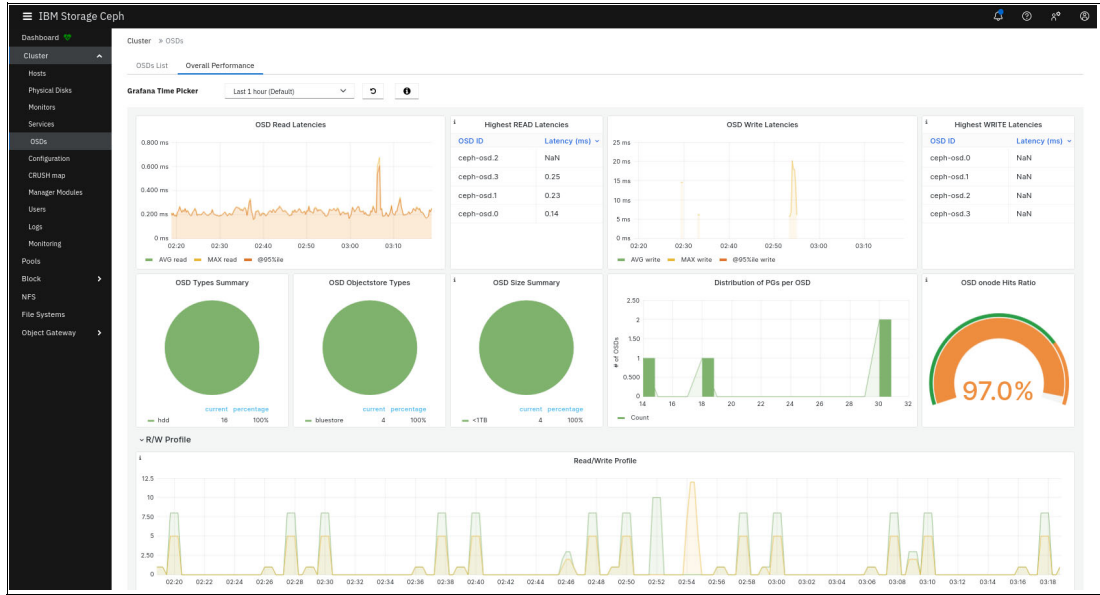


Figure 1-44 OSDs overall monitoring

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

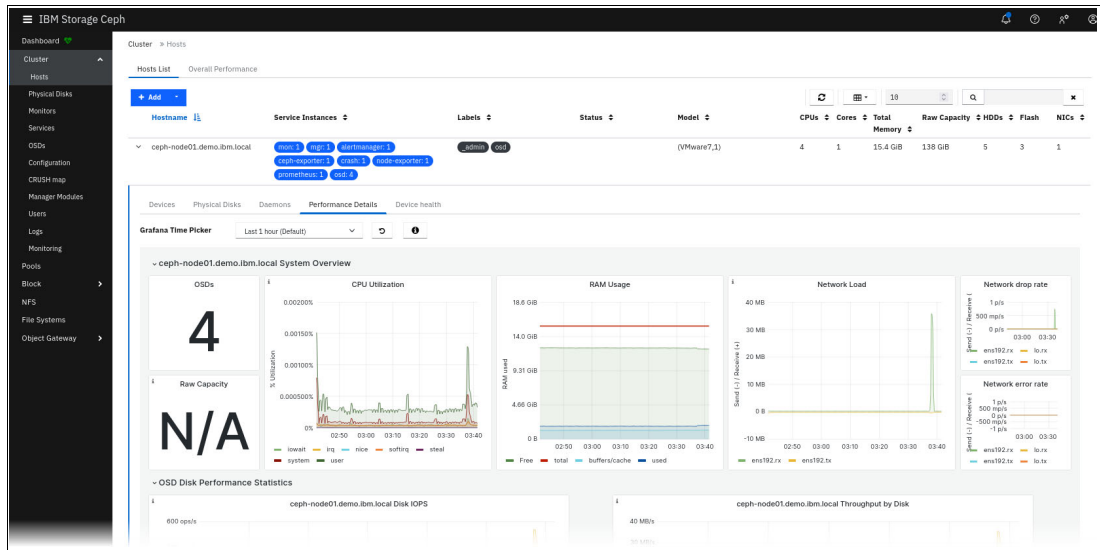


Figure 1-45 Host detail monitoring

The Cluster menu also provides an easy access to the logs which is a real help to sort events when troubleshooting a problem. See Figure 1-46 on page 35.



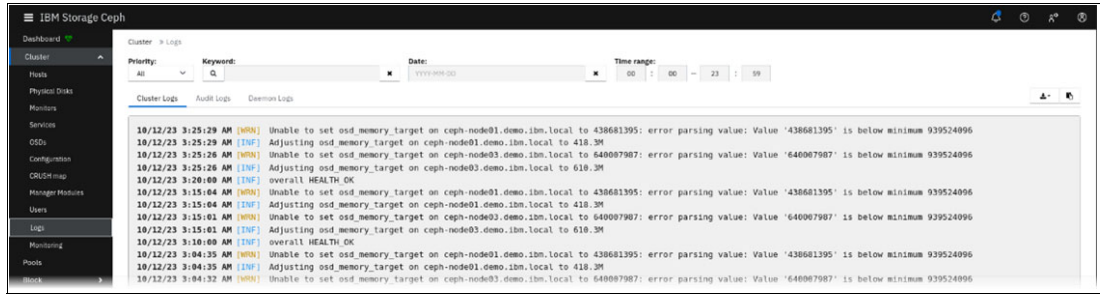


Figure 1-46 Logs monitoring

The Monitoring menu is also convenient to list active and previous alerts with detailed information about it, but the most useful tab may be the Silences tab, which allows to easily create silences on specific alerts when doing maintenance on the cluster to avoid flooding of the logs. See Figure 1-47.

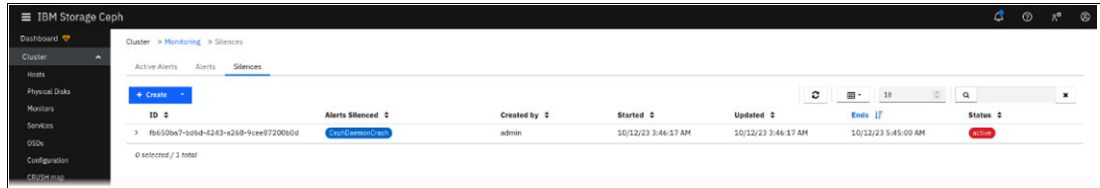


Figure 1-47 Monitoring silences

### 1.9.3 Other Grafana performance data

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

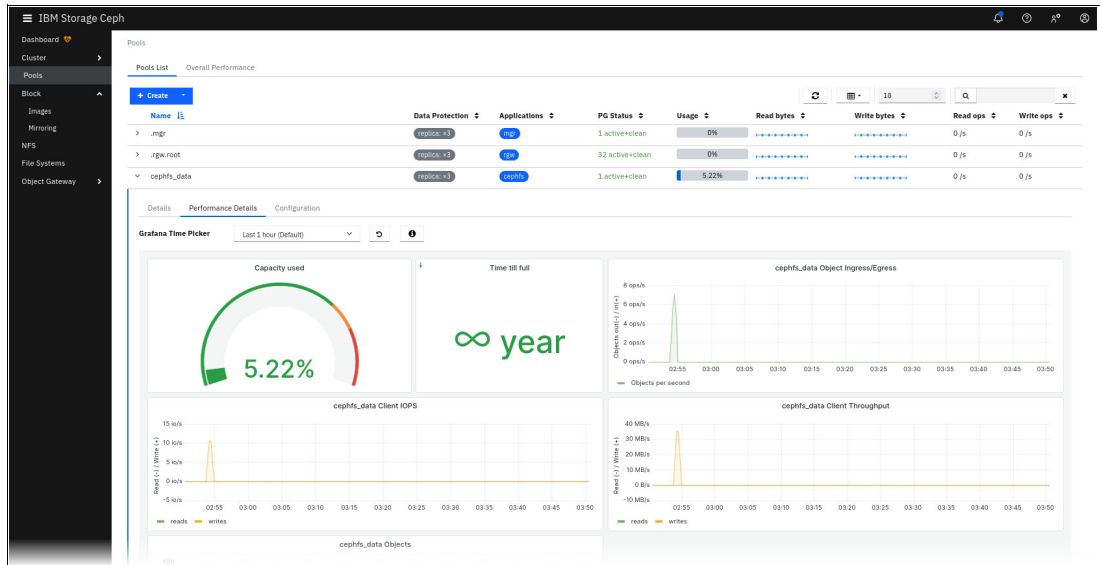


Figure 1-48 Pool monitoring

Overall, we saw that the Ceph Dashboard is a very powerful tools that can greatly ease the day to day work of a storage administrator.





# 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, leveraging 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 easy and efficient way to build a data lakehouse for IBM watsonx.data and for next-generation AI workloads. This chapter covers the configuration steps for integrating IBM Storage Ceph and IBM watsonx.data and has the following sections:

- ▶ “Using IBM Storage Ceph with IBM watsonx.data” on page 38
- ▶ “Creating users” on page 38
- ▶ “Creating buckets” on page 42
- ▶ “Performance optimizations” on page 54
- ▶ “Querying less structured data (S3-Select)” on page 55

**Note:** Refer to [Accelerating IBM watsonx.data with IBM Fusion HCI](#) IBM Redpaper for more information on watsonx.data with IBM Fusion HCI.

## 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 and data lakes, allowing you to optimize each configuration within a single Storage Ceph cluster. You can share large volumes of data through open table formats like Apache Iceberg, 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 the same data, utilizing tools like Parquet, Avro, Apache Orc, and more. In this chapter we will cover the configuration steps for integrating IBM Storage Ceph and IBM watsonx.data.

## 2.2 Creating users

First step is creating a watsonx user. Users can be created through the Ceph Dashboard or the command line interface.

### 2.2.1 Creating a user with the Ceph Dashboard

Perform the following steps to create a user with the Ceph Dashboard:

1. Using the menu on the left side navigate to the Object Gateway section and click the **Users** tab to open the pane for user management. Click the blue **Create** button to open the user creation page. See Figure 2-1.

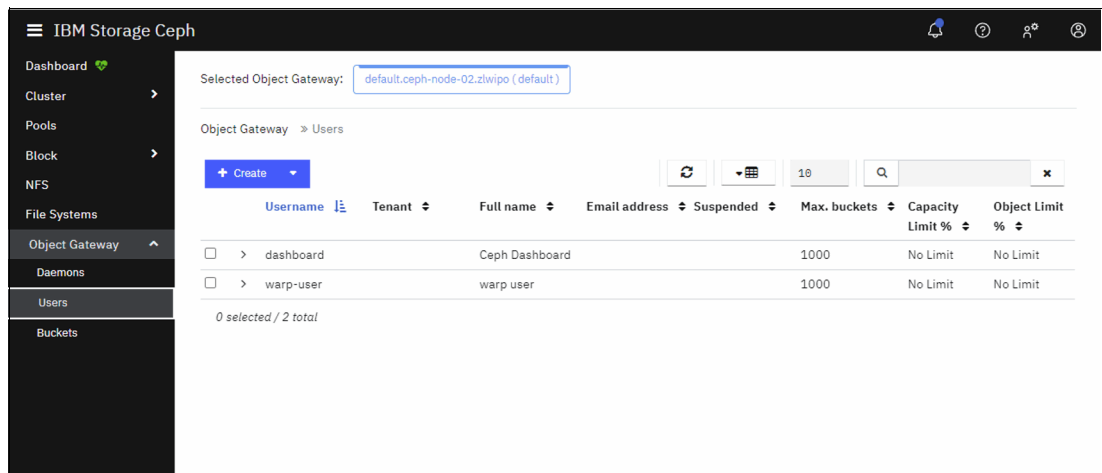


Figure 2-1 User creation page

2. The user creation page will prompt for some information about the user. Fill out all mandatory fields as indicated by the red asterisks. When finished, click the blue **Create User** button. See Figure 2-2 on page 39.

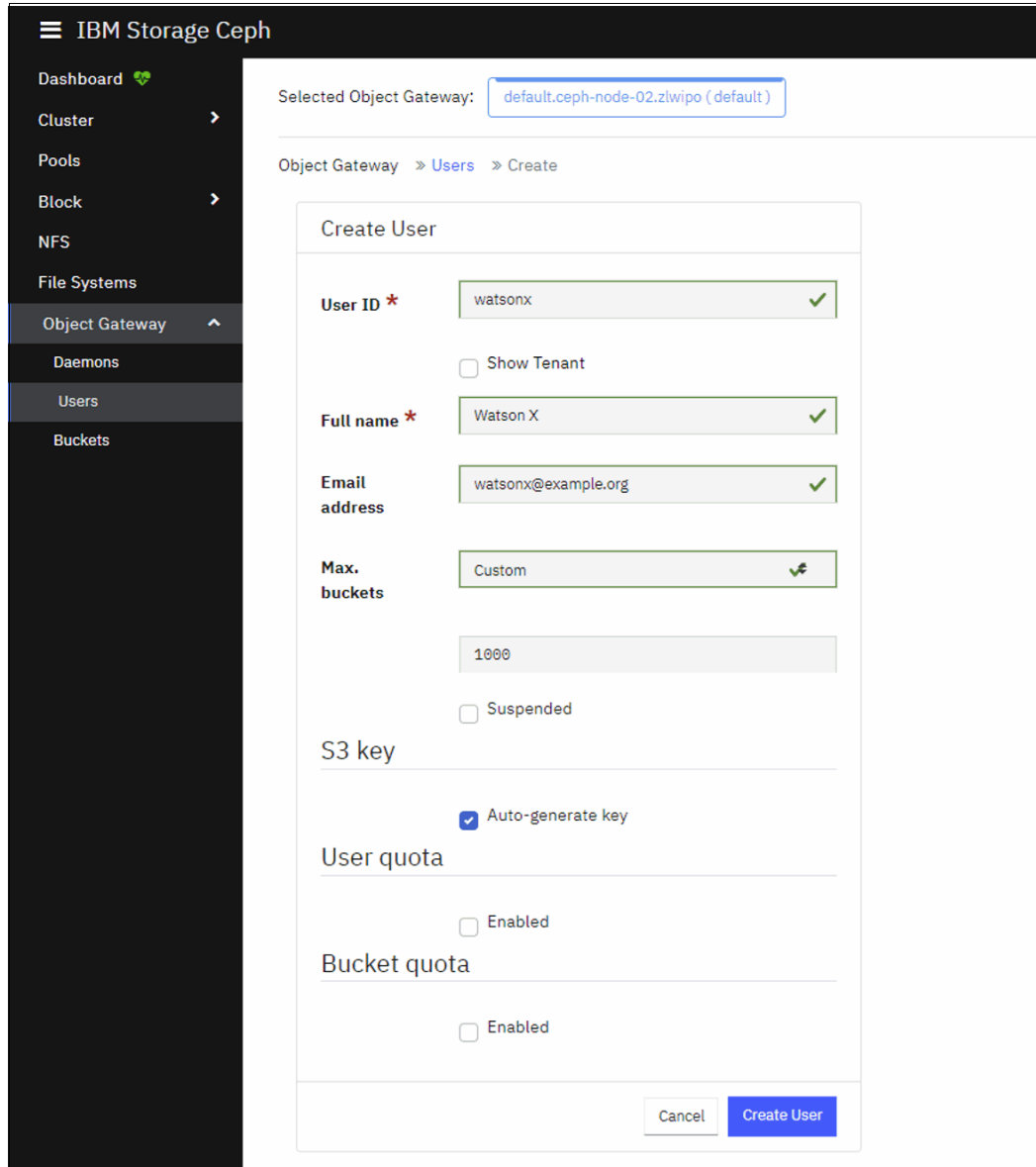


Figure 2-2 User creation

3. Upon creating 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. See Figure 2-3 on page 40.

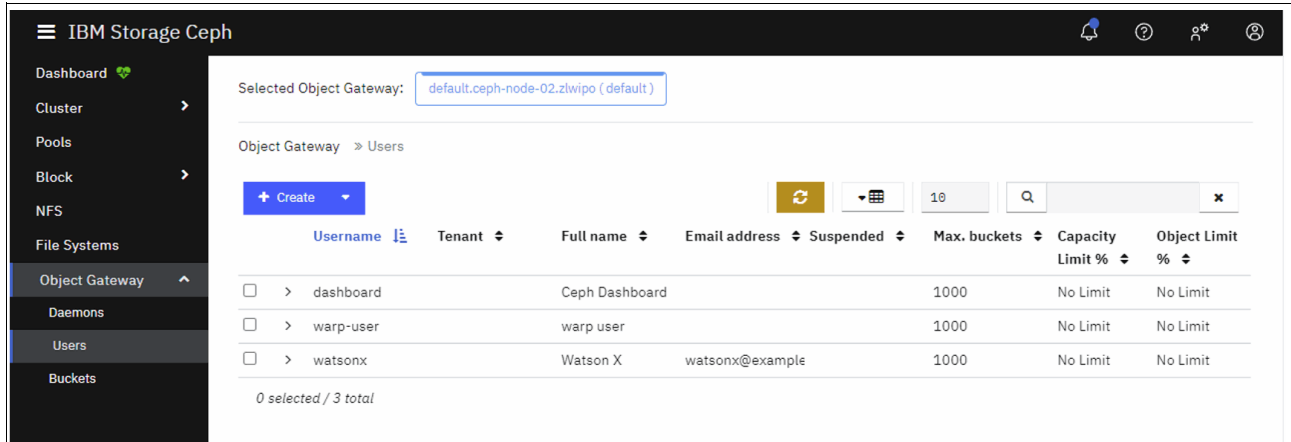


Figure 2-3 Newly created user

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

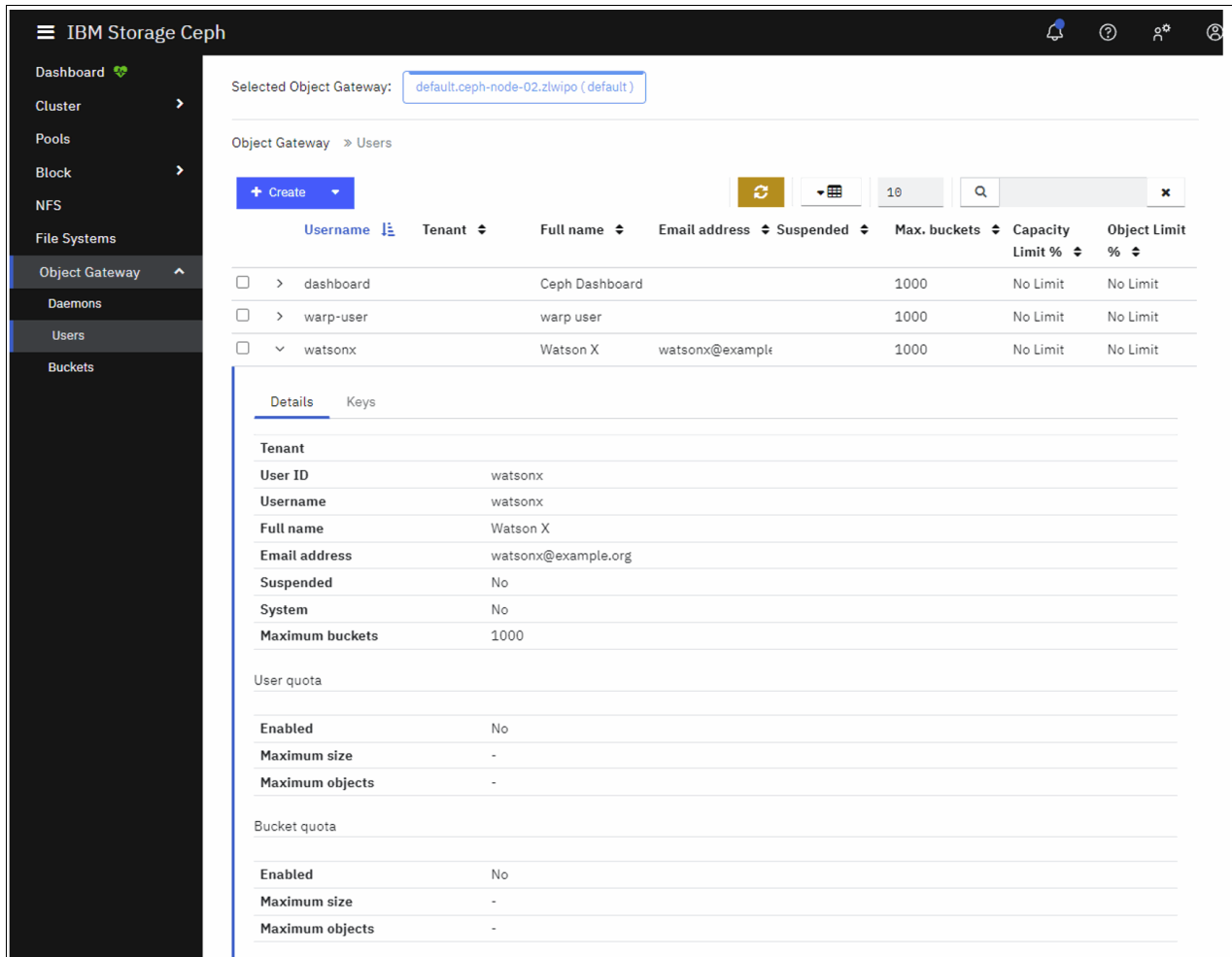


Figure 2-4 User details

- After selecting the Keys tab, click the blue **Show** button to view or copy the user's credentials. See Figure 2-5 on page 41.

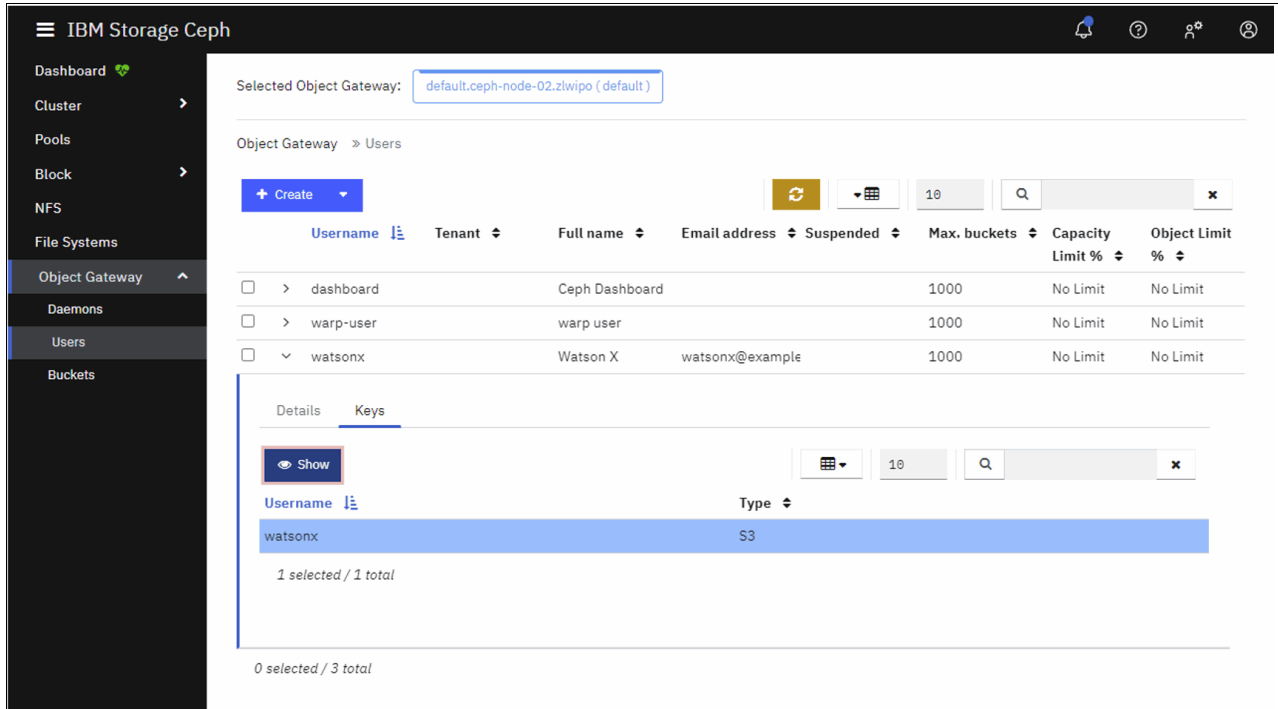


Figure 2-5 View or copy the user's credentials

- The access and secret key will be needed to add a bucket in the watsonx.data Infrastructure Manager. See Figure 2-6.

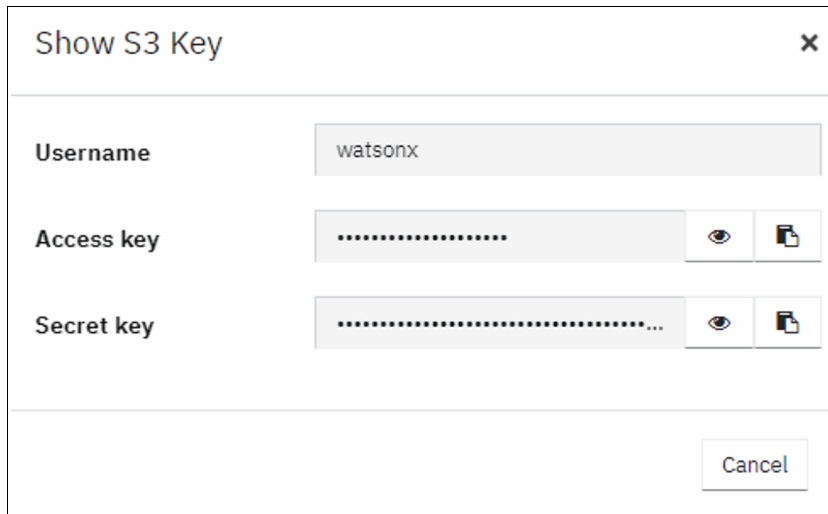


Figure 2-6 Show S3 Key

### 2.2.2 Creating user via the command line

Alternatively, users can be created using the command line method described in Example 2-1 on page 41.

*Example 2-1 Creating user via the command line*

```
radosgw-admin user create \
  --uid watsonx \
```

```
--display-name "Watson X"
```

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

## 2.3 Creating buckets

Next step is to create buckets. This can be done through the Ceph Dashboard or the command line interface.

### 2.3.1 Creating bucket with the Ceph Dashboard

Perform the following steps to create a bucket with the Ceph Dashboard:

1. Using the menu on the left side navigate to the Object Gateway section and click the **Buckets** tab to open the pane for Bucket management. Click the blue **Create** button to open the bucket creation page. See Figure 2-7.

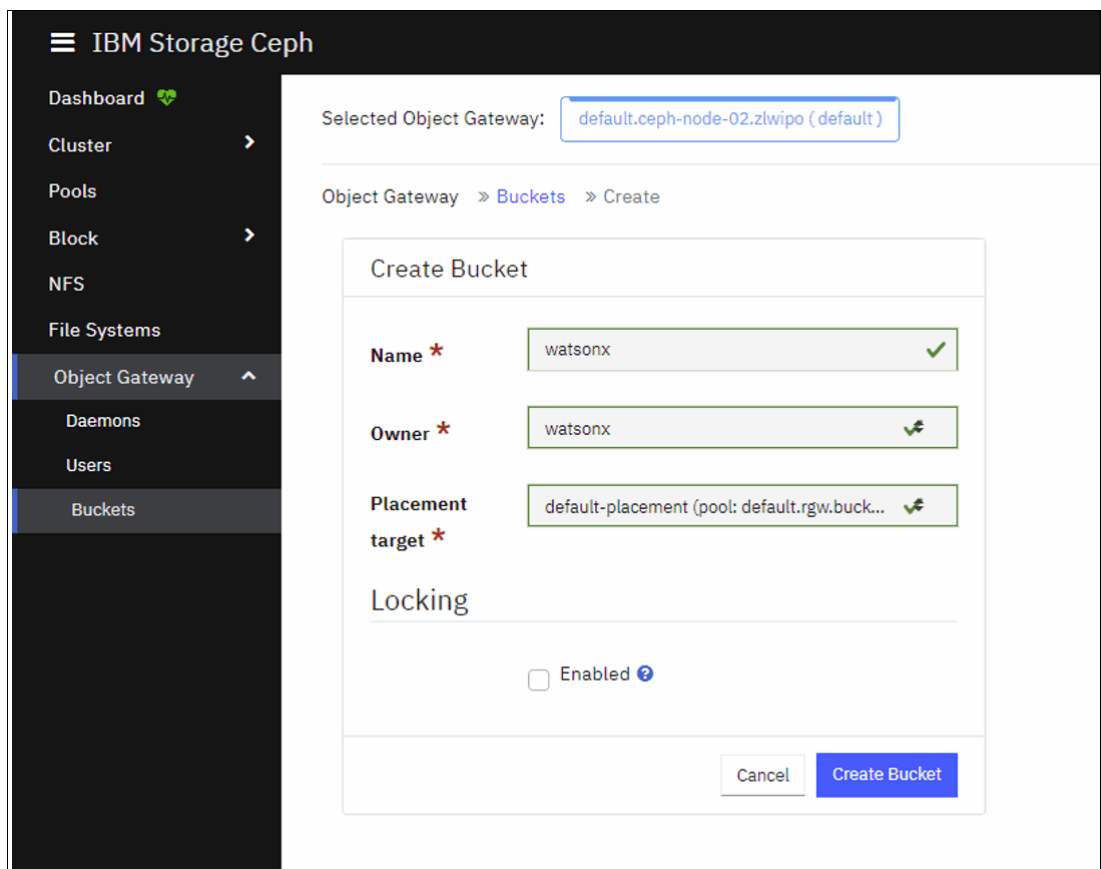


Figure 2-7 Bucket creation page

2. The bucket creation page will prompt for some information about the bucket you wish to create, fill out all mandatory fields as indicated by the red asterisks. Ensure the user created in 2.2, "Creating users" on page 38 is set as the bucket owner.
3. When finished, click the blue **Create Bucket** button.



## 2.3.2 Creating bucket at the command line

We recommend the use of `s5cmd` for command line interaction with the Ceph Object Gateway. It is both extremely fast and highly portable, making it a great choice regardless of OS.

Employing a credentials file eliminates the need to source environmental variables every time you start a new shell session. The commands shown in Example 2-2 will create a credentials file in the location expected by `s5cmd` and many other s3 utilities that utilize the S3 SDKs.

*Example 2-2 Create 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 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 a bash aliases follows:

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

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

## 2.4 Adding a bucket in watsonx.data

Perform the following steps to add a bucket in watsonx.data:

1. From the Lakehouse user interface browse to the Infrastructure Manager section. From the Infrastructure Manager click the **Add Component** dropdown on the left side of the screen. See Figure 2-8 on page 44.

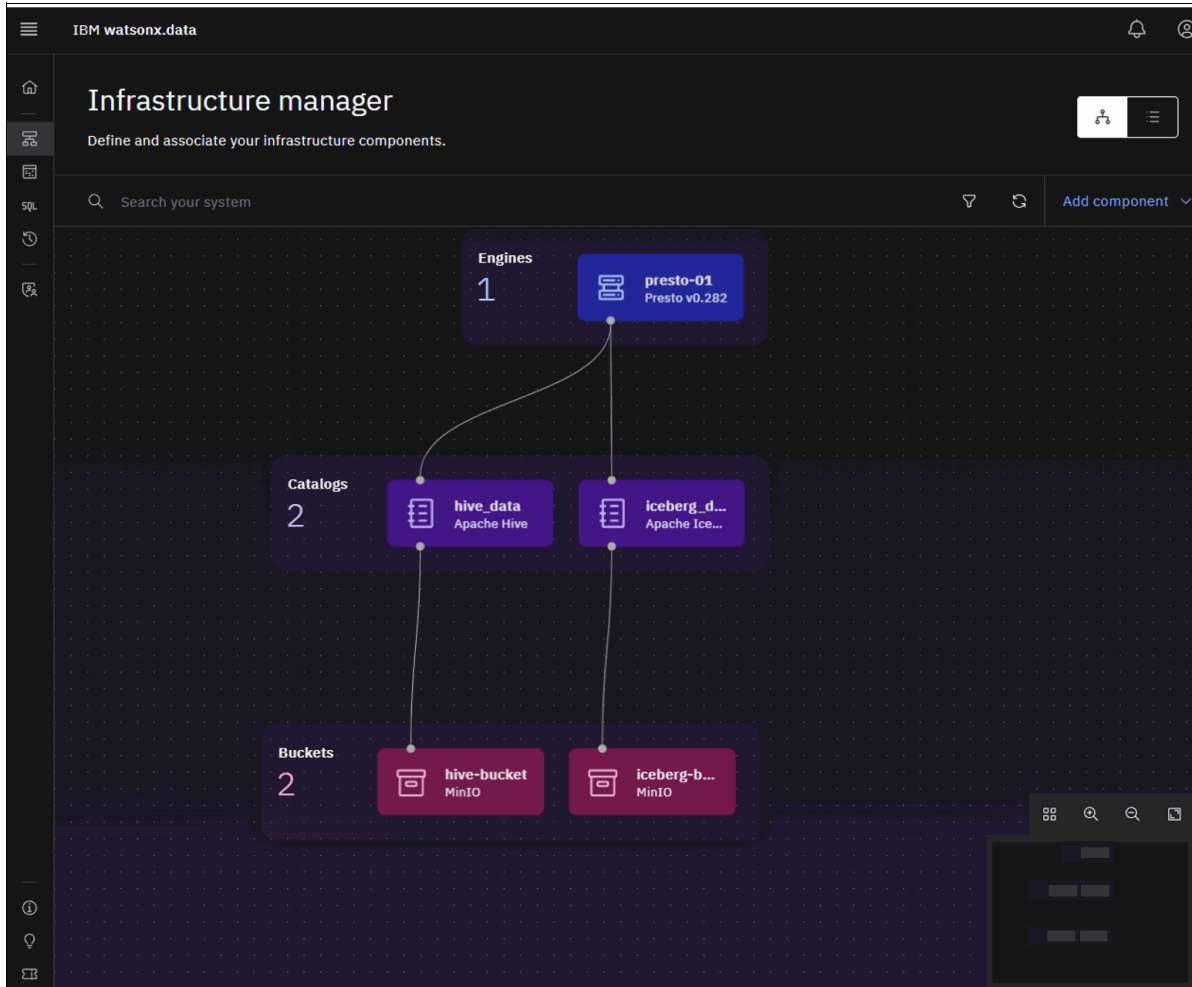


Figure 2-8 Infrastructure Manager - Add component

2. From the dropdown menu select **Add Bucket**. See Figure 2-9 on page 45.

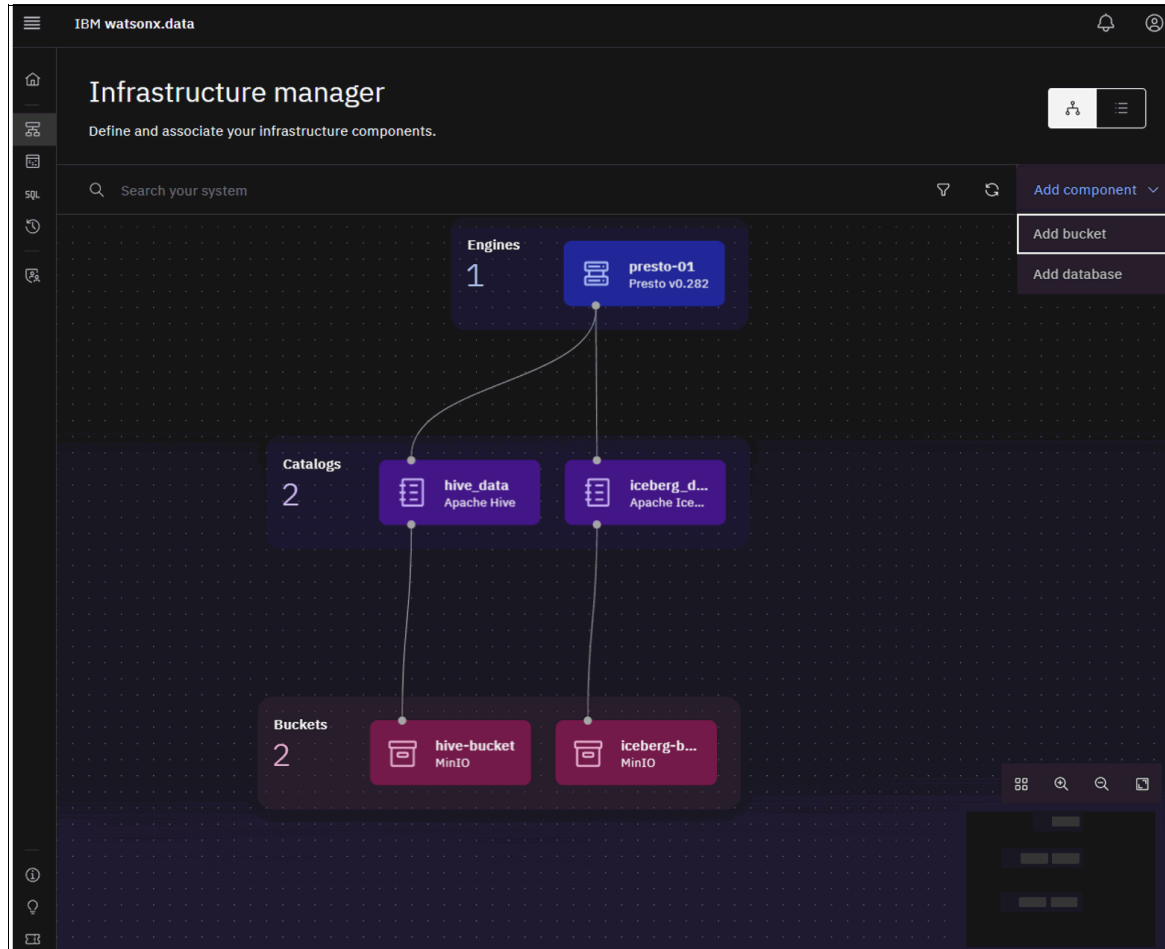


Figure 2-9 Infrastructure Manager - Add bucket

3. This will open the Add bucket prompt where you will provide information about the bucket being added to watsonx.data.
4. Select **IBM Storage Ceph** from the Bucket type dropdown menu. Then, populate the fields for the bucket name, the endpoint for the IBM Storage Ceph cluster that is being used with watsonx.data, as well as the access and secret keys. See Figure 2-10 on page 46.

### Add bucket

Register existing, externally managed object storage.

---

#### Bucket details

Bucket type	Region
IBM Cloud Object Storage ▾	Dallas (us-south) ▾
Bucket name	Display name
Example: your-bucket-01	Example: Your Bucket 01
Endpoint	
Format: https://s3.<region>.cloud-object-storage.appdomain.clo	
Access key	Secret key
Enter your access key <span>👁</span>	Enter your secret key <span>👁</span>
Connection status	
<span>🔗 Untested</span> <span style="float: right;">Test connection <span>🔗</span></span>	
Activation <input type="checkbox"/> Activate now (terminates in-flight queries against data in any bucket)	
<b>Associated catalog</b> ⓘ	
Catalog type	
Apache Iceberg ▾	
Catalog name	
Example: your_catalog_01	
Cancel	Register

Figure 2-10 Add bucket window

5. Once these steps are completed, a connection test can be performed to verify that everything is functioning properly. See Figure 2-11 on page 47.

**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

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 Test connection

6. Upon successful connection, a “*Successful*” message along with a green checkmark will be displayed. See Figure 2-12 on page 48.

The screenshot shows a dark-themed 'Add bucket' configuration form. 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' (dropdown menu with 'IBM Storage Ceph' selected), 'Bucket name' (text input with 'watsonx'), and 'Display name' (text input with 'watsonx'). The 'Endpoint' field contains '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. An 'Activation' section has an unchecked checkbox and the text 'Activate now (terminates in-flight queries against data in any bucket)'. The 'Associated catalog' section has an information icon, a 'Catalog type' dropdown menu with 'Apache Iceberg' selected, and a 'Catalog name' text input with the placeholder 'Example: your\_catalog\_01'. At the bottom are 'Cancel' and 'Register' buttons.

Figure 2-12 Connection test successful

7. Indicate whether the bucket should be activated now. See Figure 2-13 on page 49.

**Important:** Activating the bucket will terminate 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 Indicate whether the bucket should be activated now

- Finally, the bucket will be 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 will spend significantly less time planning queries that operate against large tables.

## 2.4.1 Associating a catalog

To associate a catalog with an engine perform the following steps:

- Navigate to the Infrastructure Manager page. Hover over the desired catalog in the upper right corner and click the button with the **connected dot** icon. See Figure 2-14 on page 50.

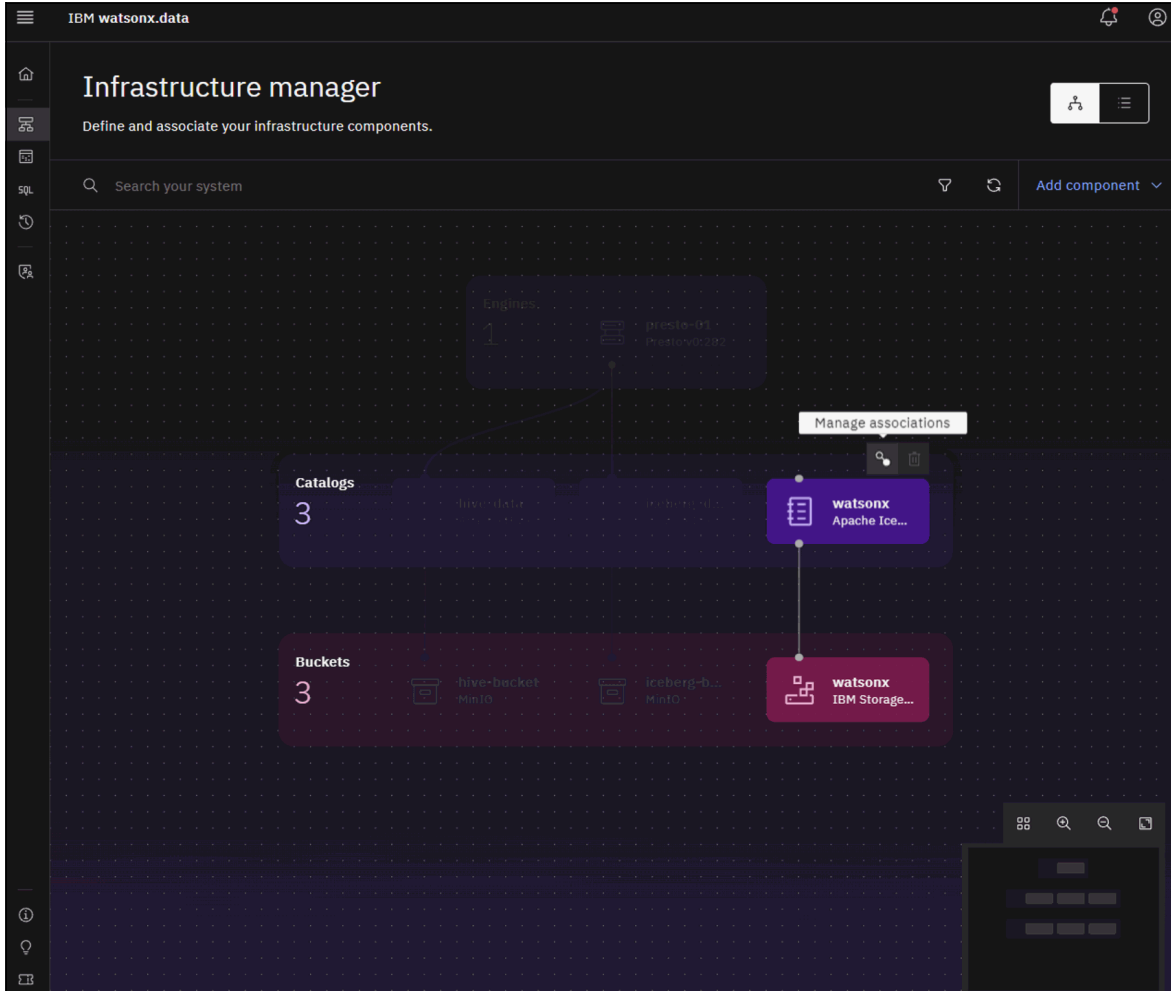


Figure 2-14 Infrastructure Manager - Manage associations

2. You will be presented with a prompt where you can select an engine to associate with the catalog. See Figure 2-15 on page 51.



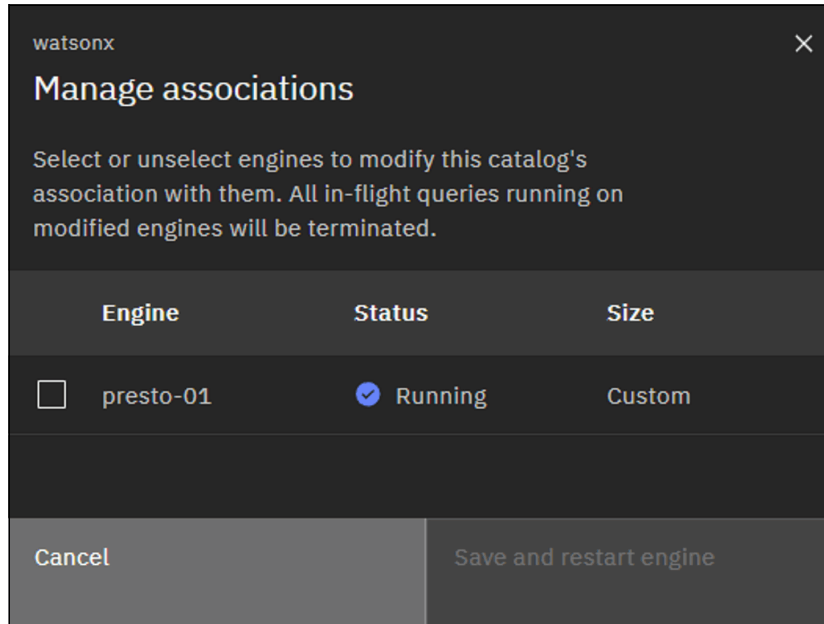


Figure 2-15 Manage associations

3. Once selected you can press the red **Save and restart engine** button.

## 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 may result in errors.

touch a

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

1. Returning to the Infrastructure Manager you can click on the **maroon bucket** icon to open the Bucket details page. See Figure 2-16 on page 52.

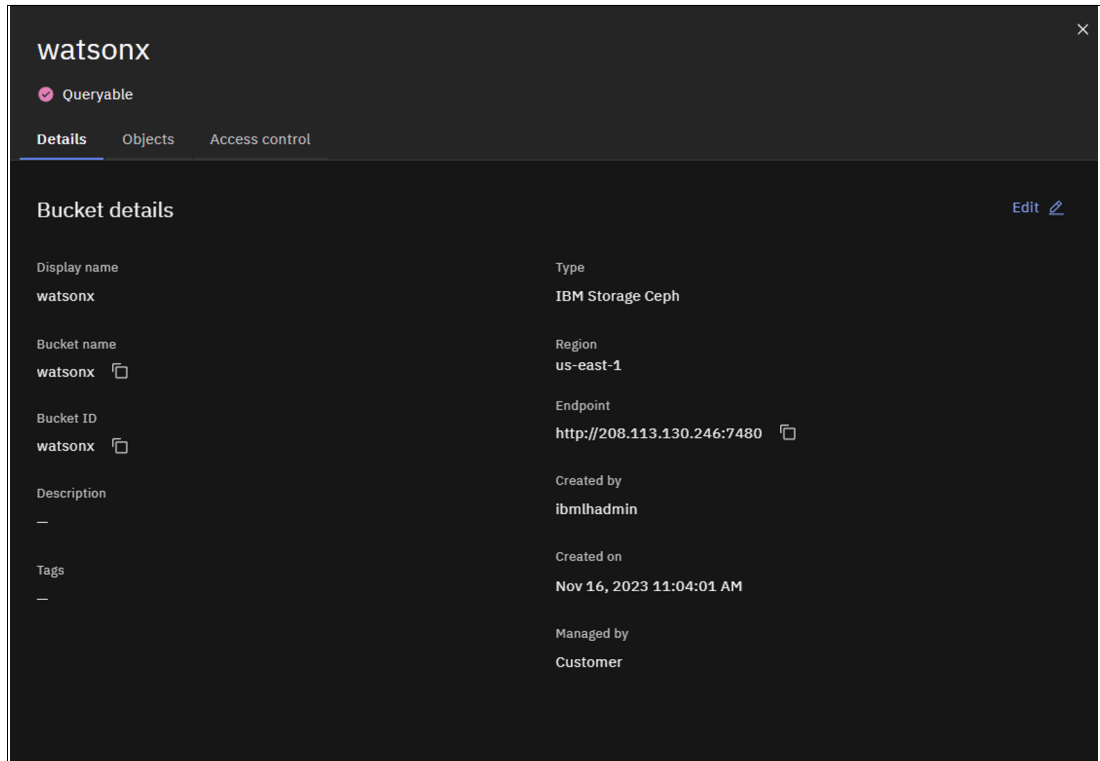


Figure 2-16 Bucket details page

2. The pseudo-directory established through the command line can be viewed by navigating to the Objects tab from the Bucket details page. See Figure 2-17.

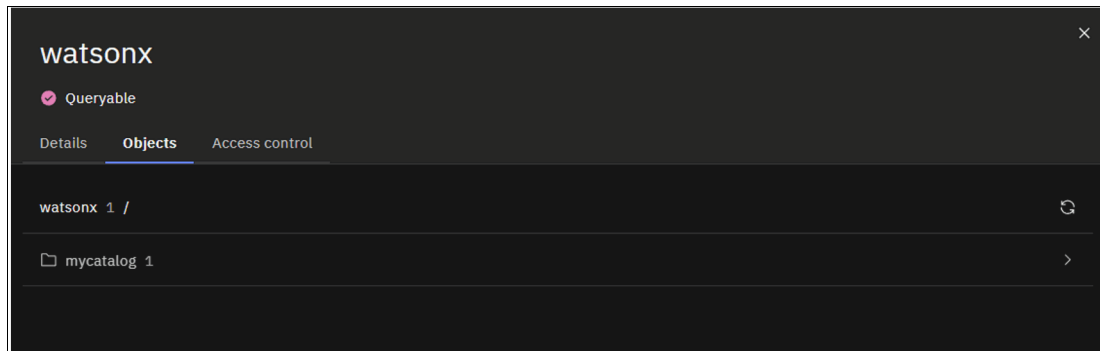


Figure 2-17 Bucket details page - Objects tab

### 2.4.3 Creating a schema

After verifying that the bucket has the necessary structure to create a schema, the Data Manager page can be used to create one.

1. Click the blue **Create** drop down menu. See Figure 2-18 on page 53.

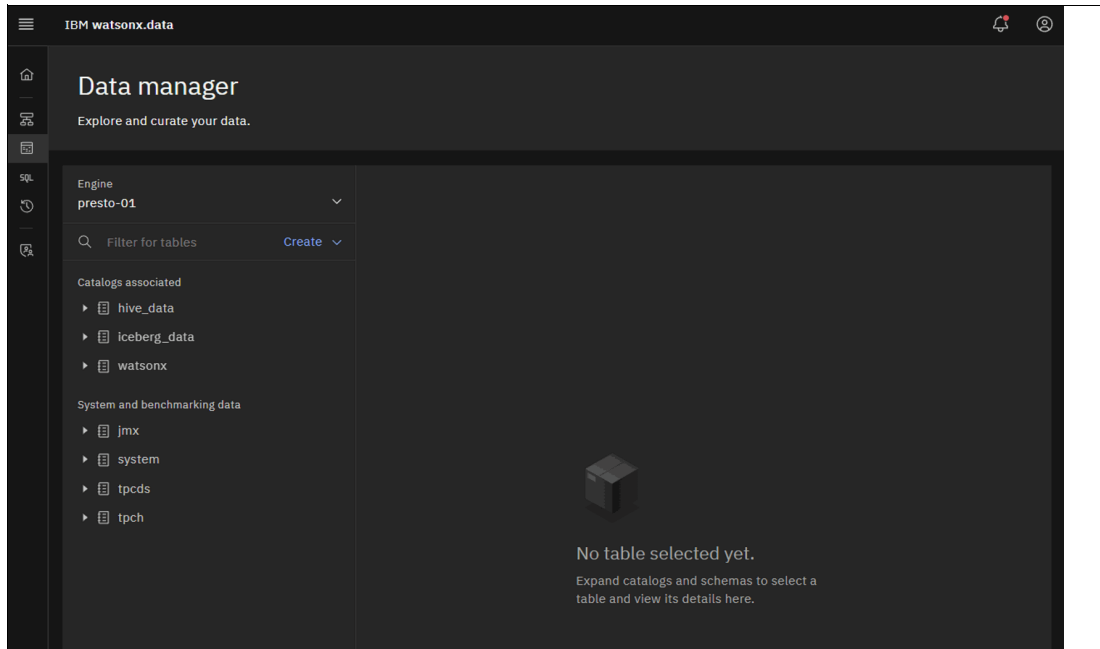


Figure 2-18 Data manager

2. To create a schema, select **Create schema** from the drop-down menu. A prompt will appear, allowing you to define your schema's details. Select the catalog associated with the IBM Ceph Storage bucket. Choose a descriptive name and use the path created in the previous section. Finally, click the blue **Create** button. See Figure 2-19.

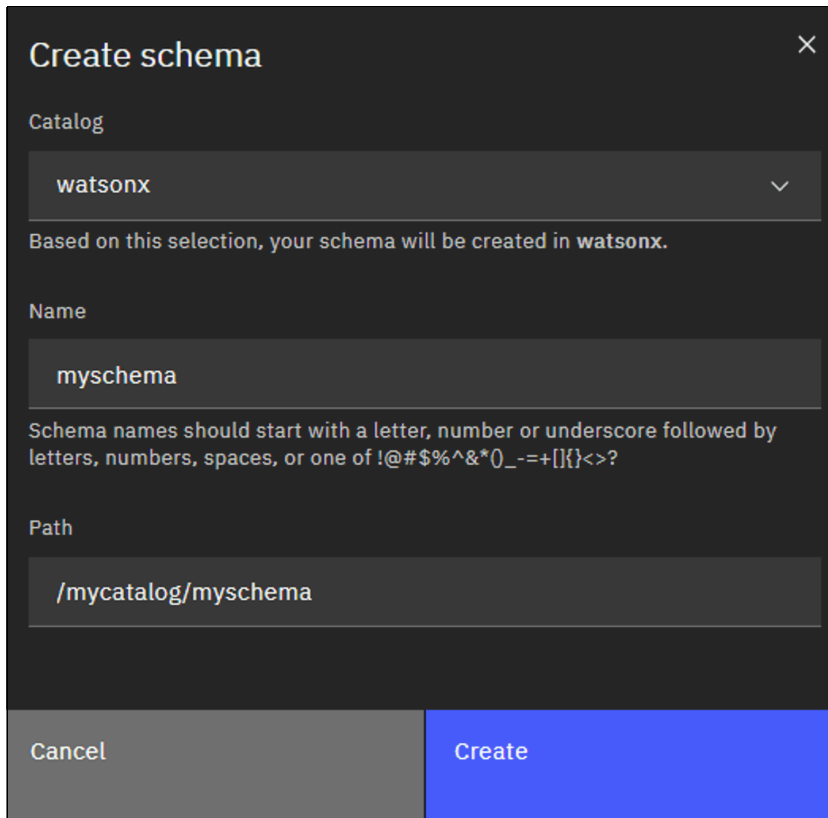


Figure 2-19 Create schema

## 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

While different engines may support a variety of formats, the most common formats are Hive and Apache Iceberg. As mentioned in the 2.4, “Adding a bucket in watsonx.data” on page 43 section, Apache Iceberg is the most well-suited table format for use with object storage. Iceberg employs manifest lists and manifest files to track which objects correspond with a particular point in time for a given 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 small number of reads against the manifests to get a list of all the objects that need to be operated on. When an engine is planning a query for execution with a Hive table, it needs to recursively list the pseudo-directory structure 1,000 keys at a time. *For large tables, the Hive approach can take significantly longer and put more load on the underlying object store. In fact, this was an influential factor in the design of Apache Iceberg.*

### 2.5.2 Columnar formats

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

### 2.5.3 Partitioning

Partitioning tables based on the values in a specific column can significantly reduce query processing time, particularly for queries involving predicate filtering (WHERE statements). A common practice is to partition tables containing order or sales data by a specific time period, such as day, month, or year. This approach enables query engines to eliminate irrelevant partitions, minimizing the number of reads from the storage system and consequently 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 not relevant to the query are pruned, resulting in fewer reads to the storage system and subsequently faster query times. The downside of bucketing is that data cannot be inserted into the table by introducing new objects. Instead, buckets need to 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 by storing partition statistics in table metadata. In other words, statistics are calculated when writing to Iceberg tables. This has clear advantages over periodic table and column analysis.

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

It is not uncommon to have tools writing a large volume of data in formats that are less optimized for analysis. The most common suspect in this category is CSV. Many of the techniques from 2.5, “Performance optimizations” on page 54 are not possible when dealing with CSV data. Some query engines, notably Presto, are well prepared to 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 not only substantially reduces the volume of data transferred over the network but also optimizes the computational resources required for query execution.

The Hive session property `s3_select_pushdown_enabled` is required to make use of `S3-Select` pushdown. It is worth noting that Hive session properties can only be used when submitting queries via the CLI. If a hive session parameter is set in the Query workspace, a successive statement will be 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 not only against accidental deletion, but 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

Lifecycle configuration allows a set of rules to be defined that trigger actions on objects by the object store. Supported actions include transitioning the object from one storage class to another and expiration. IBM Storage Ceph supports the configuration of multiple storage classes that place data on different types of storage media or using different durability strategies (that is, replication versus erasure coding). Using lifecycle can result in significant 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's 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, AMQP0.9.1, and Kafka. To establish bucket notifications, users must create a notification entry that specifies the bucket, topic, and event types to monitor. Additionally, 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 via the bucket notification mechanism, trigger a notification event from the synced RGW upon successful object synchronization. It's important to note that 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 İşbank, where they have implemented Object Storage as a Service offering using IBM Storage Ceph in their private cloud computing infrastructure. This chapter has the following sections:

- ▶ “Company profile” on page 58
- ▶ “Empowering enterprise storage: Object Storage as a Service in private cloud environments with IBM Storage Ceph” on page 58
- ▶ “Sample implementation” on page 60

## 3.1 Company profile

İşbank (<https://www.isbank.com.tr/en>) was the first national bank of Türkiye, 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, the bank has received, Euromoney named İşbank *Central and Eastern Europe's Best Digital Bank*<sup>1</sup>, while it 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

In this section, we will discuss object storage concepts, business challenges, and how an *Object Storage as a Service (OaaS)* 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, 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.

Object Storage as a Service (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:

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

In addition to these challenges, enterprises must also:

<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>



- ▶ 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 in order to stay competitive and meet their evolving storage needs effectively.

### 3.2.3 Benefits of the approach

Leveraging an Object Storage as a Service (OSaaS) model with a strong focus on self-service capabilities in an on-premises private cloud environment will enable enterprises to overcome the persistent storage challenges they face.

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

A self-service model also ensures cost optimization, as organizations can closely align storage resources with actual usage, eliminating waste. Additionally, 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 our use case scenario with İşbank, the architectural design is centered around delivering a comprehensive self-service model within the on-premises environment. This model encompasses a range of essential functionalities, empowering users to efficiently manage their storage resources without requiring administrative assistance.

Users are able to perform critical tasks such as:

- ▶ 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.

Additionally, robust documentation (see “The crucial role of documentation in self-service way” 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. This promotes enhanced data control and operational efficiency within the organization.

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

- ▶ *Empowered users*: Users are able to manage their own storage resources without requiring administrative assistance.
- ▶ *Increased efficiency*: Self-service reduces administrative overhead and allows 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 specifically 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 on 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 way

Documentation plays a pivotal role to treat your offerings as a product. It will ensure clarity, consistency, and compliance throughout the self-service storage management process. It aims to provide comprehensive guidance, step-by-step instructions, and best practices for end-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 described in the architectural model and how it is implemented in İşbank are summarized below with screenshots.

### 3.3.1 Creating a user

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

1. Whenever a new user is required, calling the catalog item will consume IBM Storage Ceph's API and create the required user. (Figure 3-1.)

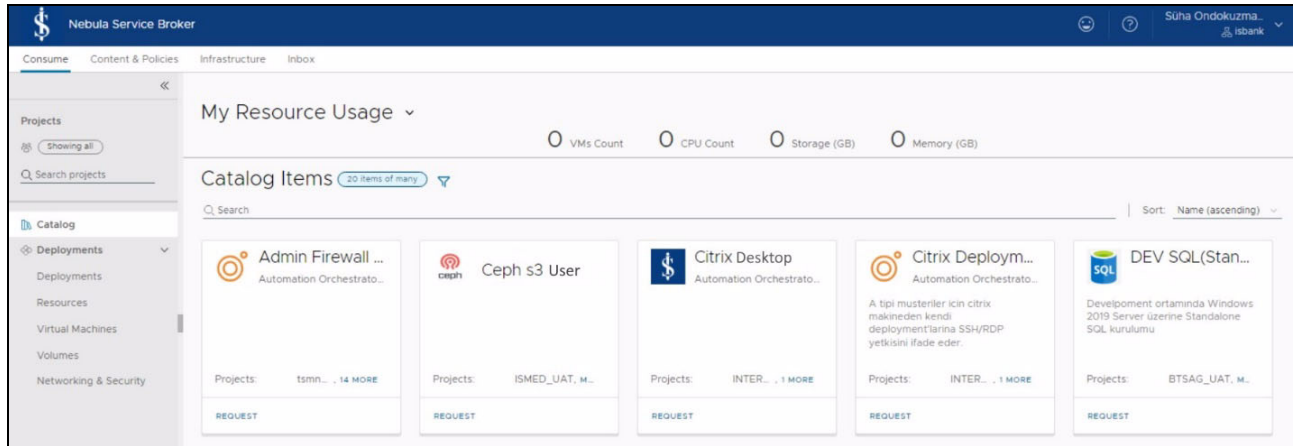


Figure 3-1 Sample portal view with a S3 user catalog item

2. The end user making the request can view user-related information on the portal and also have the information transmitted via email (Figure 3-2 on page 61). The sample integration at İşbank has been developed using Python and Javascript.

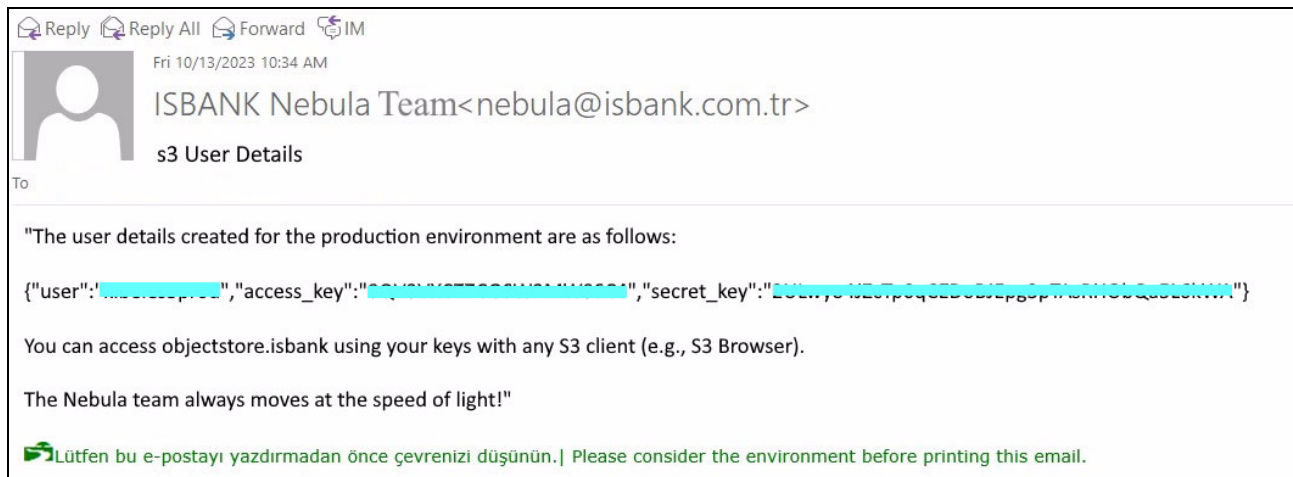


Figure 3-2 The requester can receive an email containing the keys to access the environment

### 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.

1. 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).

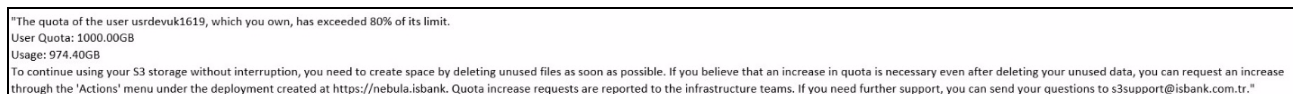


Figure 3-3 Sample alerting by email when a threshold is exceeded

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

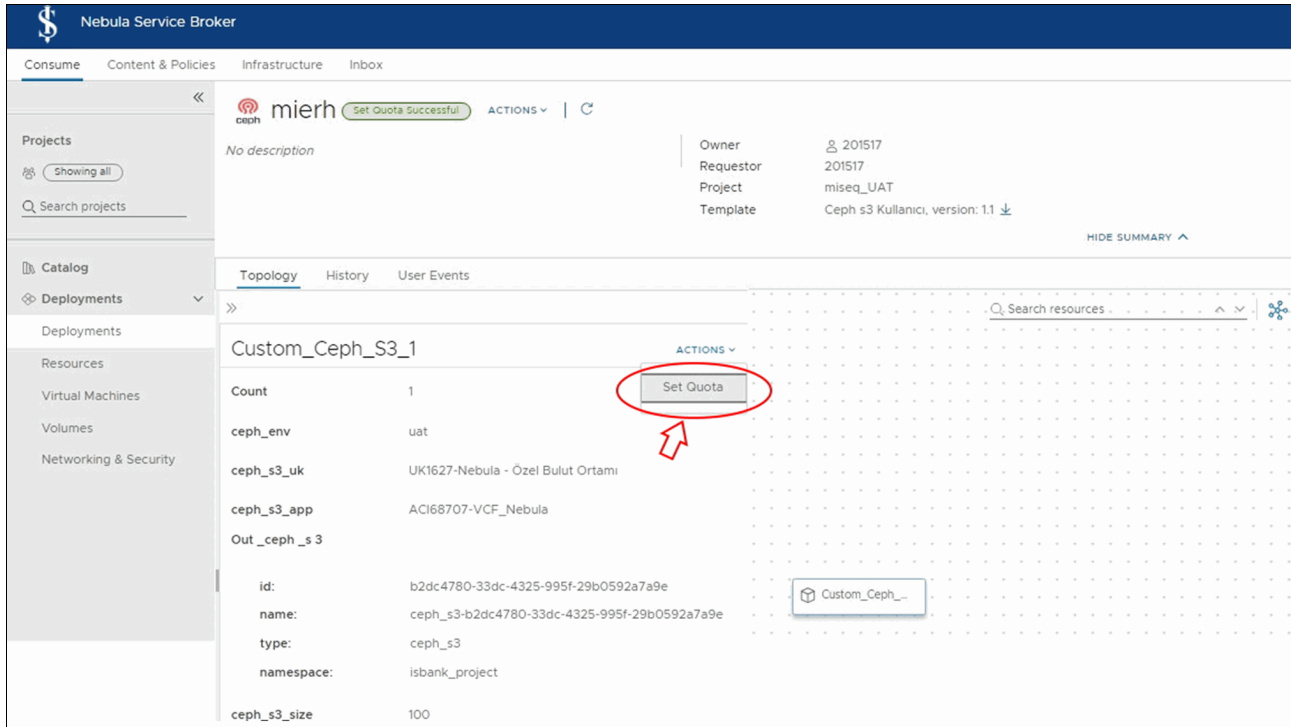


Figure 3-4 A sample view of the catalog item used during quota updates

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

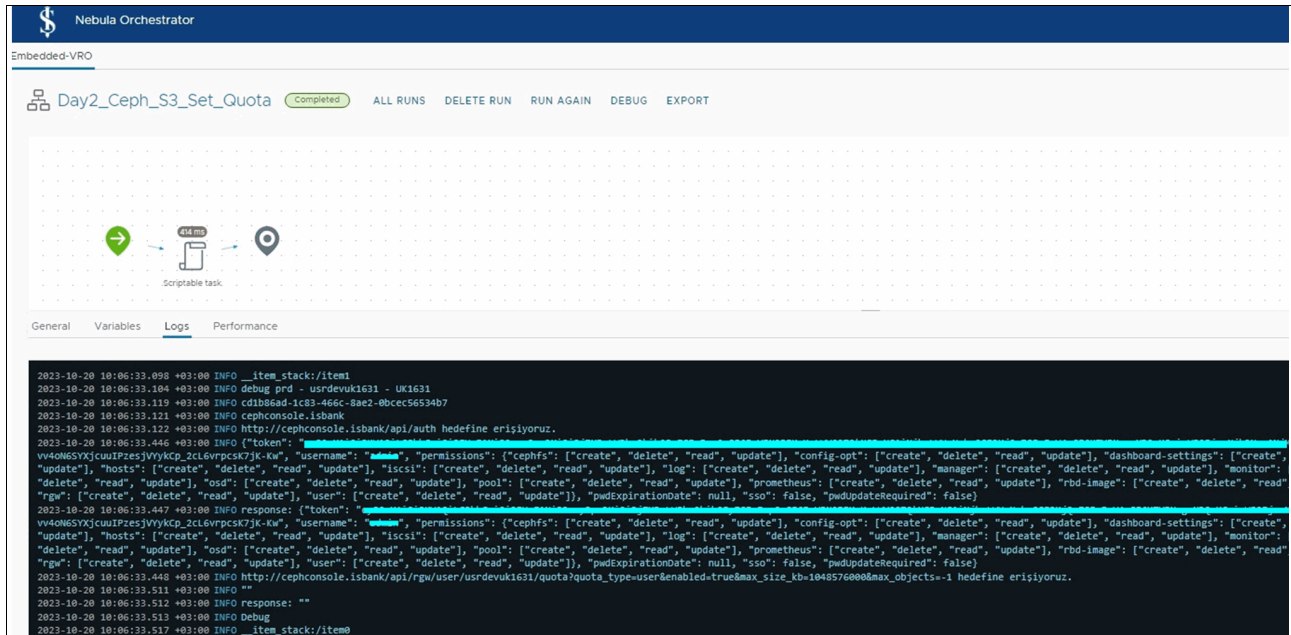


Figure 3-5 IBM Storage Ceph API response

### 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 assist in 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 will initiate the collection of detailed metrics with the option to use custom scrape intervals. (Figure 3-6 and Figure 3-7).

**Tip:** Prometheus Manager module needs to be restarted after changing its configuration.

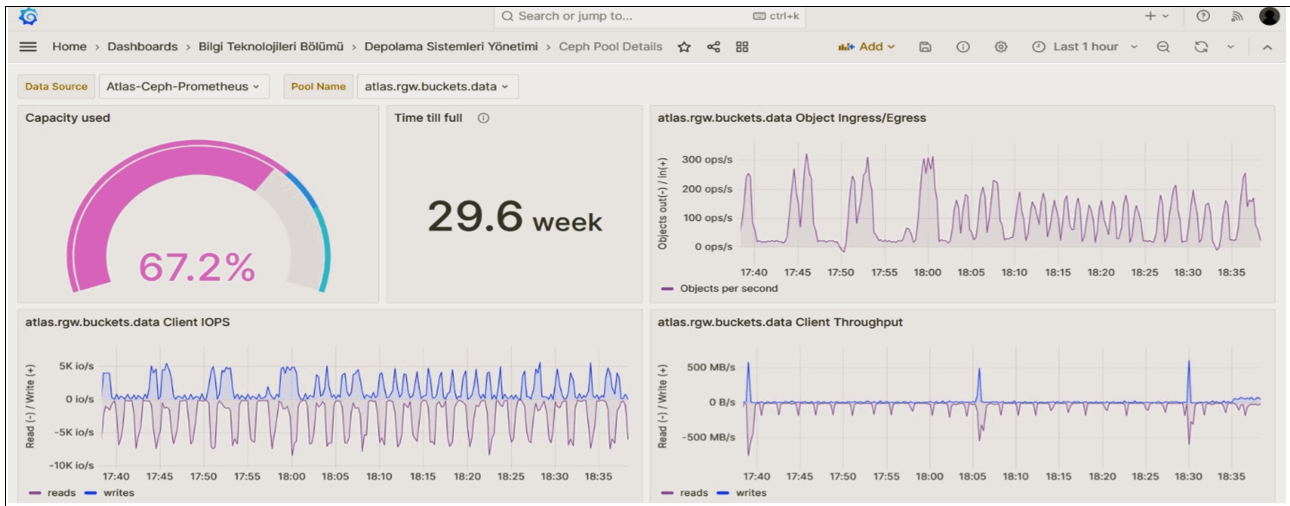


Figure 3-6 Sample Grafana dashboard showing one of the pools usage and its performance

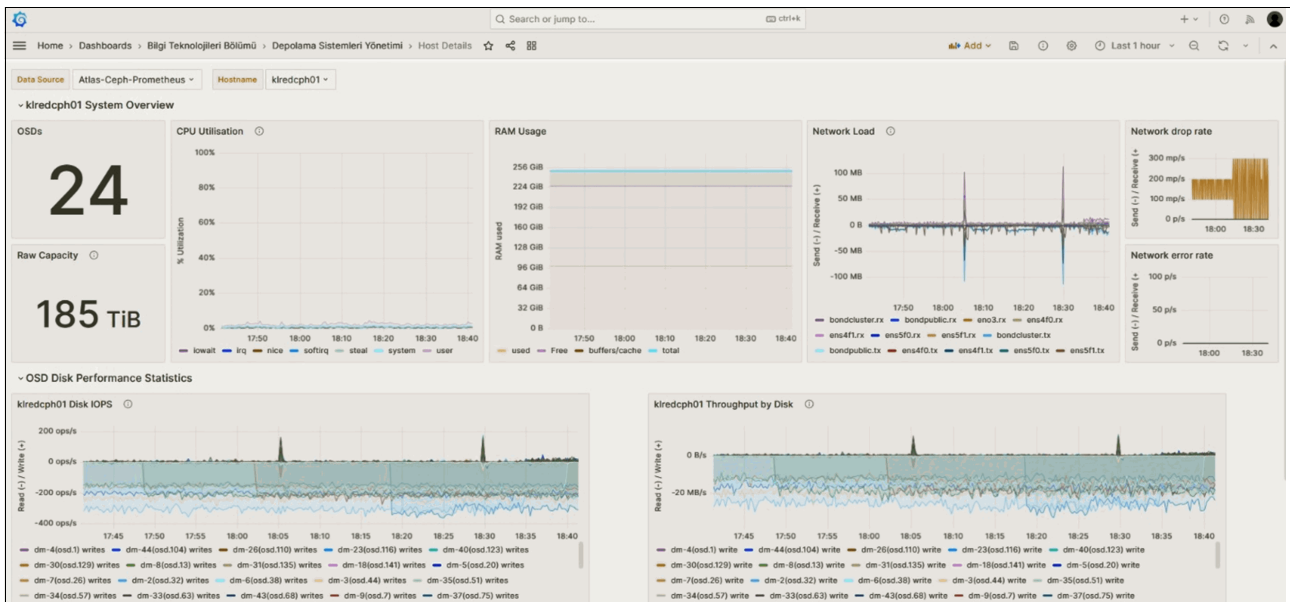


Figure 3-7 Sample Grafana dashboard showing one of the pools usage and its performance

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

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

Figure 3-8 The global config 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 parsed from the logs can be filtered and viewed through the Kibana interface (Figure 3-9 on page 64).

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

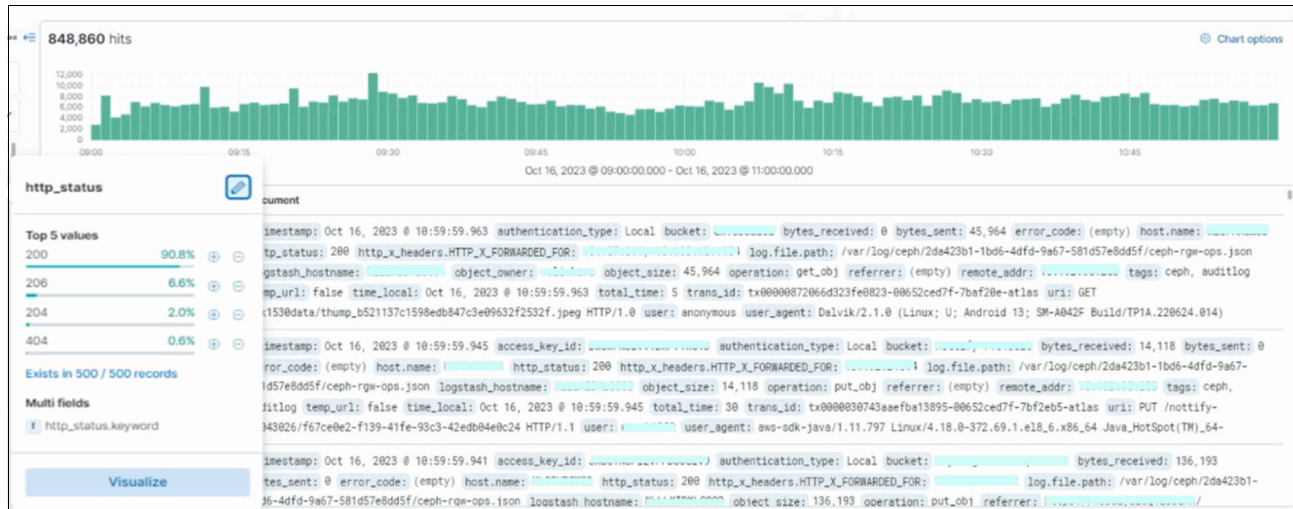
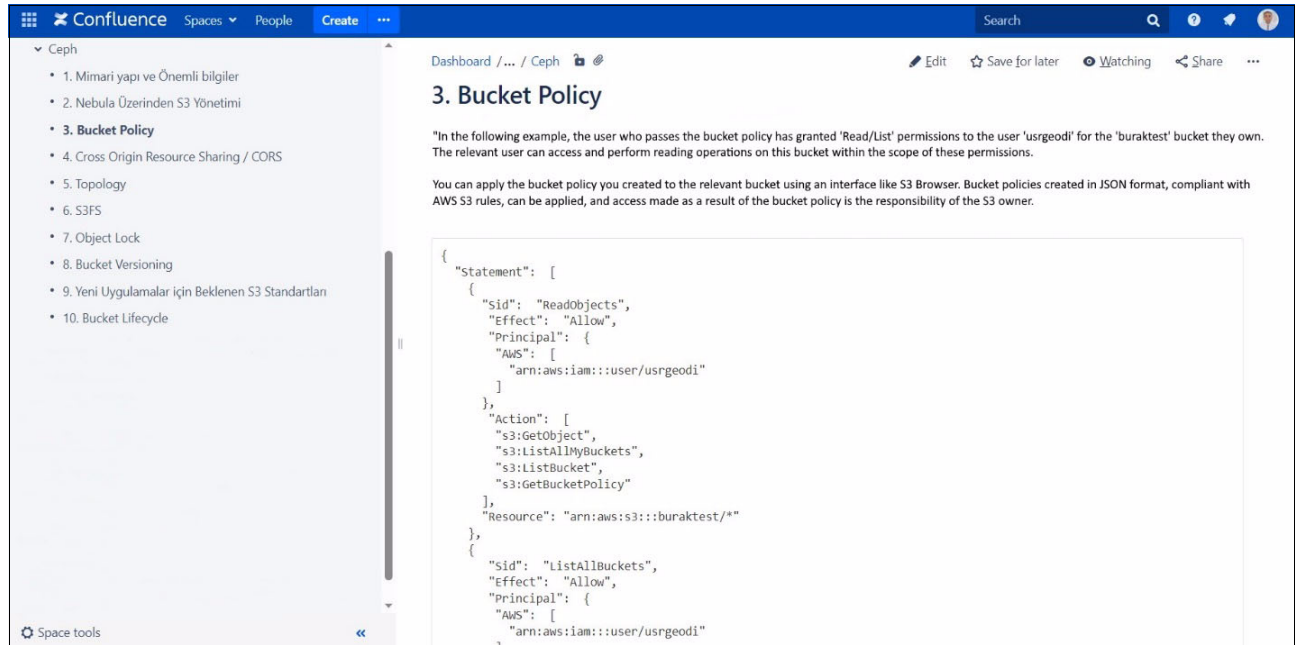


Figure 3-9 Kibana view of the "IBM Storage Ceph" index pattern

### 3.3.4 Documentation

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

Figure 3-10 on page 65 shows a sample documentation view detailing all the required information for the environment.



The screenshot shows a Confluence page titled "3. Bucket Policy" under the "Ceph" space. The left sidebar contains a table of contents with 10 items, where "3. Bucket Policy" is highlighted. The main content area has a heading "3. Bucket Policy" and two paragraphs of text. The first paragraph explains that a user with 'Read/List' permissions can access and perform reading operations on a bucket. The second paragraph states that bucket policies can be applied via an interface like S3 Browser and are in JSON format compliant with AWS S3 rules. Below the text is a code block containing a JSON policy snippet.

```
{
  "Statement": [
    {
      "Sid": "ReadObjects",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::user/usrgeodi"
        ]
      },
      "Action": [
        "s3:GetObject",
        "s3:ListAllMyBuckets",
        "s3:ListBucket",
        "s3:GetBucketPolicy"
      ],
      "Resource": "arn:aws:s3:::buraktest/*"
    },
    {
      "Sid": "ListAllBuckets",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::user/usrgeodi"
        ]
      },
      "Action": [
        "s3:ListBuckets"
      ],
      "Resource": "*"
    }
  ]
}
```

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







# 4

## **Disaster recovery and backup and archive: IBM Storage Ceph as an S3 Backup Target**

In this chapter we use Veritas NetBackup™, a backup and recovery product, as an example to demonstrate how it can use IBM Storage Ceph as a backup target.

This chapter has the following sections:

- ▶ “Introduction” on page 68
- ▶ “Implementing Veritas NetBackup with IBM Storage Ceph” on page 69
- ▶ “IBM Storage Ceph multi-site replication with Veritas NetBackup” on page 70
- ▶ “IBM Storage Ready Nodes for Ceph” on page 75

## 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

In this chapter we use Veritas NetBackup, a backup and recovery product, as an example 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:

### 4.1.2 Scalability and flexibility

IBM Storage Ceph is a massively scalable object storage software, allowing 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 enables the addition of inexpensive industry-standard servers as needed for performance and capacity requirements. IBM Storage Ceph is highly available and resilient out of the box, with default configurations able to withstand loss of multiple nodes without compromising availability.

### 4.1.3 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 three-way replication and erasure coding with different profiles: 4+2, 8+3, 8+4 etc. In addition, Ceph Object Gateway supports multi-site active/active replication for disaster recovery.

### 4.1.4 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 allows organizations to take advantage of 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 decreased hardware footprint. Thus, 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:** In this chapter we discuss IBM Storage Ceph, but this scenario also applies to Red Hat Ceph.

You need to configure Veritas NetBackup to use IBM Storage Ceph as the cloud storage provider using Amazon S3 API. See Figure 4-1 on page 69. Veritas NetBackup version 10.1.1 or later supports IBM Storage Ceph as a backup target via the S3 application programming interface (API). A list of S3 cloud storage vendors certified for NetBackup can be found [here](#).

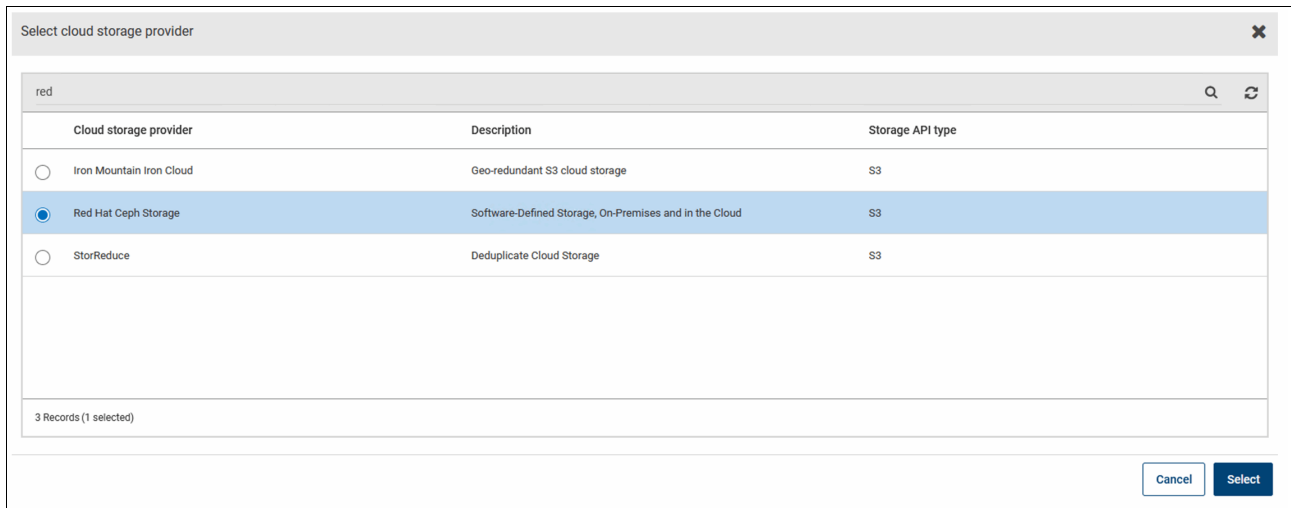


Figure 4-1 Select cloud storage provider

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

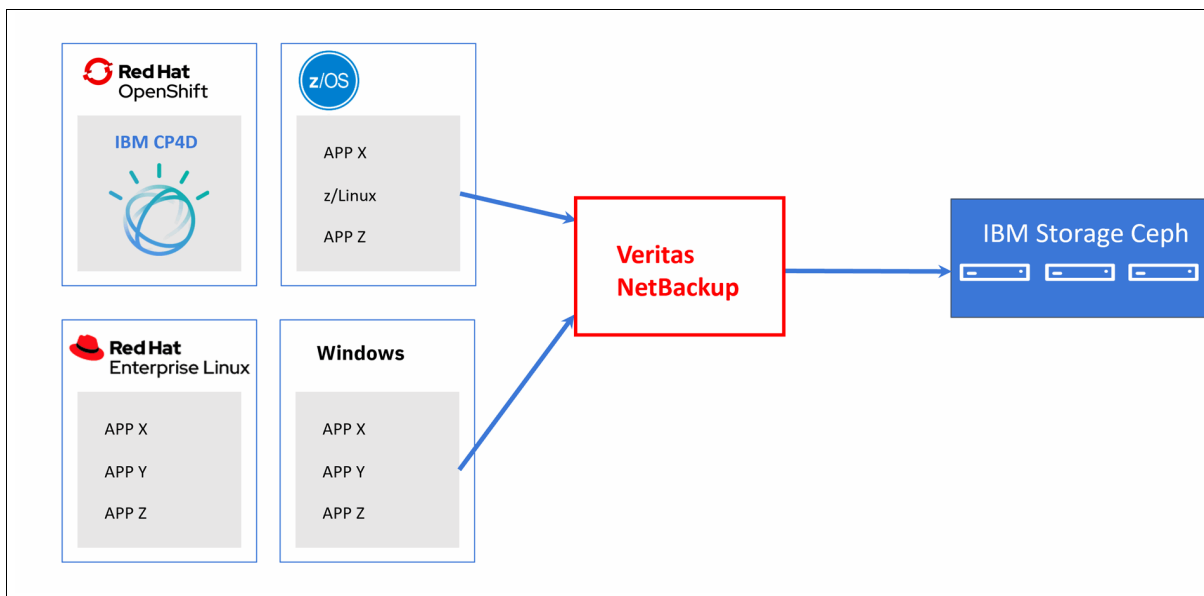


Figure 4-2 Veritas NetBackup with IBM Storage Ceph for a single site

Organizations can also implement IBM Storage Ceph 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 Prod IBM Storage Ceph cluster is back online, the data will be synchronized, and the load balancer will failback the access to the Prod IBM Storage Ceph cluster.

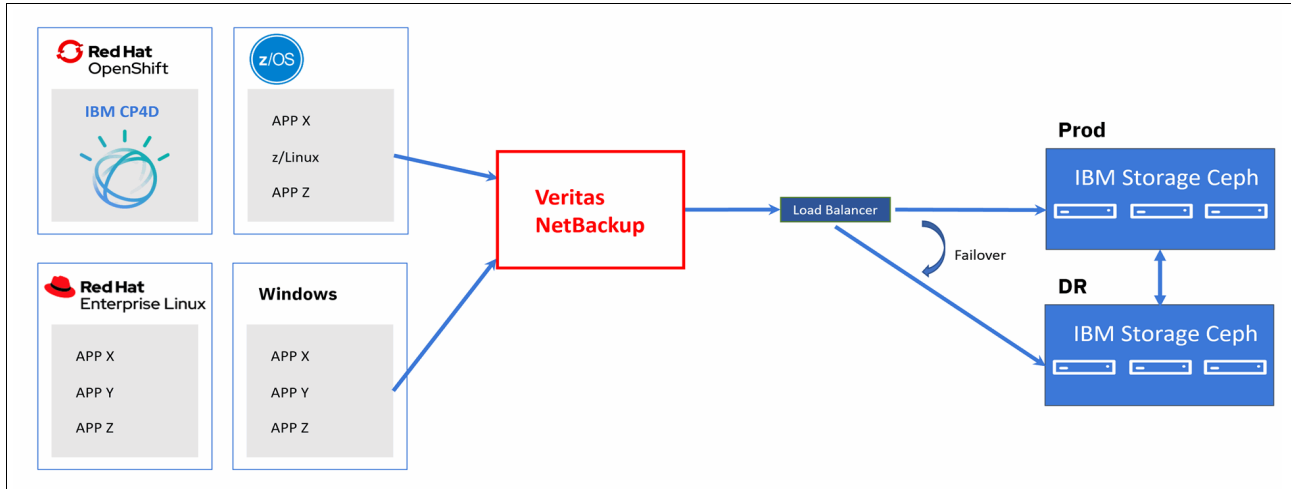


Figure 4-3 IBM Storage Ceph multi-site replication with Veritas NetBackup

### 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, while IBM Storage Ceph is a scalable and reliable self-healing storage platform for storing enterprise data. The compatibility between these two solutions ensures a 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, used, available and total capacity of IBM Storage Ceph clusters. 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 is a high available Storage Ceph setup for Veritas NetBackup. Two Ceph clusters are configured with multi-site RGW replication fronted by a HAProxy server for Veritas NetBackup. Configuration of Ceph multi-site RGW replication is available [here](#).

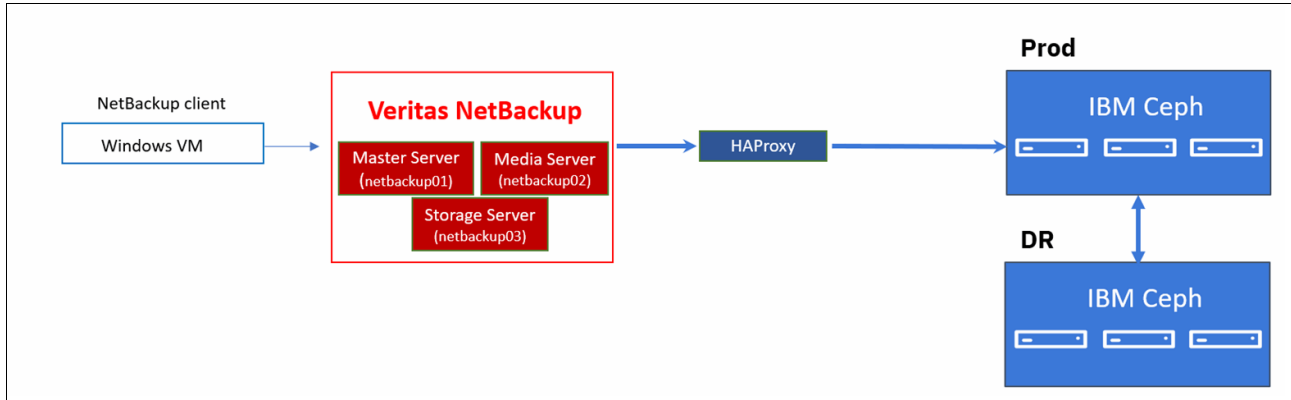


Figure 4-4 Two Ceph clusters are configured with multi-site RGW replication fronted by a HAProxy server for Veritas NetBackup

### 4.3.1 Configuration of Veritas NetBackup components

The Veritas NetBackup installation guides are available [here](#).

Install and setup the following components:

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

Perform the following steps to configure storage server for IBM Storage Ceph:

1. Click **Add storage server**. See Figure 4-5.

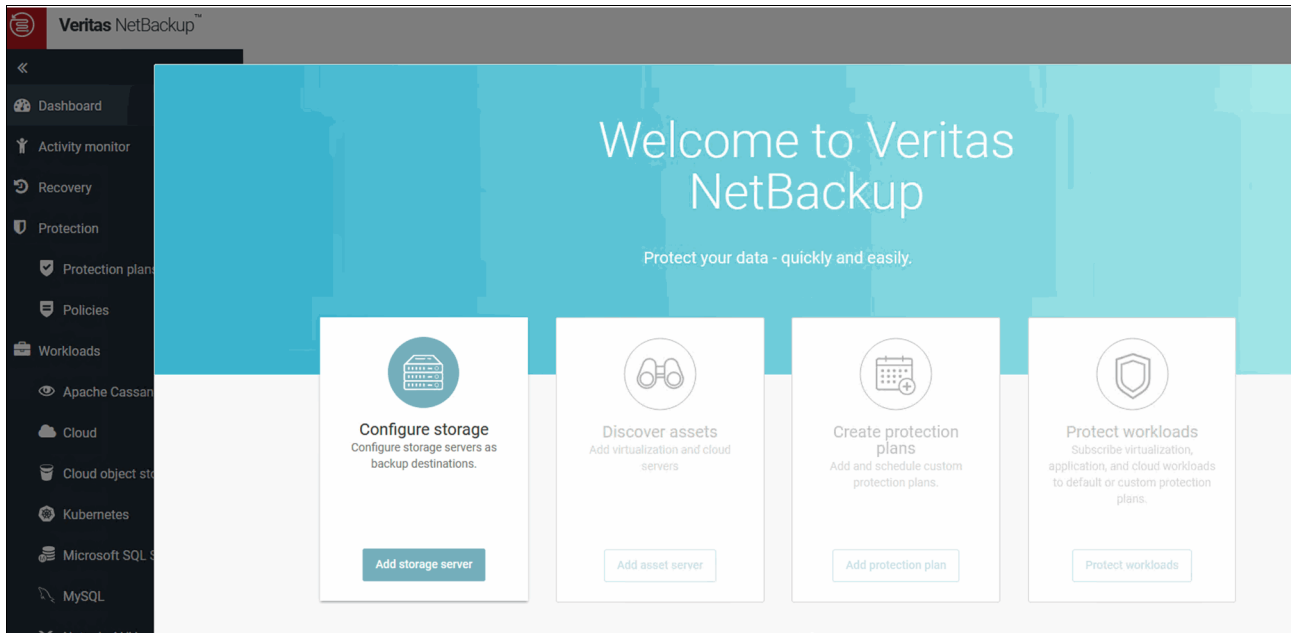


Figure 4-5 Add Storage server

2. Select **Cloud storage** and click **Start**. See Figure 4-6 on page 72.

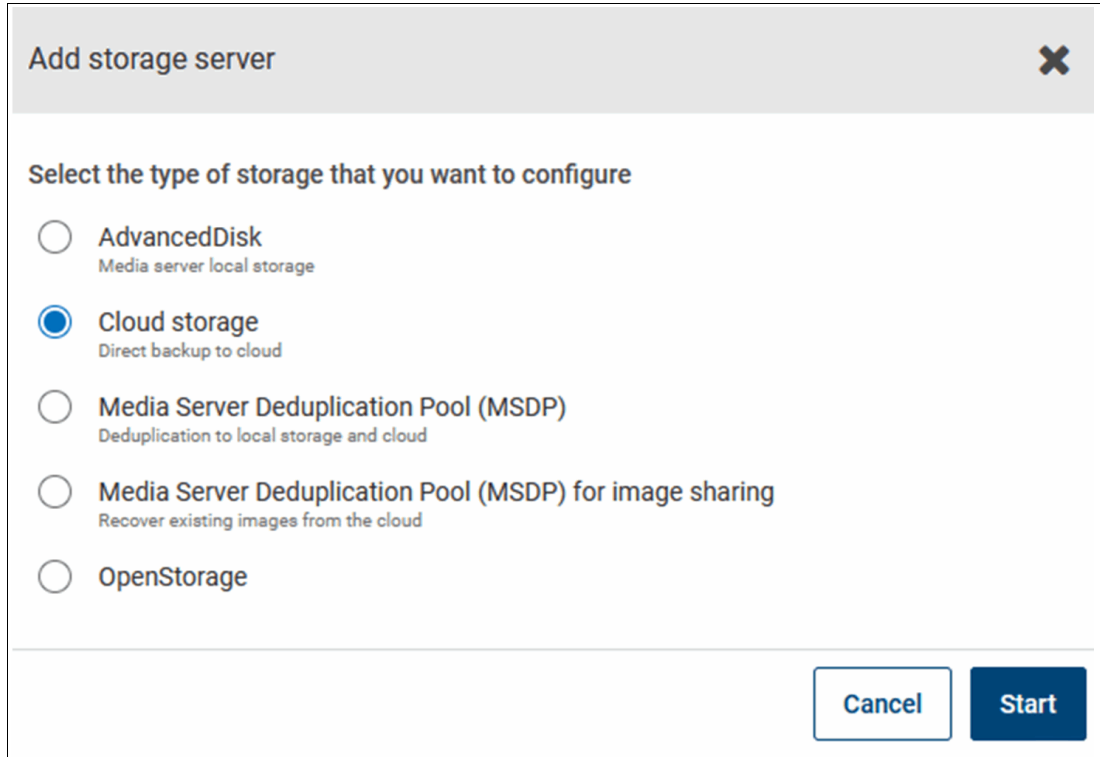


Figure 4-6 Add storage server

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

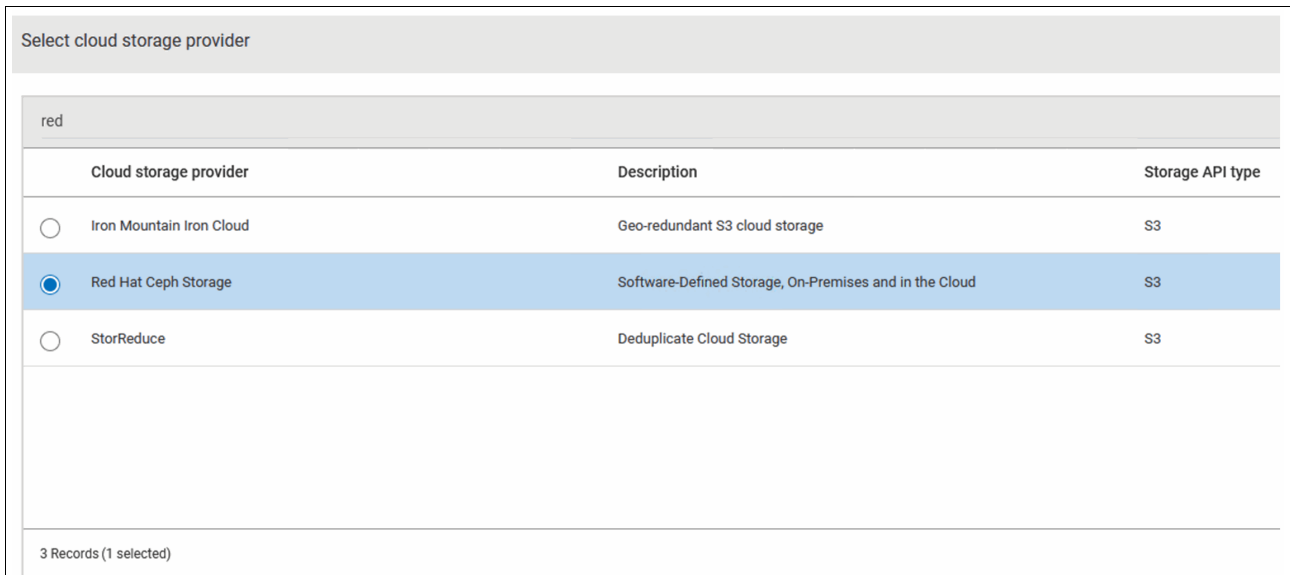


Figure 4-7 Select cloud storage provider

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 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**. See Figure 4-8 on page 73.

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

- Region name \***: A text input field containing "multizg".
- Location constraint \***: A text input field containing "multizg".
- Service URL \***: A text input field containing "haproxy.testlab.net".
- Endpoint access style**: A dropdown menu currently showing "Path style".
- HTTP port \***: A dropdown menu currently showing "80".
- HTTPS port \***: A dropdown menu currently showing "443".

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

Figure 4-8 Add a region

5. The region is added, as shown in Figure 4-9 on page 74. Click **Next**.

The screenshot shows the 'Add cloud storage server' dialog box with the 'Basic properties' step selected. The form contains the following fields and data:

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

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

Buttons: Cancel, Previous, Next

Figure 4-9 Add the region name

6. In the next window, as shown in Figure 4-10, enter the access key and secret access key and click **Next**. This is the access key and secret access key of the Ceph S3 user.

The screenshot shows the 'Add cloud storage server' dialog box with the 'Access settings' step selected. The form contains the following fields and options:

- Access details for the account
  - Access key ID \*: password
  - Secret access key \*: [masked]
- Advanced settings
  - Security
    - Use SSL
  - Proxy
    - Use proxy server

Buttons: Cancel, Previous, Next

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

7. NetBackup divides the backup data into chunks called objects. The performance of NetBackup to S3 storage is determined by the combination of object size, number of parallel connections and the read or 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**. See Figure 4-12 on page 75.

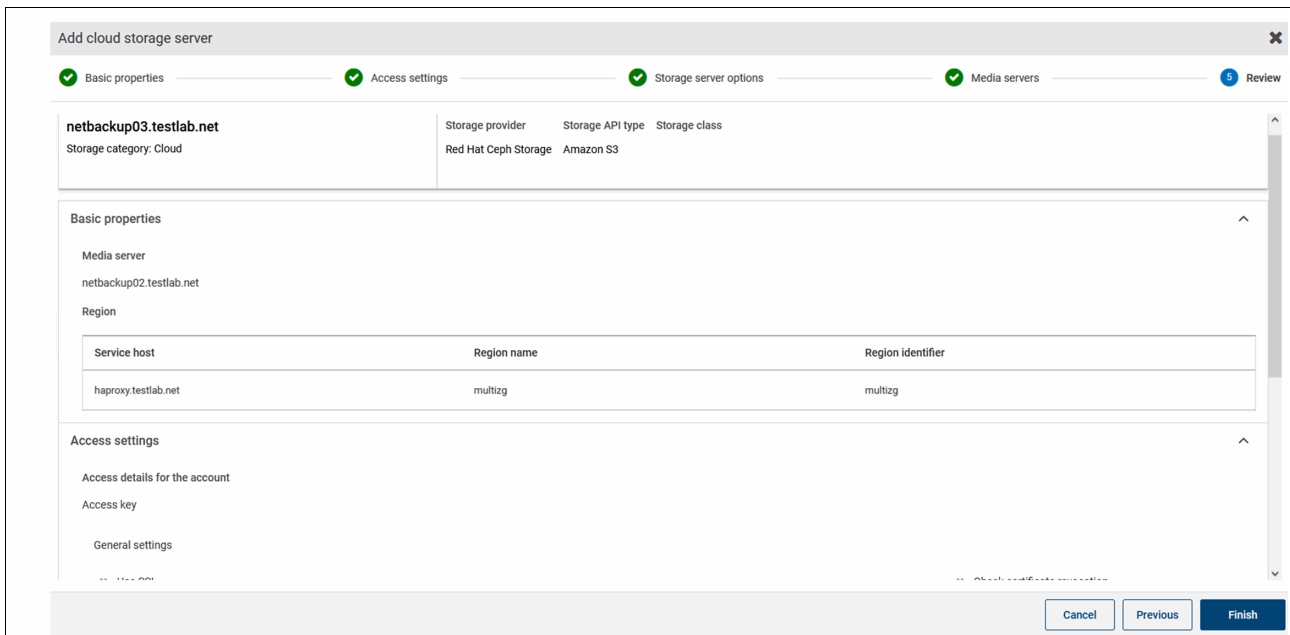


Figure 4-12 Click Finish to complete the configuration

## 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. That means customer only have a single support line to call for both hardware and software. Ready Nodes for Ceph comes with 4 profiles with different HDD disk sizes: 8 TB, 12 TB, 16 TB and 20 TB. See Figure 4-13 on page 76.

	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.

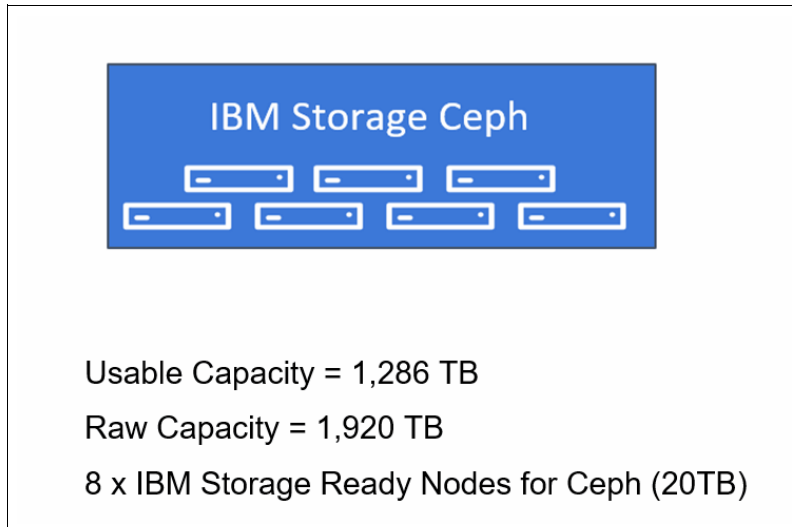


Figure 4-14 Example hardware configuration of a 1.2 PB IBM Storage Ceph cluster

#### 4.4.1 IBM Storage Ceph Editions

IBM Storage Ceph offers two editions. Both editions can be used to implement a 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 Red Hat Enterprise Linux (RHEL) server operating system software license and therefore organizations have the flexibility to bring their own or use their existing RHEL licenses.





# 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. In this chapter, we will show you some real-world use cases where we use bucket notifications.

This chapter has the following sections:

- ▶ “Introduction” on page 80
- ▶ “Cloud native data pipelines versus legacy data pipelines” on page 80
- ▶ “S3 bucket notifications: Components and workflows” on page 82
- ▶ “Event-driven data pipeline example, powered by the S3 bucket notification feature” on page 83
- ▶ “Step by step basic example of configuring event notifications for a bucket in IBM Storage Ceph” on page 85
- ▶ “Configuring S3 bucket notifications in OpenShift using the S3 RadosGW object storage provided by Fusion Data Foundation” on page 89

## 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 take action. S3 bucket notifications can be very useful for a variety of tasks, such as:

- ▶ Automating workflows: For example, you could 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 could use S3 bucket notifications to monitor changes to your data and send alerts if something unexpected happens.
- ▶ Auditing: You could use S3 bucket notifications to log all changes to your data so that you can track who made what changes and when.

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

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

## 5.2 Cloud native data pipelines versus legacy data pipelines

In this section we compare 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 an NFS (Network File System) or CIFS (Common Internet File System) network file system share from a storage server, and the same shared network file system would need to be mounted by a dedicated application server that processes the incoming data in batch mode. See Figure 5-1.

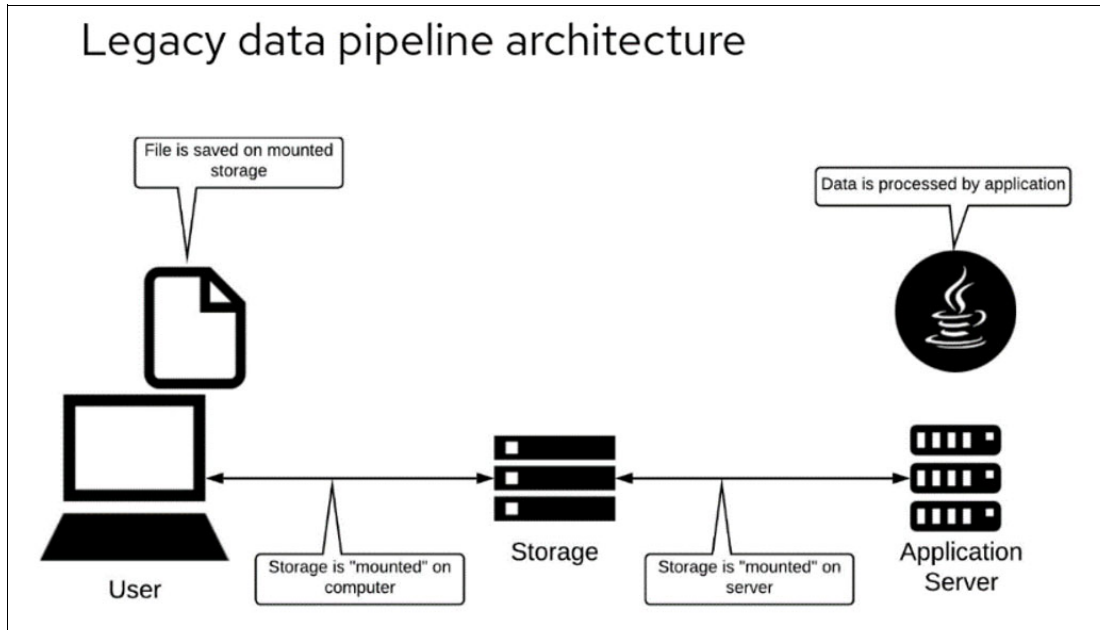


Figure 5-1 Legacy data pipeline architecture pattern

### 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 using simple HTTP/S communication through the S3 API. The S3 HTTP endpoint is accessible from anywhere with network connectivity to port 443.

We can then use the bucket notification feature in IBM Storage Ceph to send a message to a Kafka topic whenever a bucket is uploaded or deleted. The topic can then be consumed by a serverless OpenShift service like Knative, which can scale from 0 to the number of compute services needed to process the ingested data in the S3 object store. See Figure 5-2 on page 81.

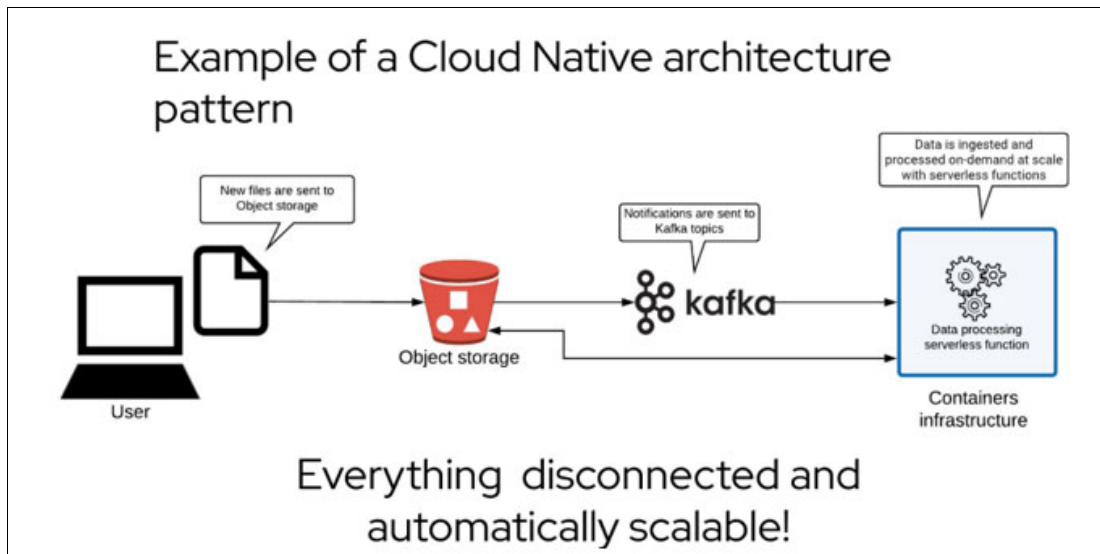


Figure 5-2 Cloud-native architecture pattern

## 5.3 S3 bucket notifications: Components and workflows

In this chapter will cover S3 bucket notifications.

### 5.3.1 Bucket notification workflow

As mentioned, bucket notifications provide a mechanism for sending information out of Ceph via the RADOS Gateway 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, we need one of those endpoints configured and accessible from the Ceph nodes where the RGW services are running.

Next, we create a topic in RGW that points to the preconfigured endpoint where we want to send events for each action that happens on the bucket, such as object PUT, DELETE, and so on.

After we configure the S3/RGW topic, we need to 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.

The following is an example of the workflow steps:

1. Create a dedicated topic in Kafka called *BucketEvents*.
  - Kafka endpoint: `kafka.example.com:9092`
2. Configure a topic in RGW called *KafkaTopic*.
  - Configure the endpoint so 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 will trigger an event:
    - `s3:ObjectCreated:*`
    - `s3:ObjectRemoved:*`

At this point we start receiving notifications in the Kafka Topic called `BucketEvents` when a user PUTs or DELETEs an object in bucket `DemoBucket`.

### 5.3.2 Bucket notification reliability

We will discuss 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 (ACKed) only after the notification is sent to the topic's configured endpoint. This means that the round-trip time 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 will slow down your production S3 operations in the RGW as RGW will not get an acknowledgement 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 will not be 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 allow for reliable and asynchronous delivery of notifications from the RADOS Gateway (RGW) to the endpoint configured at the topic.

With persistent notifications, RGW will retry 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, powered by the S3 bucket notification feature

Let us go through 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 to have an automated data pipeline to fulfill the use case requirements.

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

All of the data modelling 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 using the service increases over time, leading to an increased number of images that need to be processed.

We need to 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 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):*

1. Ingest chest X-rays into an S3 object store 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 located at the Central Science Lab.

*At the main site (Central Science Lab)*

1. Ingested anonymized images go through a human (manual) assessment.
2. Depending on the results of the manual assessment, the images will be uploaded to the S3 object storage bucket for pneumonia or to the S3 object storage bucket for normal images.
3. Again, 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. Once the process finishes, it triggers a CI/CD workflow to deploy the new updated model to all remote sites/hospitals.

Figure 5-3 on page 84 shows the Chest X-ray architecture diagram.

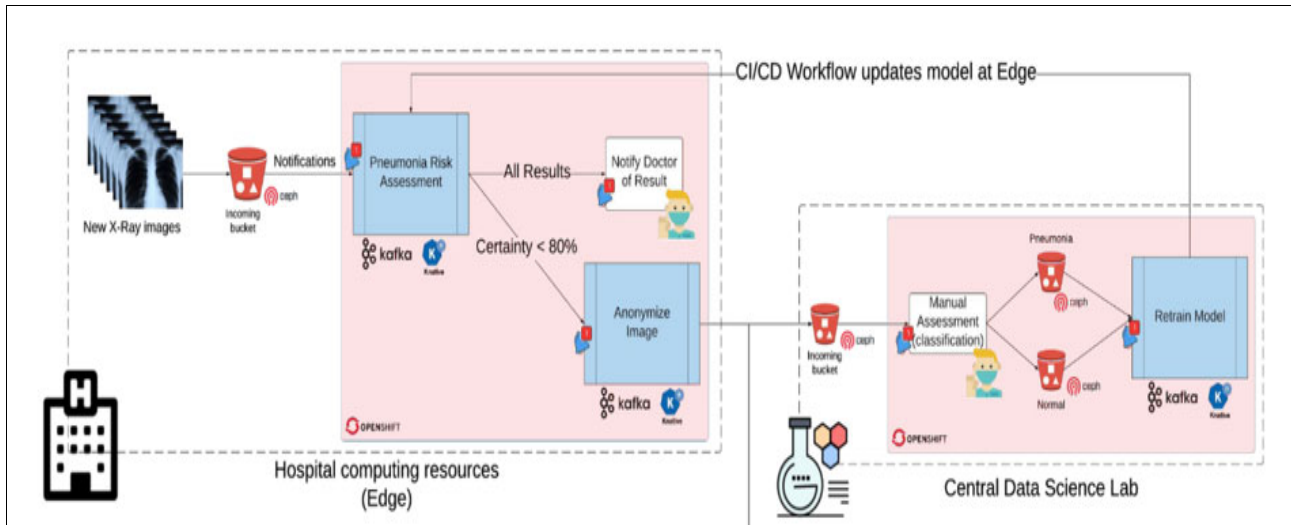


Figure 5-3 Chest X-ray architecture diagram

This pipeline is showcased by Guillaume Moutier from Red Hat in this OpenShift Commons YouTube video (slides are available [here](#)). There is also information on how to demo this event-driven pattern at [this link](#).

## 5.5 Step by step basic example of configuring event notifications for a bucket in IBM Storage Ceph

In this section we cover step by step basic example of configuring event notifications for a bucket in IBM Storage Ceph.

### 5.5.1 Prerequisites

Let us start with the prerequisites.

1. We need to have an RGW service configured and running. In our example, we have an RGW service running on node ceph-node02 on port 8080. See Example 5-1.

*Example 5-1 RGW service configured and running*

---

```
$ ceph orch ps | grep rgw
rgw.objectgw.ceph-node02.mrhvqg 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 standalone on a server: workstation.example.com, listening to requests on port 9092.

*Example 5-2 Running Kafka standalone on a 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. We need to create a Kafka topic to publish events when an S3 operation occurs on our bucket. In this example, we will create a topic called BucketEvents. See 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. Once the Kafka topic is set up, we will create a new set of S3 credentials (an access key and secret key) via the RGW. We will then use an S3 CLI client, such as the AWS CLI, to create a bucket called demobucket. See Example 5-4.

**Note:** You can use any other S3 client, We downloaded and installed the AWS CLI client following the instructions at [this link](#).

*Example 5-4 Create 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
```

---

5. At this point, we have all of our prerequisites covered and are ready to start the configuration.

## 5.5.2 Configuring RGW S3 event bucket notifications

Perform the following steps to configure RGW S3 event bucket notifications:

1. The first step is to 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` using the topic. There are several ways to create the required topic in RGW, but in this example, we will use the AWS CLI.

To configure our RGW topic, we need to create a JSON file with the topic attributes we want to set. In this example, we are using the following attributes:

- **Push-endpoint:** The Kafka endpoint where we want to send the events.
- **Verify-ssl:** As this is only an example, we have not configured SSL in Kafka
- **Kafka-ack-level:** We will wait for the Kafka endpoint ACK before considering the event sent.
- **Persistent:** Setting this option to true configures async messaging (available with IBM Storage Ceph 5.3 and higher). See Example 5-5.

*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. Once the JSON file is created, we can use the AWS CLI to create the RGW topic. The name of the RGW topic will match the name of the Kafka topic we created previously, `bucketevents`. See 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. The next step is to set up the bucket notification. We will add a new bucket notification to our bucket demobucket. This notification will send an event to the RGW topic bucketevents whenever an object is created or deleted. See 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 the above example, we set the ID/name of our topic to `kafkanotification`. We set the `TopicArn` to the ARN of the RGW topic we created in the previous step (Example 5-6). Finally, we specified the actions that will trigger events, which in this case are object creation and deletion (Example 5-7). To get a list of all supported actions/events follow [this link](#).

Once the JSON notify file has been created, we can apply its configuration to the demobucket. See Example 5-8.

*Example 5-8 Applying 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:*"
      ]
    }
  ]
}

```

```
]
}
```

---

At this point if the setup is working as expected, any new object creation or deletion will send a bucket notification to the Kafka topic.

### 5.5.3 Validating the bucket notification configuration

Now that we have finished configuring our bucket notification for bucket demobucket, let us verify that the events are reaching the Kafka topic when we create or delete an object inside the bucket.

1. First, we need to configure a Kafka consumer on the topic bucketevents we created previously. We will use the Kafka CLI tools provided with the RPM to connect a client and consume the messages from the bucketevents topic. Once you run the `kafka-console-consumer`, it will just sit there running in the foreground waiting to consume messages from the topic. See 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 we will create and delete an object in our demobucket to trigger a bucket notification event. See 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 consumed the event notification created by the creation of an object in the demobucket, under the JSON path `.Records.s3` we have information on our bucket and object. See Example 5-11 on page 88.

*Example 5-11 client has consumed the event notification 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, we would have an application consuming 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 RGW are not reaching Kafka, a good place to start is the RGW logs. You will need to enable debug mode in the RGW subsystem. This [link](#) contains instruction on how to enable debug mode for the RGW service.

## 5.6 Configuring S3 bucket notifications in OpenShift using the S3 RadosGW object storage provided by Fusion Data Foundation

We can also use S3 bucket notifications inside OpenShift with the help of Fusion Data Foundation. [IBM Storage Fusion](#) includes Fusion Data Foundation (FDF). The engine under Fusion Data Foundation is Ceph, and with Fusion Data Foundation you get Ceph deployed

inside your OpenShift cluster (There is also the possibility of connecting Fusion Data Foundation in external mode to an already deployed IBM Storage Ceph stand-alone cluster).

Deploying Ceph in OpenShift gives us the advantage of all the automation and lifecycle management provided by OpenShift Operators. In the case of Fusion Data Foundation, we have the Rook Operator taking care of deploying Ceph. If we are deploying on an OpenShift on-premises cluster, we will get the RadosGW service, which provides S3 object storage functionality, deployed for us.

One of the advantages of working with Fusion Data Foundation is the ease of use. As an example we will show how to configure S3 bucket notifications in 3 easy steps with the help of the Custom Resources provided by Rook.

## 5.6.1 Prerequisites

The following are the prerequisites.

1. Ensure that OpenShift cluster is running. See Example 5-12.

*Example 5-12 Ensure that 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 the Fusion Data Foundation service deployed. See Example 5-13.

*Example 5-13 Ensure that IBM Storage Fusion with the Fusion Data Foundation service 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                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. We need to configure a RADOS Gateway (RGW) `cephobjectstores` custom resource. This custom resource deploys the RGW service, which gives us access to our Ceph Storage S3 RGW endpoint. We also have an OpenShift service created and an external route, if S3 connectivity is required from outside the OpenShift cluster. Here is an improved version of your text. We are deploying the service without SSL for simplicity. For production deployments, HTTPS/SSL is recommended. The RGW service will get deployed with FDF out of the box for on-premises deployments. If you are running on a cloud deployment, you can follow [this link](#) for instructions on how to set up the `cephobjectstores` custom resource step by step. See Example 5-14.

*Example 5-14 configure 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. We need to create the Strimzi Kafka Operator. See Example 5-15.

*Example 5-15 Create the Strimzi Kafka Operator.*

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

5. We need to create a Strimzi Kafka topic, called bucketevents. See Example 5-16.

*Example 5-16 Create 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 standalone also apply here, with the difference that we will take advantage of the custom resources (CRs) 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 created via ObjectBucketClaims by adding labels to an ObjectBucketClaim.

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

### Topics

A CephBucketTopic represents an endpoint (of types: Kafka, AMQP0.9.1 or HTTP), or a specific resource inside this endpoint (e.g. a Kafka or an AMQP topic, or a specific URI in an HTTP server). The CephBucketTopic also holds any additional info needed for a CephObjectStore's RADOS Gateways (RGW) to connect to the endpoint. Topics don't belong to a specific bucket or notification. Notifications from multiple buckets may be sent to the same topic, and one bucket (via multiple CephBucketNotifications) may send notifications to multiple topics.

1. We will start with the configuration of the Topic, using the CephBucketTopic custom resource provided by rook, we will name our topic `bucketevents`, most of the parameters have already been explained in 5.5, “Step by step basic example of configuring event notifications for a bucket in IBM Storage Ceph” on page 85. You can follow [this link](#) if you would like to see a detailed explanation of any of the parameters we are using in this example.

As the URI for Kafka, we are using the service FQDN as we will be accessing 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. Second step is to create the bucket notification using the custom resource `CephBucketNotification` provided by Rook. In the specs we specify the name of the RGW topic, where the events will be sent once they are triggered, and also the events that we want to send out to the topic. In this example, only the S3 PUT and COPY actions will trigger an event notification into Kafka. See Example 5-18.

*Example 5-18 Create 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

---

- Next using the RGW storage class, we create an Object Bucket Claim (OBC). The OBC will take care of creating a bucket and user credentials in RGW for us. By using bucket notification labels, OBC will also configure the bucket notification, In our case, we will use the bucket notification we created in the previous step, called bucket-notification. See Example 5-19.

*Example 5-19 create 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

Perform the following steps to validate the bucket notification configuration.

- First, we will verify that the OBC notify labels worked as expected and that the notification is configured in our new bucket. We will 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 Extract the credentials*

---

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

---

- We 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 Add 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 OCP so we will use the OCP service FQDN as the RGW S3 endpoint. We can get the OCP service FQDN from the OBC config map with the same name. See Example 5-22.

*Example 5-22 get the OCP service FQDN from the OBC config map 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, we can run the `get-bucket-notification` `s3api` command to check if the configuration is in place for our bucket `databucket2`. See Example 5-23.

*Example 5-23 Check if the configuration is in place for our 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, let us upload something to the bucket and check if it triggers the event and sends it to our Kafka topic `bucketevents`. See Example 5-24.

*Example 5-24 Upload something to the bucket and check if 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 our configuration is functional and that the events are reaching their destination topic, `bucketevents`, we will use the Kafka consumer available in the Kafka pod. See Example 5-25.

*Example 5-25 Double-check 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 have concluded that our configuration is working and we are able to configure S3 bucket notifications in our OpenShift application buckets.





# IBM Storage Fusion disaster recovery

This chapter will cover the IBM Storage Fusion disaster recovery configurations. This chapter has the following sections:

- ▶ “Introduction to Fusion Data Foundation” on page 98
- ▶ “IBM Storage Ceph stretch mode” on page 103
- ▶ “Fusion MetroDR for RPO Zero” on page 114

## 6.1 Introduction to 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, packaged as an operator to facilitate simple deployment and management.

Fusion Data Foundation 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 workloads, for example, Red Hat OpenShift Container Platform logging and monitoring, and PostgreSQL.
- Shared and distributed file system, catering primarily to software development, messaging, and data aggregation workloads, for example, Jenkins build sources and artifacts, Wordpress uploaded content, Red Hat OpenShift Container Platform registry, and messaging 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, primarily targeting data intensive applications, for example, 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.

Fusion Data Foundation version integrates a collection of software projects, including:

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

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

### 6.1.1 Introduction to Fusion Disaster Recovery (DR)

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 Fusion Data Foundation (FDF) DR capability allows DR across multiple Red Hat OpenShift Container Platform clusters (OCP). It is categorized into three types:

#### – **Metro-DR**

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

#### – **Disaster Recovery with stretch cluster**

The Stretch cluster solution ensures business continuity with no-data loss disaster recovery protection with Fusion Data Foundation (FDF) synchronous replication in a



single OpenShift cluster, stretched across two data centers with low latency and one arbiter node.

– **Regional-DR (Tech Preview)**

Regional-DR ensures business continuity during the unavailability of a geographical region, accepting some loss of data in a predictable amount. In the public cloud, this would be 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 usually expressed 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 need to 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 our system take to recover after we are notified of a business disruption?"

Disaster recovery strategies can be grouped into two main categories:

– **Active/passive**

In an active/passive disaster recovery strategy, all traffic is routed to one datacenter while the other is in standby mode. In a disaster, traffic may be down for a short period of time as operations are switched to the standby datacenter. This strategy is best suited for situations where only two datacenters are available.

– **Active/active**

In an active-active disaster recovery 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
<b>Regional-DR</b>	<ul style="list-style-type: none"> <li>▶ Disaster recovery solution across two OpenShift clusters in two different geographical locations connected by WAN network.</li> <li>▶ Based on asynchronous volume level replication for block and file volumes.</li> </ul>	<ul style="list-style-type: none"> <li>▶ Sustaining the loss of a data center or a geographical region.</li> </ul>

Offering	Definition	Use Case
<b>Metro-DR</b>	<ul style="list-style-type: none"> <li>▶ No-data loss disaster recovery protection with FDF/ODF-based synchronous replication between two OpenShift clusters in two data centers with low latency.</li> <li>▶ Based on IBM Storage Ceph stretched cluster.</li> </ul>	<ul style="list-style-type: none"> <li>▶ Sustaining the loss of cluster in a campus or Metro DR scenario.</li> <li>▶ Follows the best practice DR principles with fault-isolated local clusters.</li> </ul>
<b>DR with stretch cluster</b>	<ul style="list-style-type: none"> <li>▶ No-data loss disaster recovery protection with FDF/ODF-based Synchronous replication in a single OpenShift cluster, stretched across two data centers with low latency and arbiter node.</li> </ul>	<ul style="list-style-type: none"> <li>▶ Sustaining component failures in a campus or Metro deployment.</li> <li>▶ Benefits from quick recovery using HA and DR principles.</li> </ul>

### 6.1.2 RegionalDR overview

This 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.

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

The following lists the details of the Regional-DR offering:

- Two OCP/FDF internal clusters.
- FDF/ODF external not supported with Regional-DR.
- Red Hat Advanced Cluster Manager is required (Version 2.7 or higher).
- Supported Platforms: Public Cloud and on-premises, 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 on page 101 shows the IBM Fusion Data Foundation RegionalDR offering.

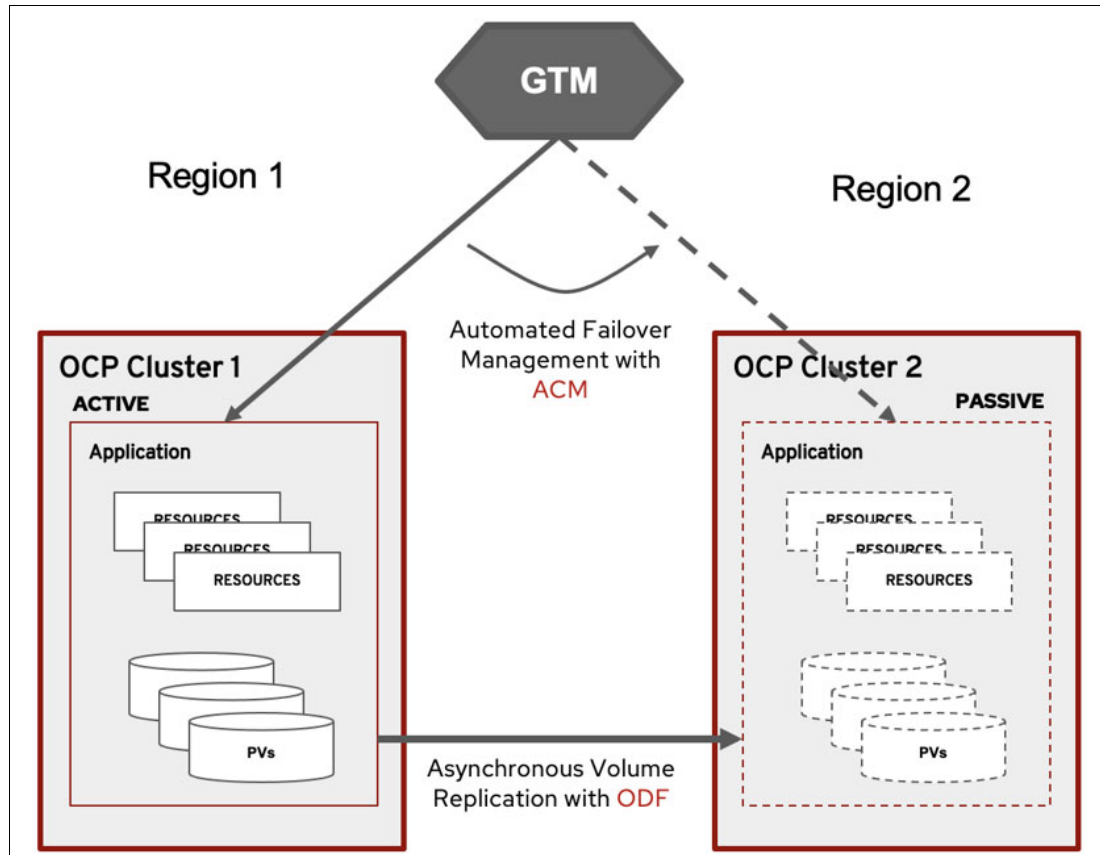


Figure 6-1 IBM Fusion Data Foundation RegionalDR

### 6.1.3 MetroDR overview

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

The following lists the details of the Metro-DR offering:

- Two OCP/FDF clusters are deployed in separate data centers connecting to an external IBM Storage Ceph cluster.
- Single IBM Storage Ceph stretch cluster across three zones with the third zone only having one MON running as tie-breaker, Single IBM Storage Ceph stretch cluster serving both the FDF clusters.
- Red Hat Advanced Cluster Manager is required (Version 2.7 or higher).
- Supported platforms: VMware and baremetal only.
- Data types: Block and file.
- Latency: <10 ms RTT for data nodes, <100 ms RTT for arbiter/MON node.
- Network: High bandwidth (Campus Network).
- Arbiter: Third zone or site required for the arbiter, <100 ms RTT latency required.
- Number of zones or sites: 3, third zone running with arbiter/MON only, could be a VM.
- RPO: 0.
- RTO: Between 5 and 15 minutes (Depends on the application start-up time).

Figure 6-2 on page 102 shows the IBM Fusion Data Foundation MetroDR offering.

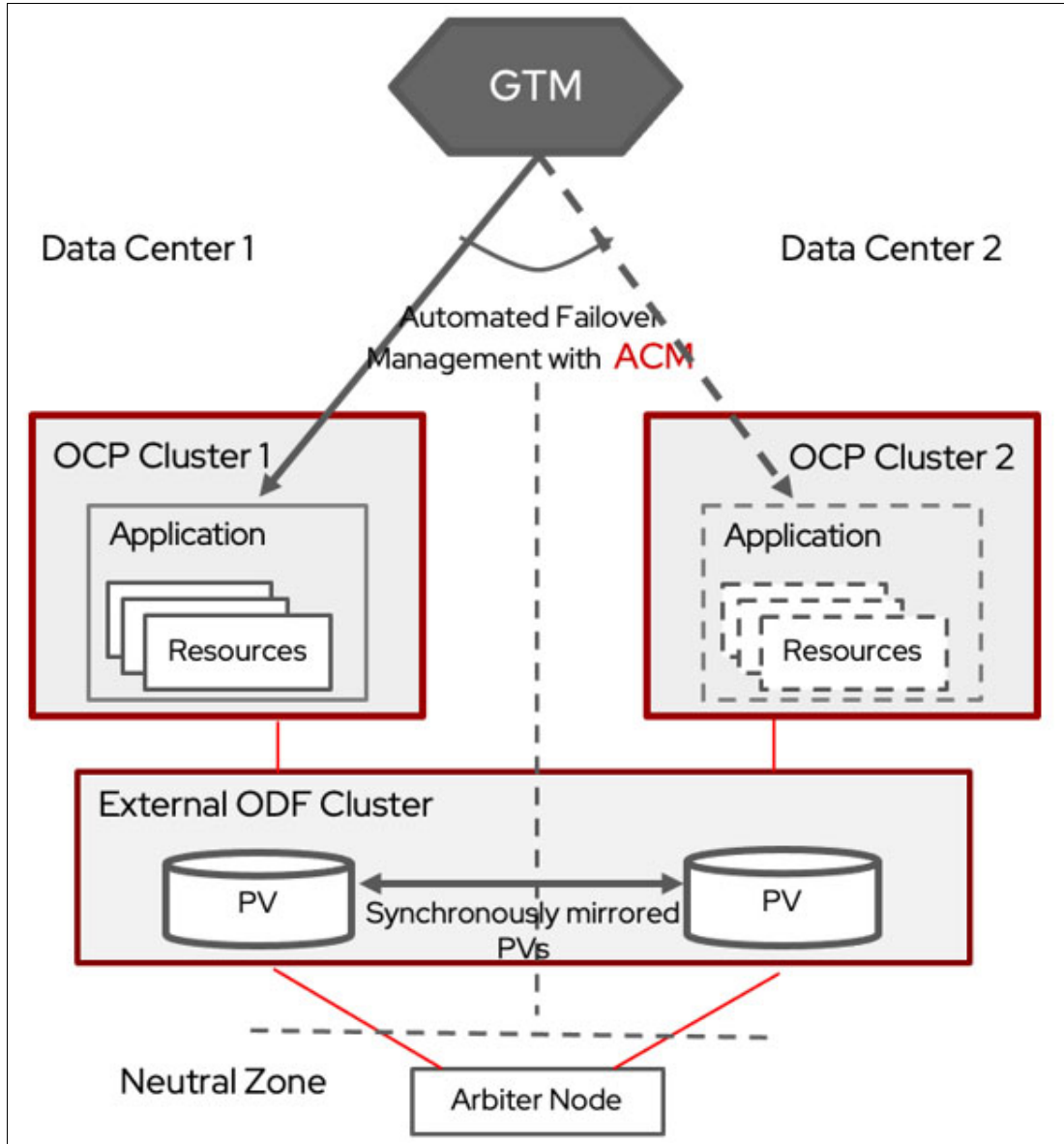


Figure 6-2 IBM Fusion Data Foundation MetroDR

### 6.1.4 OpenShift DR with stretched cluster

Stretch cluster solution ensures business continuity with no-data loss disaster recovery protection with Fusion Data Foundation (FDF) synchronous replication in a single OpenShift cluster, stretched across two data centers with low latency and one arbiter node.

The following lists the details of the Stretched Cluster OpenShift DR offering details:

- One OpenShift and one FDF internal cluster stretched across 2/3 data centers.
- FDF is deployed internally. No external IBM Storage Ceph is required.
- No Advanced Cluster Manager is required.
- Supported platforms: VMware and baremetal only.
- Data types: Block, file and object.
- Latency: <10 ms RTT for all nodes.
- Network: High bandwidth (Campus Network).

- Third zone for Ceph and OpenShift arbiter required.
- Three zones with 3rd zone having control plane and MON running on an arbiter node.
- RPO: 0.
- RTO: < 1 min (depending on the application).

Figure 6-3 shows the IBM Fusion Data Foundation stretched mode offering.

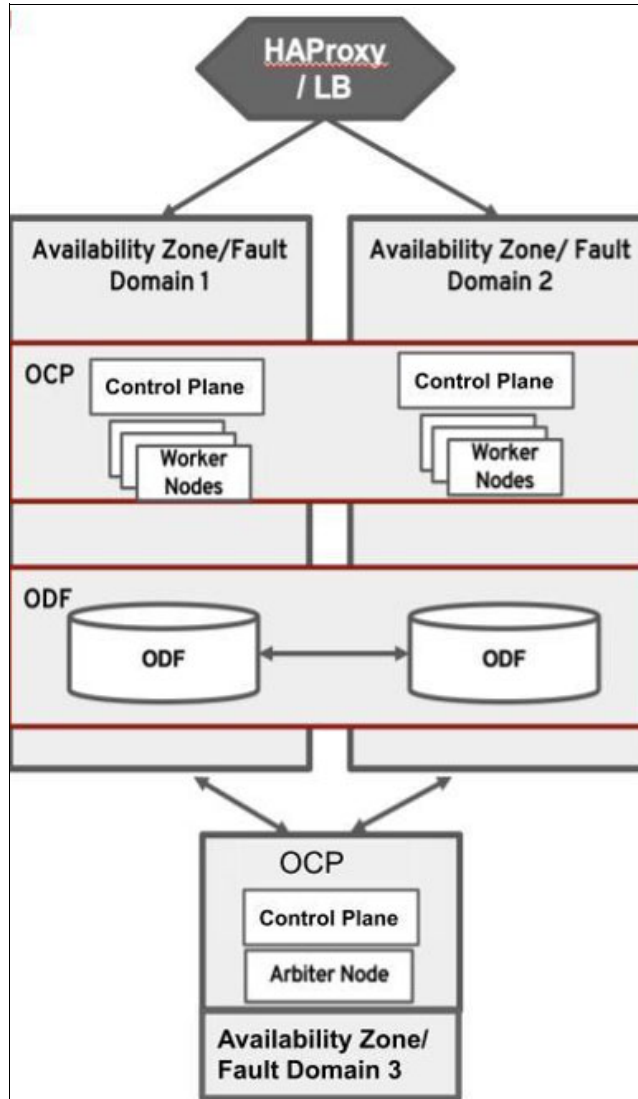


Figure 6-3 IBM Fusion Data Foundation stretched mode

## 6.2 IBM Storage Ceph stretch mode

A stretch cluster is a group of servers in separate data centers 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, under 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 OSDs and monitors. However, in some cases, such as a "stretched-cluster" deployment where a significant portion of the cluster relies on a single network component, stretch mode may 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 may have high latency compared to the main sites.

Ceph's 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 will recover. If a data center is lost but a quorum of monitors is still present, Ceph will maintain availability, provided that the cluster has enough data copies to meet the `min_size` configuration option or the CRUSH rules will cause the cluster to re-replicate the data until the `min_size` configuration option is met.

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

MetroDR is a synchronous replication solution. Write IO at the primary site is replicated to the secondary site before returning the write acknowledgment. This 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 4 copies of the data degrades the IOPS performance that you can achieve with MetroDR, especially noticeable with random writes of small blocks.

A deep dive into MetroDR concepts is available in this OpenShift Data Foundation video series:

[MetroDR part 1. RHCS Stretched + ODF External](#)

[MetroDR part 2. RHCS demo + Connect ODF External](#)

[MetroDR part 3. ACM + Failover/Failback App Demo](#)

[MetroDR Stateful Application Disaster Recovery with Zero Data Loss](#)

## 6.2.1 Stretched cluster pitfalls we avoid with Ceph stretched mode

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

While Ceph prioritizes data integrity and consistency, stretched clusters may compromise data availability.

Examples of that can be:

- ▶ **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 down and remove them from the acting placement groups (PGs). This failure to mark ODSs down will occur

even though the primary PG cannot replicate data (a situation that, under normal non-netsplit circumstances, would result in the marking of affected OSDs as down and their removal from the PG). If this happens, Ceph cannot satisfy its durability guarantees, and consequently, IO will be blocked.

- **Constraints do not guarantee the replication of data across data centers:** However, it may 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 may go active with two replicas in A and zero replicas in B. This means that if A is lost, the data will be lost and Ceph will not be able to operate on it. This situation is surprisingly difficult to avoid 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 will be explained in detail, going through the most important aspects to consider around networking, server hardware and Ceph component placement.

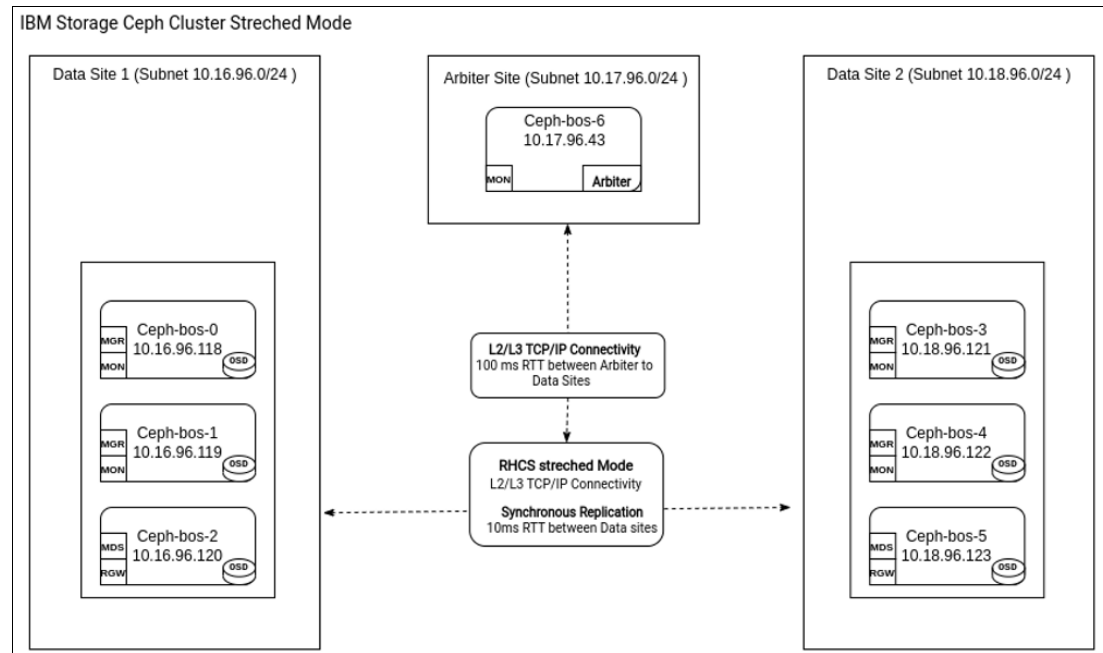


Figure 6-4 IBM Storage Ceph stretched mode

### 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, allowing for different subnets/CIDRs on each site and enabling communication between Ceph servers and components via L3 routing.

IBM Storage Ceph can be configured with two different networks the public and cluster network. The Ceph public network must be accessible and connected between all three sites (data + arbitrator Site), as all IBM Storage Ceph services such as MONS, OSDs, RGW require communication over the public network. The private or cluster network is used by the OSDS

services, so it only needs to be configured on the two data sites where the OSDs reside. There is no need to configure the cluster network on the arbiter site.

Network reliability is of huge importance. Flapping networks between the three sites will introduce data availability and performance issues into the cluster. It is not just about the network being accessible but also the consistency of its latency. Frequent spikes in network latency can lead to false alarms and other problems.

Regarding latency, a minimum of 10ms 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 100ms RTT, so it may be beneficial to deploy the arbiter node as a VM in a cloud provider, if allowed by security.

Suppose the arbiter node is configured on a cloud provider or a remote network that goes through a WAN. In that case, the recommendation is to set up a VPN between the data sites and the arbiter site and enable encryption in transit. The messenger v2 encryption feature will encrypt the communications between the different Ceph services, so all the communications between the MON in the arbiter site and the other components on the data sites will be 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 configured in the replica count before the client receives an acknowledgment (ACK). This means that the client will not receive the ACK for a write until two copies have been written to each site, adding the round-trip time (RTT) between sites to the latency of every write workload.

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

By default, Ceph always reads data from the primary OSD, regardless of its location. This 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 Fusion Data Foundation (FDF) in version 4.13/4.14, and work is underway to make it available for MetroDR deployments. *This is a vital feature for stretched site setups, as it drastically reduces the number of times we need to contact OSDs on the other site for reads, reducing 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 will cover in the following paragraph. Here is a [link](#) to the official Hardware section in the IBM Storage Ceph documentation.

Important differences:

- ▶ IBM Storage Ceph in stretched mode only supports full flash configurations. HDD spindles are not supported. The reasoning behind this is that if we lose a full datacenter for what could be a long period, we still have two replicas/copies of the data available on the remaining site; replica 2 configurations are only supported with flash media.



- ▶ IBM Storage Ceph only supports replica 4 as the replication schema. Any other replication or erasure coding scheme is not supported. So, we need to calculate our total usable storage accordingly.
- ▶ IBM Storage Ceph cluster can only be used for the MetroDR use case; it can't 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 only be deployed in baremetal or VMware platforms.
- ▶ VMware deployments of IBM Storage Ceph are supported for the MetroDR use case. The following are some best practices for VMware deployments:
  - You need to have reserved resources CPU and memory for the Ceph Virtual Machines.
  - On the VMs set the latency sensitive option to high.
  - In general, applying all VMware recommendations for low-latency use cases would make sense.

VMware Storage provided to IBM Storage Ceph for the MetroDR use case supports using .VMDK files for the VMs. However, it is important to note that Ceph will only perform as well as the data store providing the .VMDKs. If we use a backing store using an underperforming VSAN, we can expect Ceph performance to be abysmal. Conversely, if we use a dedicated all-flash SAN datastore for a specific set of VMs, we can expect Ceph performance to be much better.

- ▶ Avoid using VSAN datastores for Ceph node drives, as the performance will not be satisfactory for any IO-intensive applications.

### Component placement

Ceph services (MONs, OSDs, RGWs and so forth) 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.

- ▶ **MONs:** A minimum of 5 MONs are required, 2 MONs per data site, and 1 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:** We can configure two or four managers, with a minimum of one manager per data site. Four managers is recommended to provide high availability with an active/passive pair on the remaining site in the event of a data site failure.
- ▶ **OSDs:** Distribute equally across all the nodes in both of the data sites. Custom CRUSH rules providing two copies in each site (using 4 copies) must be created when configuring the stretch mode in the Ceph cluster.
- ▶ **RGWs:** The minimum recommended is four RGW services, two per data site, to ensure that we can still provide high availability for object storage on the remaining site in the event of a site failure.
- ▶ **MDS:** For CephFS Metadata services, the minimum recommended is four MDS services, two per data site. In the case of a site failure, we will still have two MDS services on the remaining site, one active and the other one on standby.

### 6.2.3 Deployment example

In this section we will discuss some deployment examples.

## Deploying the Ceph cluster

The deployment of Ceph is well documented in Chapter 1, “IBM Storage Ceph Dashboard” on page 1, and also in the official documentation following this [link](#).

During the cluster bootstrap using the cephadm deployment tool, we can inject a service definition yaml file that will do 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 configured in stretch mode. Remember, this is just an example that must be tailored to your deployment.

*Example 6-1 Example of deploying an IBM Storage Ceph Cluster 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 ## IPs, hostnames, etc will need to be replaced
hostname: ceph1 ## depending on each individual use case
location:      ## this is just an example template
  root: default
  datacenter: DC1 ## DC1 is the label we set in the crushmap
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:
  datacenter: DC1
labels:
  - osd
  - mon
  - mgr
---
service_type: host
addr: 10.0.40.4
hostname: ceph3
location:
  datacenter: DC1
labels:
  - osd
  - mds
  - rgw
---
service_type: host
addr: 10.0.40.5
hostname: ceph4
location:
  datacenter: 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
  datacenter: DC2 ## Datacenter 2 label for the crushmap
labels:
  - osd
  - mon
  - mgr
---
service_type: host
addr: 10.0.48.3
hostname: ceph5
location:
  datacenter: DC2
labels:
  - osd
  - mon
  - mgr
---
service_type: host
addr: 10.0.48.4
hostname: ceph6
location:
  datacenter: DC2
labels:
  - osd
  - mds
  - rgw
---
service_type: host
addr: 10.0.48.5
hostname: ceph7
location:
  datacenter: DC2
labels:
  - osd
  - mds
  - rgw
---
service_type: host ## This is the arbiter node, it has no osds
addr: 10.0.49.2 ## so also no Datacenter crushmap 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

```

---

Once the yaml is modified, we can make use of the `--apply-spec` parameter available in the `cephadm bootstrap` command, so the provided yaml file configuration is fed into `cephadm` to get the cluster into the state that we have described in the yaml file, as shown in Example 6-2.

*Example 6-2 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 the official IBM Storage Ceph documentation at this [link](#).

IBM Storage Ceph in stretched mode only supports the replica schema of four, so we will have two copies of the data available on each site, meaning that even if we lose a full site, we will have two copies available of our data.

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

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

*How does Ceph write 2 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, we 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 ceph osd tree command*

---

```
# ceph osd tree
ID CLASS WEIGHT TYPE NAME STATUS REWEIGHT PRI-AFF
-1 0.58557 root default
-3 0.29279 datacenter 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 datacenter 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 be able to achieve our goal of having two copies per-site, we need to define our failure domain at the pool level, so as an example if we take pool `rbdpool`, we can see it is using a “`crush_rule`” with an id of “1”. See Example 6-4.

*Example 6-4 Define 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 we check the failure domain that is configured on `crush_rule 1`, we can see that the rule is going to select two hosts from each datacenter to store the four copies of each client write to the pool. See Example 6-5.

*Example 6-5 Check 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": "datacenter"  
  },  
  {  
    "op": "chooseleaf_firstn",  
    "num": 2,  
    "type": "host"  
  },  
  {  
    "op": "emit"  
  }  
]  
}
```

---

In Figure 6-5 on page 113 you can see the correlation between the OSD crush map, and the crush rule we have defined in Example 6-4 on page 111.

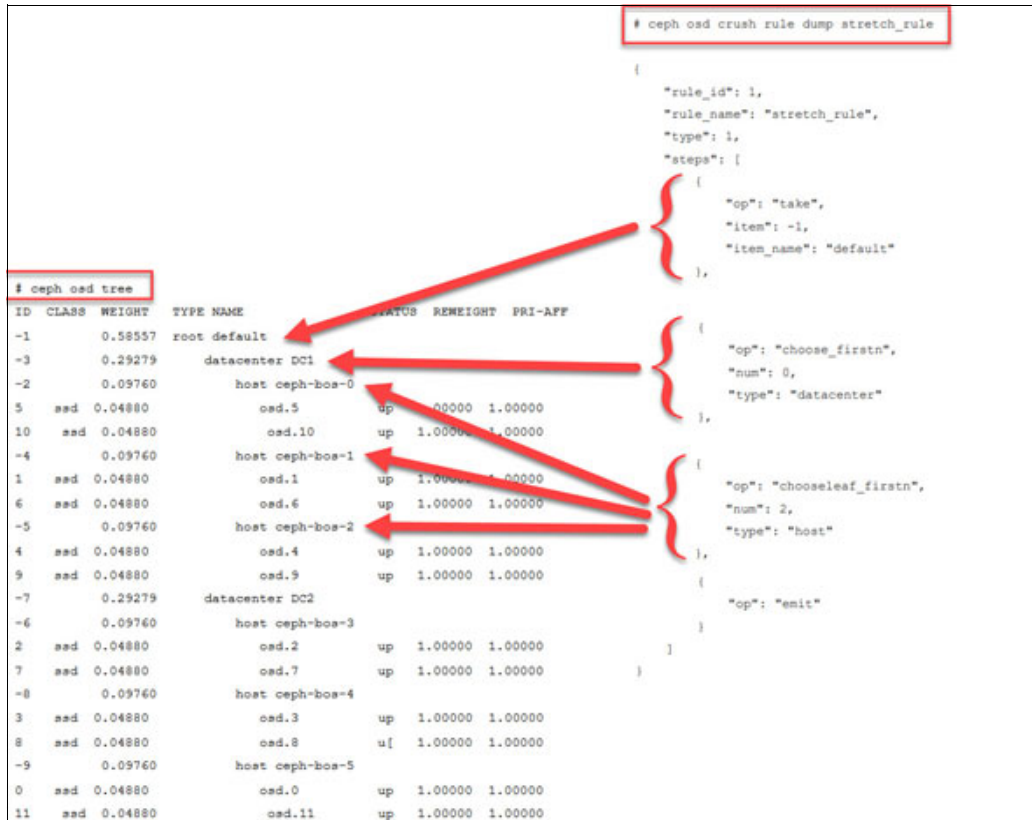


Figure 6-5 IBM Storage Ceph Crush map and Crush rule

When the stretch mode is enabled, the OSDs will only take PGs active when they peer across datacenters, assuming both are alive with the following constraints:

- ▶ Pools will increase in size from the default 3 to 4, expecting two copies on each site.
- ▶ OSDs will only be allowed to connect to monitors in the same datacenter.
- ▶ New monitors will not join the cluster if they do not specify a location.

If all the OSDs and monitors from a datacenter become inaccessible at once, the surviving datacenter will enter a degraded stretch mode, which implies:

- ▶ This will issue a warning, reduce the pool's `min_size` to 1, and allow the cluster to go active with data in the remaining site.
- ▶ The pool size parameter is not changed, so you will also get warnings that the pools are too small.
- ▶ Although, the stretch mode flag will prevent the OSDs from creating extra copies in the remaining datacenter (so it will only keep two copies, as before).

When the missing datacenter comes back, the cluster will enter recovery stretch mode triggering the following actions:

- ▶ This changes the warning and allows peering but still only requires OSDs from the datacenter, which was up the whole time.
- ▶ When all PGs are in a known state and are neither degraded nor incomplete, the cluster transitions back to the regular stretch mode where:
- ▶ The cluster ends the warning.
- ▶ Restores `min_size` to its starting value (2) and requires both sites to peer.

- ▶ Stops requiring the always-alive site when peering (so you can failover to the other site, if necessary).

## 6.3 Fusion MetroDR for RPO Zero

In this section we discuss Fusion MetroDR for RPO Zero.

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

#### *Internal deployment*

Deployment of IBM Storage Fusion Data Foundation (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 graphical user interface (GUI) to deploy Fusion Data Foundation (FDF) in internal mode using the local storage operator and local storage devices.

Ease of deployment and management are the highlights of running Fusion Data Foundation (FDF) services internally on OpenShift Container Platform.

#### *External deployment*

IBM Storage Fusion Data Foundation (FDF) exposes the IBM Storage Ceph services running outside of the OpenShift Container Platform cluster as storage classes.

The external approach is best used when:

- ▶ Storage requirements are significant (600+ storage devices).
- ▶ Multiple OpenShift Container Platform clusters must consume storage services from a common external cluster.
- ▶ Another team, Site Reliability Engineering (SRE), storage, and so on, needs to manage the external cluster providing storage services. This team might already exist.
- ▶ Storage team with already deployed Ceph Standalone clusters that like to customize and tune their IBM Storage Ceph clusters.
- ▶ Disaster recovery with RPO 0, using the MetroDR solution.

### 6.3.2 MetroDR solution components

The following lists the MetroDR solution components.

#### ***Red Hat Advanced Cluster Management for Kubernetes***

*Red Hat Advanced Cluster Management (RHACM)* can manage multiple clusters and application lifecycles. Hence, it serves as a control plane in a multi-cluster environment.

#### ***IBM Storage Ceph***

IBM Storage Ceph is a massively scalable, open, 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 significantly 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***

Fusion Data Foundation (FDF) provides the ability to provision and manage storage for stateful applications in an OpenShift Container Platform cluster. Ceph backs it as the storage provider, whose lifecycle is managed by Rook in the Fusion Data Foundation component stack. Ceph-CSI provides the provisioning and management of Persistent Volumes for stateful applications.

### ***OpenShift DR***

OpenShift DR is a disaster recovery orchestrator for stateful applications across a set of peer OpenShift clusters, which are deployed and managed by using RHACM and provide cloud-native interfaces to orchestrate the life cycle of an application's state on persistent volumes. These include:

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

OpenShift DR is split into three components:

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

## **6.3.3 MetroDR architecture and layout**

Figure 6-6 on page 116 illustrates a MetroDR architecture, featuring three OpenShift clusters, one on each site. The data sites host OpenShift clusters that execute applications, while the arbiter site houses the ACM Hub cluster, responsible for initiating application failover and failback.

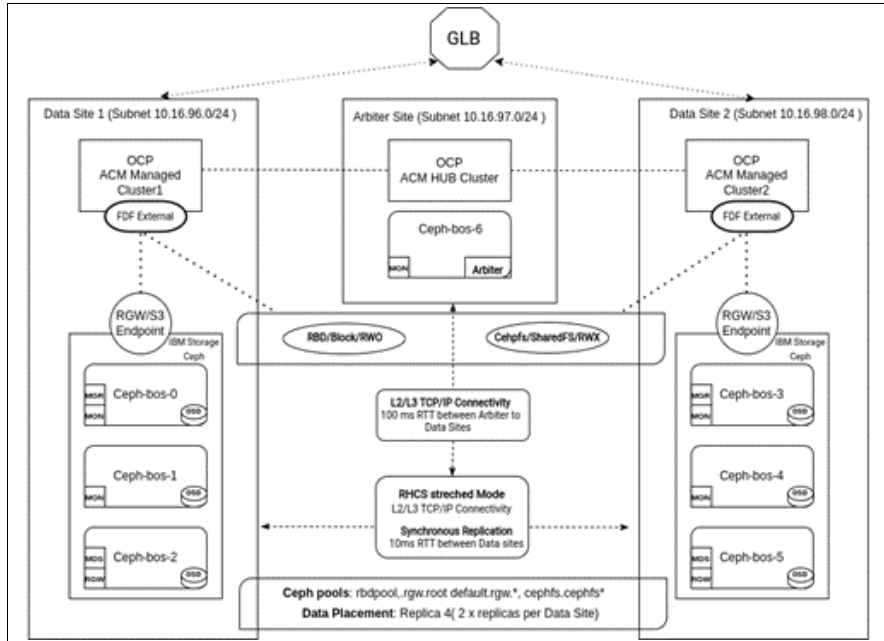


Figure 6-6 IBM Fusion Data Foundation MetroDR architecture

Both OpenShift clusters in the data sites will be managed ACM clusters. Each OCP cluster will deploy its instance of Fusion Data Foundation; they will use the external mode deployment so each FDF in external mode will connect to the same IBM Storage Ceph cluster in stretched mode.

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

Concurrent access to the same volume by both OpenShift clusters using a single stretched IBM Storage Ceph cluster can corrupt the filesystem. We must avoid this at all costs. To avoid this issue, MetroDR uses a Ceph fencing mechanism called *blocklisting*. We 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 the previous statement in mind, we have to be aware that with MetroDR once we start the failover, it is going to be a failover of all applications that MetroDR protects; we do not have granular, per-application failover in MetroDR because we need to fence off all worker nodes that are part of the OpenShift cluster to ensure that no filesystem/PV is still being accessed on the primary/failed site.

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

This also brings us to the topic of capacity planning. We will always need to have enough free resources CPU/MEM on the OCP clusters to match all the applications protected with MetroDR.

The global load balancer (GLB) depicted in Figure 6-6 on page 116 is not included in the MetroDR solution. Instead, site failure detection and redirection are handled by the site-level components. In the event of a site failure, client requests are automatically redirected to the remaining site. While application-level health checks can be implemented, they are not as crucial in MetroDR as compared to RegionalDR, as MetroDR involves failing over the entire site, resulting in all application requests being redirected to the remaining site.

### 6.3.4 MetroDR application Failover and Failback workflow

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

- ▶ ACM Subscription-based application:
  - Follow this [link](#) for an example of deploying an application using ACM subscriptions.
- ▶ ACM ArgoCD ApplicationSet-based application:
  - Follow this [link](#) for an example of deploying an application using ACM ArgoCD ApplicationSets.

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

1. Our monitoring system has detected a problem with Site 1: health checks are failing, and the application is inaccessible.
2. After the operator verifies that there is an issue with Site 1, the failover workflow is started.
3. First, we must fence all worker nodes in Site 1.
4. Once the worker nodes are fenced, we can trigger the failover from the ACM HUB UI.
5. The application PV metadata is restored in Site 2, making the PV available for the application to start up.
6. ACM HUB deploys the application in Site 2, using the PV to restore all previous data.
7. The application successfully starts up on Site 2.
8. The global load balancer detects that the application is healthy on Site 2 and redirects client traffic to Site 2..
9. At this point, our service is recovered, and the app is running again with zero data loss.
10. At some point, Site 1 is recovered, and all the FDF/ODF worker nodes are running again after a restart.
11. We removed the fencing that was in place for the worker nodes in Site 1.
12. We reallocated 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 available on the following [link](#).

There is also a YouTube video by Annett Clewett that shows the Failover and Failback of an application using a queue from the IBM MQ series product. The video is available at the following [link](#).

### 6.3.5 MetroDR deployment example

We will not be covering all the deployment steps, as they are very well laid out in the official documentation they can be accessed by clicking on the following. [link](#).





## S3 enterprise user authentication (IDP) and authorization (RBAC)

In this chapter, we discuss IBM Storage Ceph Object Storage authentication methods and dive deep into using the S3 API Secure Token Service (STS) and IAM (Identity and Access Management) roles to provide S3 users with the means to authenticate against an identity provider and gain access to specific S3 datasets based on their role.

This chapter has the following sections:

- ▶ “Introduction to RadosGW Auth methods” on page 120
- ▶ “Step-by-step deployment guide: S3 enterprise user authentication (IDP) and authorization (RBAC)” on page 124
- ▶ “IBM Storage Ceph STS configuration” on page 143
- ▶ “Testing Assume Role With Web Identity for role based access control” on page 158

## 7.1 Introduction to RadosGW Auth methods

To access Object Storage using the s3 protocol, you must provide an S3 access key and a secret key. The actual content of the access key can be varied by the implementation of the S3 API but it serves as your identity. When you send a request to the S3 API, the API will check it knows about your s3 access key and look 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 will be validated and, upon successful validation, will be executed. S3 treats a successful validation as proof that you hold the secret key and are the identity you have sent.

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

- ▶ RGW local database authentication.
- ▶ LDAP authentication
- ▶ Keystone authentication
- ▶ Secure Token Service (STS):
  - with LDAP (STS-lite)
  - with keystone (STS-lite)
  - with OpenID connect authentication

### 7.1.1 Introduction to RadosGW Secure Token Service

Amazon Secure Token Service (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 Secure Token Service (STS) APIs. Users first authenticate against STS and, upon success, receive a short-lived s3 access and secret key that can be used in subsequent requests.

IBM Storage Ceph Object STS (Security Token Service) gives us the availability to Authenticate Object Storage users against your enterprise Identity provider (LDAP, AD, and so forth) 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 OpenID connect. See Figure 7-1 on page 121.

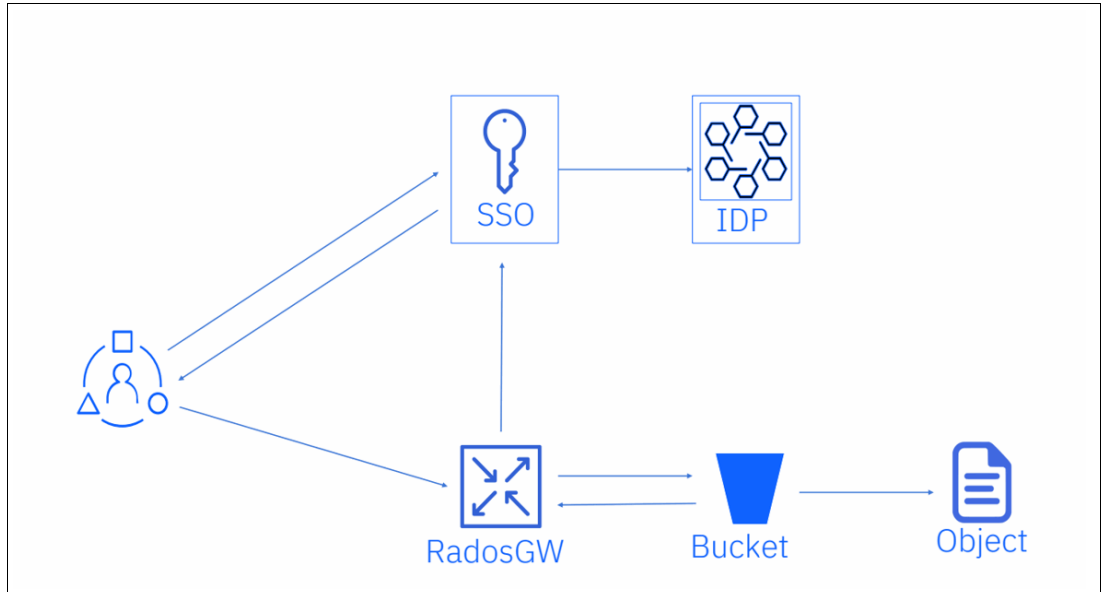


Figure 7-1 Security Token Service configuration methods

## 7.1.2 Introduction to STS with OIDC (SSO) provider

With OAUTH authentication, we use an OIDC service to authenticate. We need to have an OpenID Connect/ OAuth2 compatible service. Red Hat has tested the integration with RGW of 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 first authenticates against an OIDC to get a JWT (token).
2. The user would then call the `AssumeRoleWithWebIdentity` API against the STS API, passing in the role they want to access and the path to the JWT token created in Step 1.
3. RGW will check with the OIDC provider about the validity of the token.
4. STS will create and provide temporary credentials for the user
5. The user can access the S3 resources with a specific role.

See Figure 7-2 on page 122.

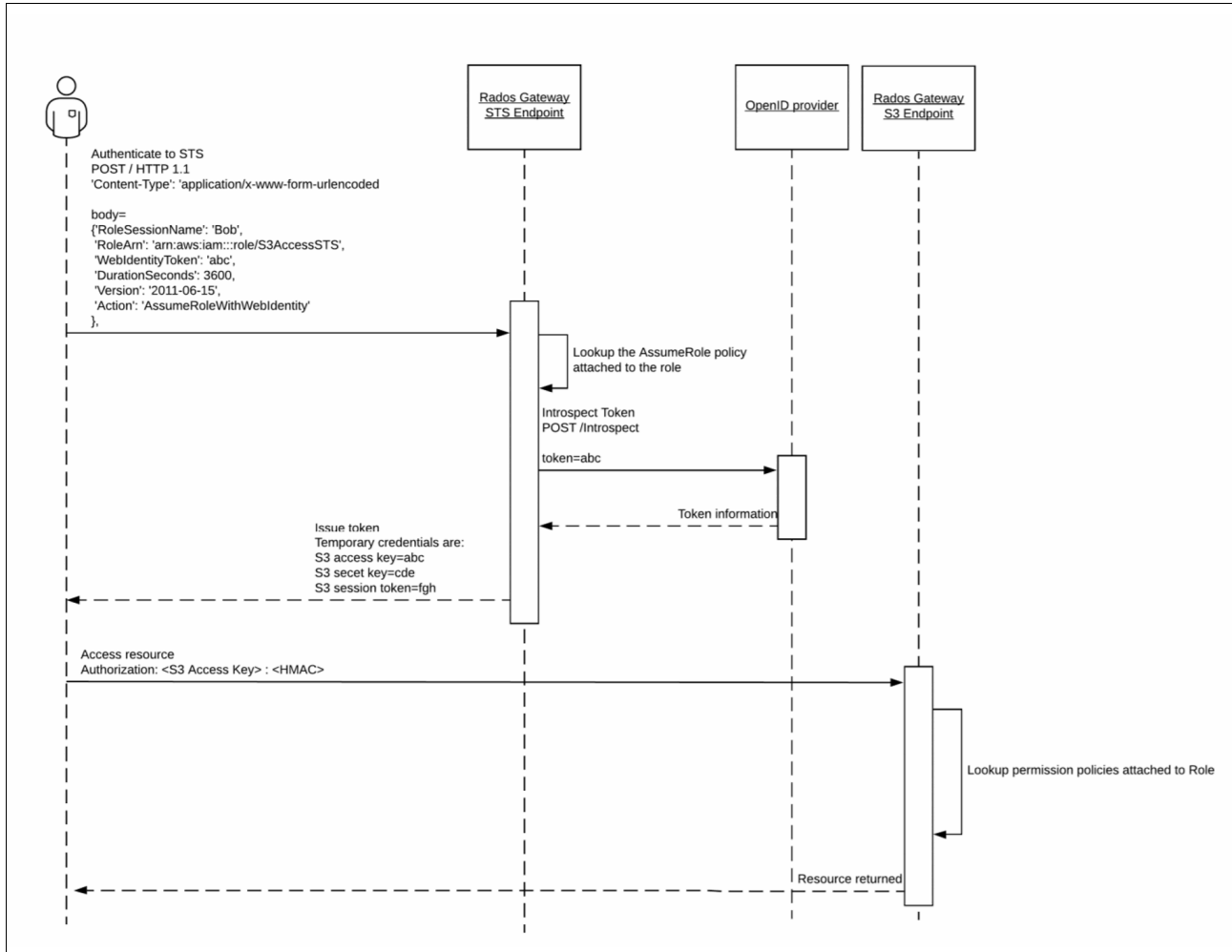


Figure 7-2 High-level authentication workflow

### 7.1.3 Introduction to IAM role policies

First, we need to introduce the concept of an Amazon Resource Name, permission policies, bucket policy and roles.

An *Amazon Resource Name (ARN)* is used to identify a user, group or a role amongst others. The ARN of a username has the general format of `arn:aws:iam:::user1`. ARNs are important as they are the 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 via 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 is generally used to restrict who can read/write objects. The bucket policy is administered via the S3 API and can be self-managed by end users if they have appropriate permissions defined.

A *role* is similar to a user and has its own ARN in the general 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 are then able to assume the role



based on another permission policy which we refer to as the `assume-role-policy-doc`. The `assume-role-policy-doc` grants access to the roles if the user is able to meet its conditions.

IAM role policies, during STS authentication, users can request to assume a role and inherit all the S3 permissions configured for that role by an RGW administrator, allowing to configure RBAC or ABAC authorization policies. See Figure 7-3.

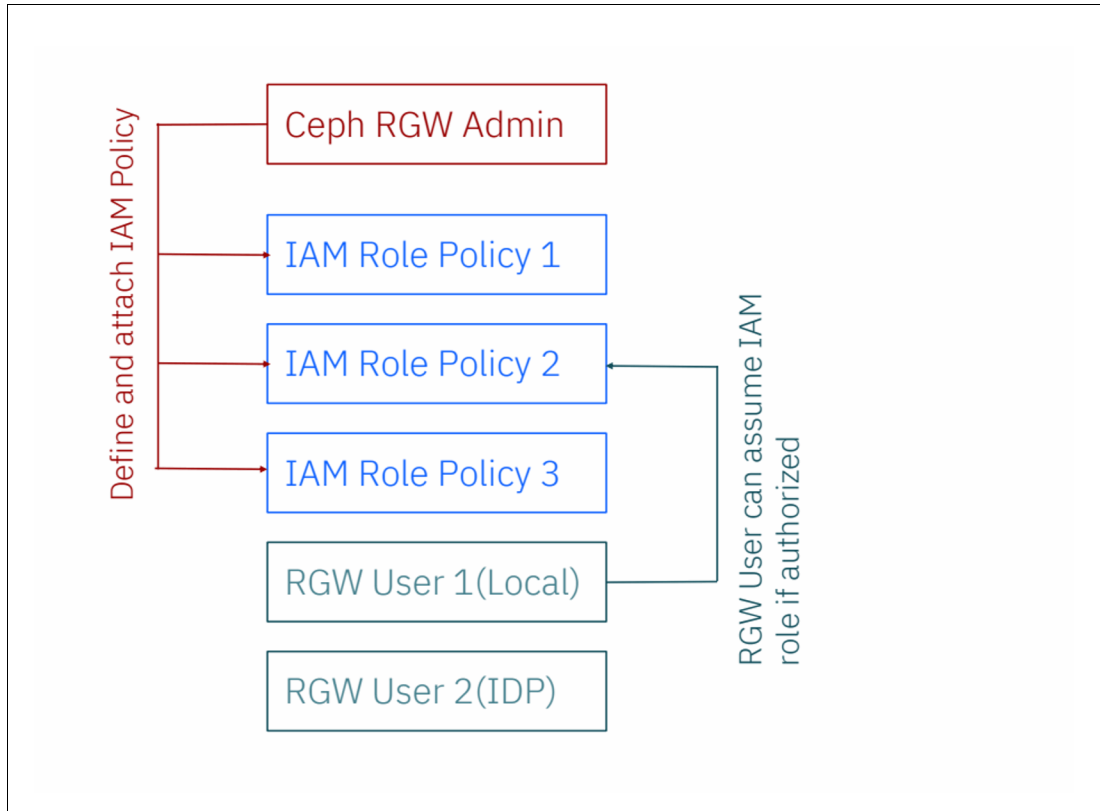


Figure 7-3 IAM role policies,

ABAC authorization model, attribute-based access control (ABAC) is an authorization model that evaluates attributes (or characteristics), rather than roles, to determine access. See Figure 7-4 on page 124.

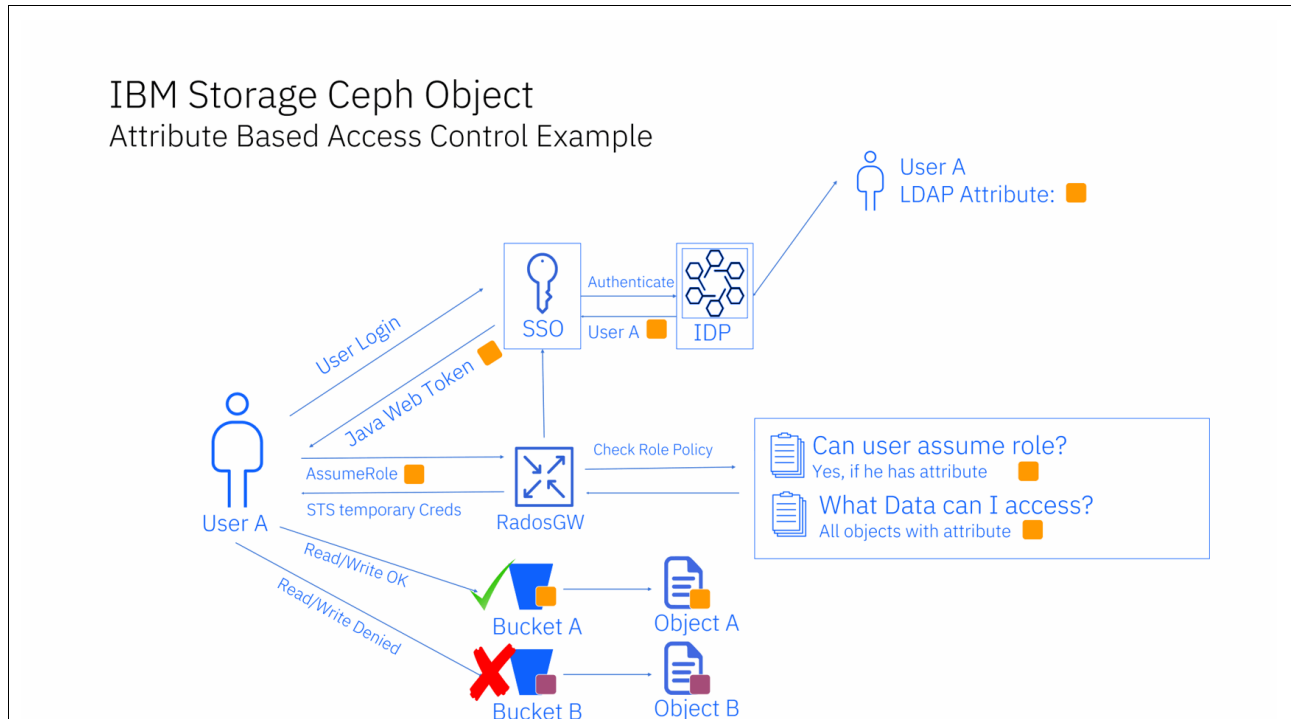


Figure 7-4 Attribute-based access control (ABAC)

## 7.2 Step-by-step deployment guide: S3 enterprise user authentication (IDP) and authorization (RBAC)

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.

We will cover the basic deployment and configuration of Keycloak and IDM, as we need an SSO (OIDC) and an IDP to test the STS/IAM functionality. The configuration we are using for Keycloak and IDM is just for functional testing, use the [RHSSO \(Keycloak\)](#) and [IDM](#) documentation to get a bulletproof production deployment in place.

### 7.2.1 Installation of RH SSO (Keycloak) on OpenShift

Perform the following steps to install RH SSO (Keycloak) on OpenShift.

1. Create a new project on the OpenShift Platform with the name of your preference. See Example 7-1.

*Example 7-1 Create a new project on the OpenShift Platform*

```
[root@ocpbastion ~]# oc new-project kcso
Now using project "kcso" 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. Then, navigate to Red Hat OpenShift Operator Hub and search for **Red Hat Single Sign-On Operator**. See Figure 7-5.

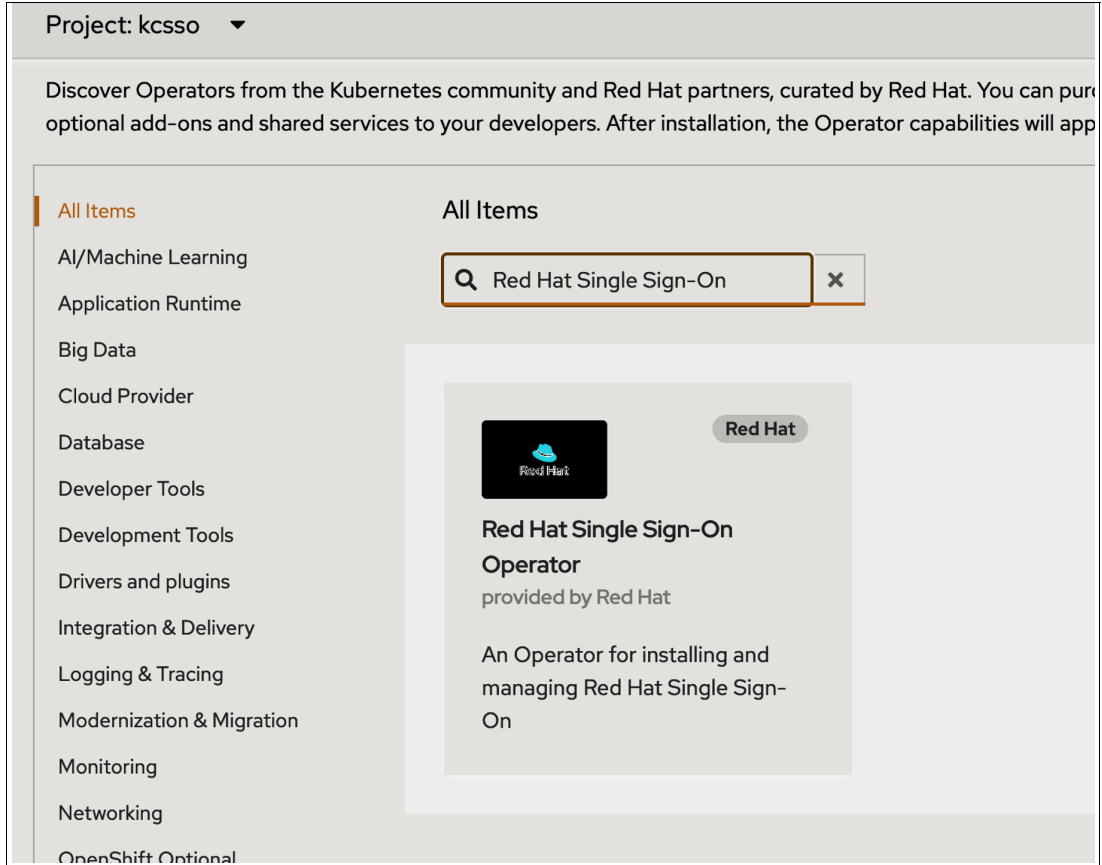


Figure 7-5 Search for Red Hat Single Sign-On Operator

3. Install the Red Hat Single Sign-On operator. See Figure 7-6 on page 126.

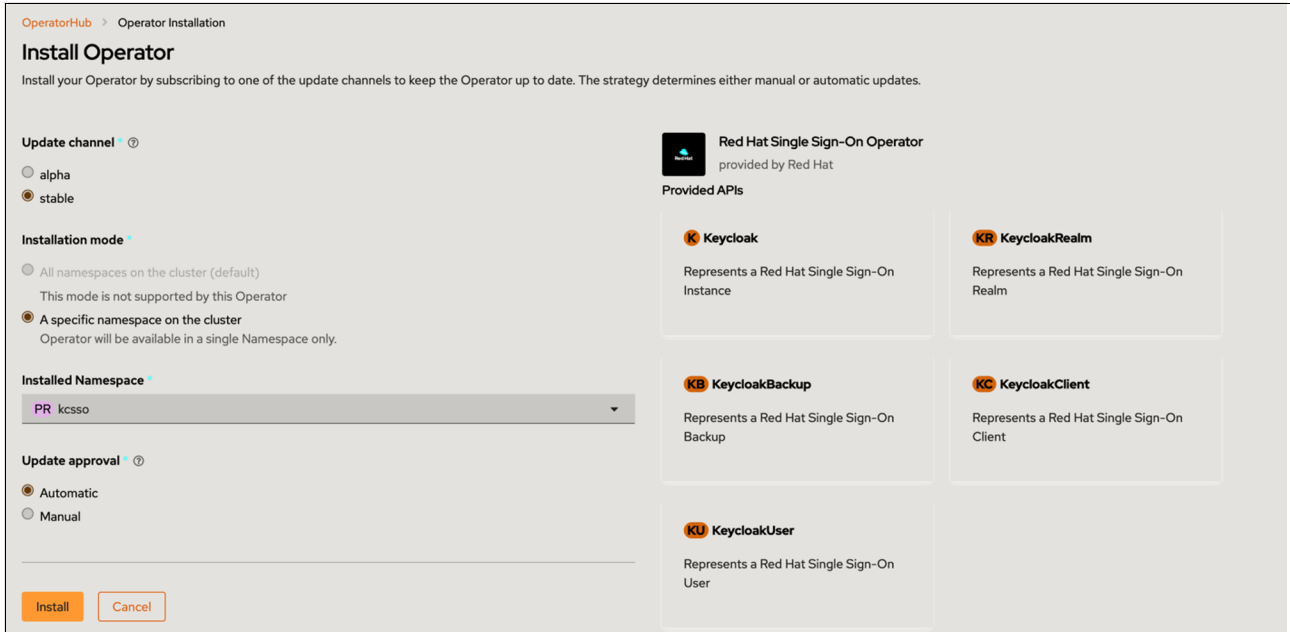


Figure 7-6 Install Operator

- After the installation of the operator is complete, go to the **Red Hat Single Sign-On Operator** page and create a Keycloak instance. See Figure 7-7 on page 126.

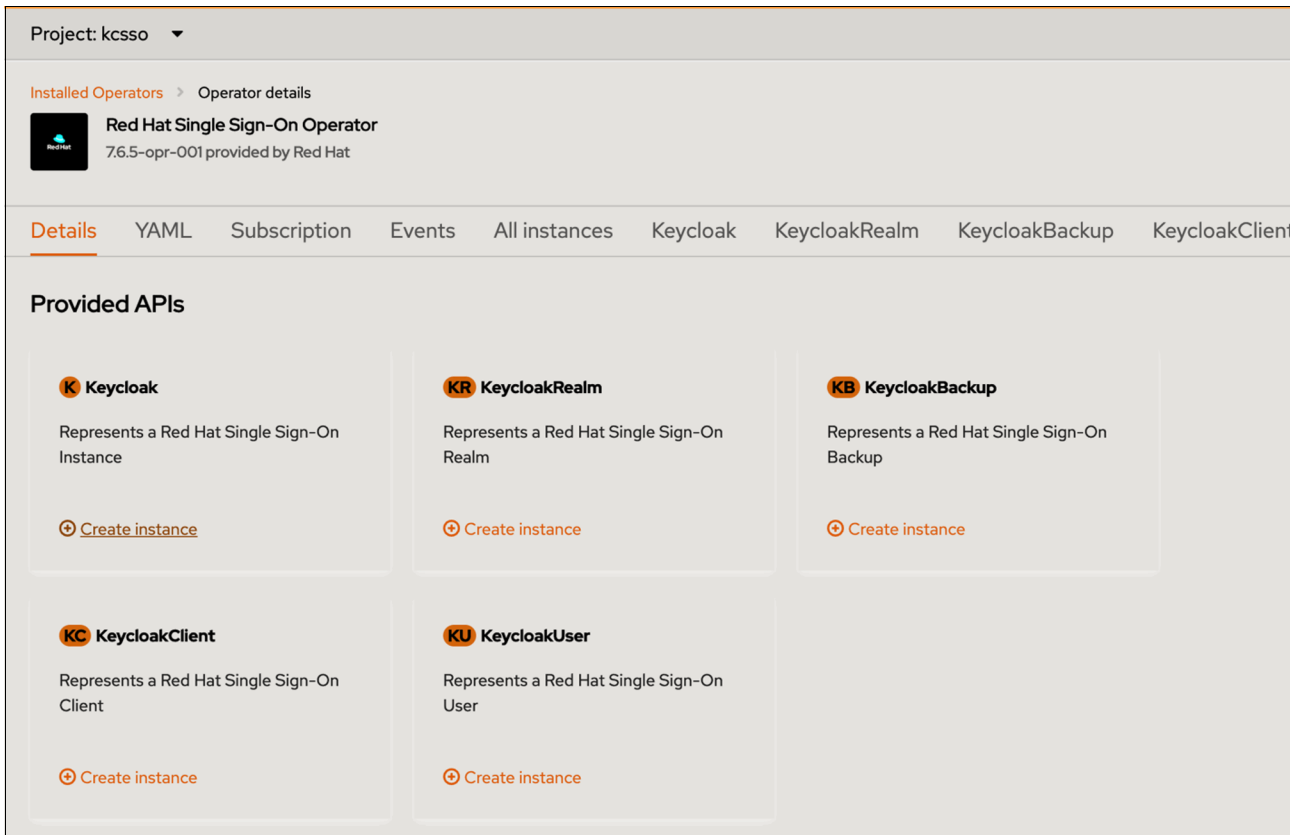
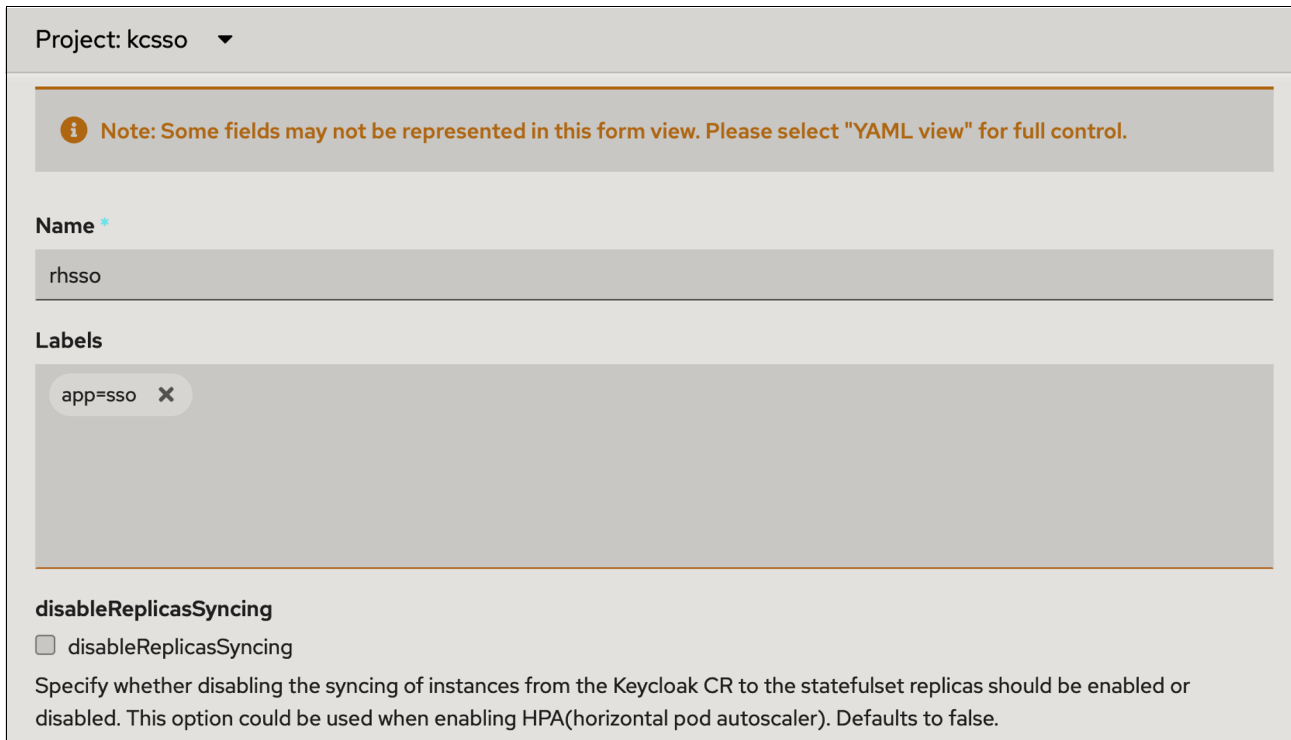


Figure 7-7 Red Hat Single Sign-On Operator

5. Give a name to your Keycloak instance. See 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 ✕

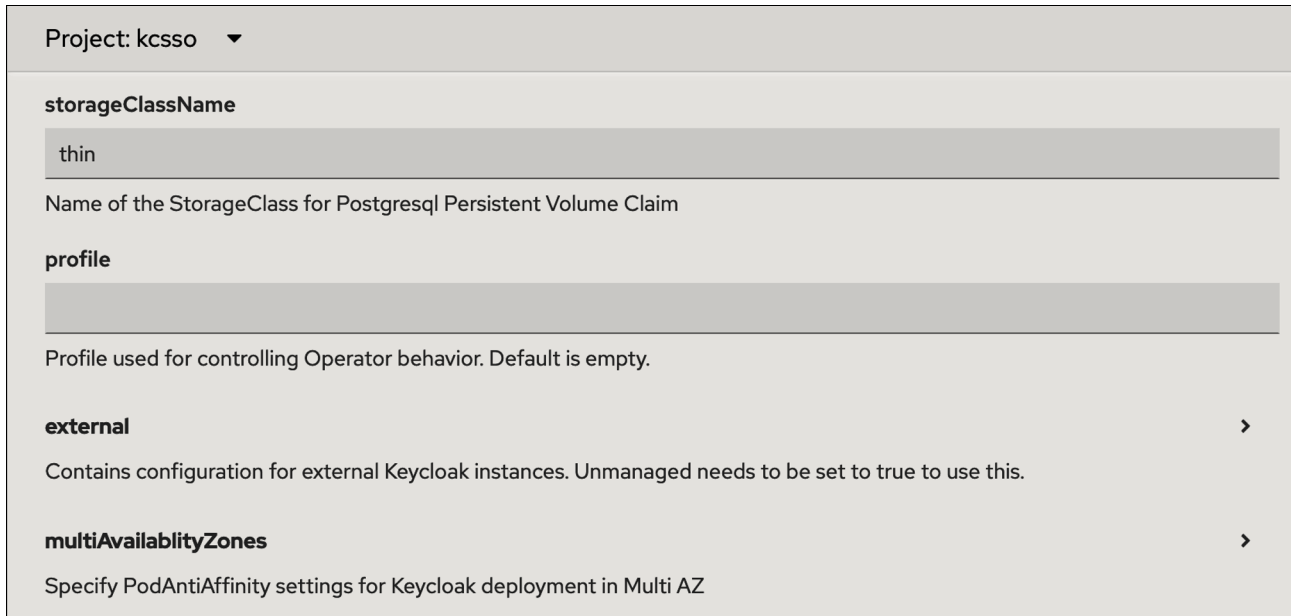
**disableReplicasSyncing**

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 Give a name to your Keycloak instance

6. Since we are not doing an advanced installation of Keycloak, we only need to specify the storage class name we want to use and leave the other settings empty. See 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 will now install a Keycloak instance and a PostgreSQL to keep the records. The storage requirement for the total installation is 2x2GB. The total installation time might take up to 1 hour, depending on the resources of your cluster.

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

## 7.2.2 Installation of Red Hat Identity Management Server

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

- ▶ 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 would like to use an external DNS server with your IdM installation, a list of records that would need to be added to DNS server will be provided by IdM at the end of the installation. The installation will not be complete without adding these records.

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

Before the installation, you need to configure your RHEL system for IdM.

## 7.2.3 Hostname and DNS requirements

As the first step, update your `/etc/hosts` file to include your server IP address with FQDN. Your hostname should not be `localhost` or `localhost6`. See Example 7-2.

*Example 7-2 Update 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 using the `nmtui` tool, for example. The installer may change the hostname to `localhost` during installation, and Apache will refuse to use `localhost` as the hostname and fail to start. This will cause the installer to fail as well.

## 7.2.4 Port requirements

The following ports should be open for IdM. See Example 7-3

*Example 7-3 Ports that need to be open for IdM:*

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

## 7.2.5 Installing packages

IdM installers are located in the BaseOS and AppStream repositories. Enable these repositories with 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 required for installation of IdM with integrated DNS with the following command:

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

*IdM installer requires umask value to be set to 0022 for the root account to allow other users to read files created during the installation.* If this value is not set to 0022, the installer will give a warning, and some functions of the IdM will not be working correctly. You can change this value to its original after the installation. See Example 7-4.

*Example 7-4 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

You can now start the IdM installation. Since we will not perform any configurations during the installation, default values will be used. You can find an example of answers to installer questions in Example 7-5.

*Example 7-5 Start 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 will set 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 host name [idm.stg.local]:
```

The domain name has been determined based on the host name.  
 Please 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.  
 Please 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, named 'admin'.  
 This user is a regular system account used for IPA server administration.  
 IPA admin password: <<ENTER ADMIN PASSWORD>>  
 Password (confirm): <<VERIFY ADMIN PASSWORD>>  
 Trust is configured but no NetBIOS domain name 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 address(es): 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, IdM installation will provide a list of records to be added to your  
 DNS server under the /tmp directory:

Add records in this file to your DNS system: /tmp/ipa.system.records.hazugazn.db.  
 Contents of this list is shown in Example 7-6.

*Example 7-6 Add 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 are required DNS records for your IdM to respond to requests properly. Therefore, the installation will not be complete without adding these records.

If you are using a DNS server that does not support URI record registrations, you can install IdM with `--allow-zone-overlap` option enabled. With this option, you need to make sure your DNS server is forwarding to IdM properly. You can find an example install configuration answers to `--allow-zone-overlap` option as shown in Example 7-7.

*Example 7-7 Install 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 will set 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 host name `[idm.stg.local]`:

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

Please 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.

Please 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, named 'admin'.

This user is a regular system account used for IPA server administration.

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

Checking DNS domain stg.local., please wait ...  
 DNS zone stg.local. already exists in DNS and is handled by server(s):  
 dc001.stg.local. Please make sure that the domain is properly delegated to this  
 IPA server.

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

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, please wait ...

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

Reverse record for IP address 172.16.29.32 already exists

Trust is configured but no NetBIOS domain name 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 address(es): 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 zone(s): 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 will be our LDAP server for RH SSO (Keycloak) integration. RH SSO will be synchronized from IdM with the users and their respective groups.

1. Log in to the IdM interface from your browser using the admin user and the password you have defined during the installation. You will be welcomed with user's page. Navigate to the **Groups** section under the Identity tab and add the following groups:

- ▶ rgwadmins
- ▶ rgwreaders
- ▶ rgwwriters

See Figure 7-10.

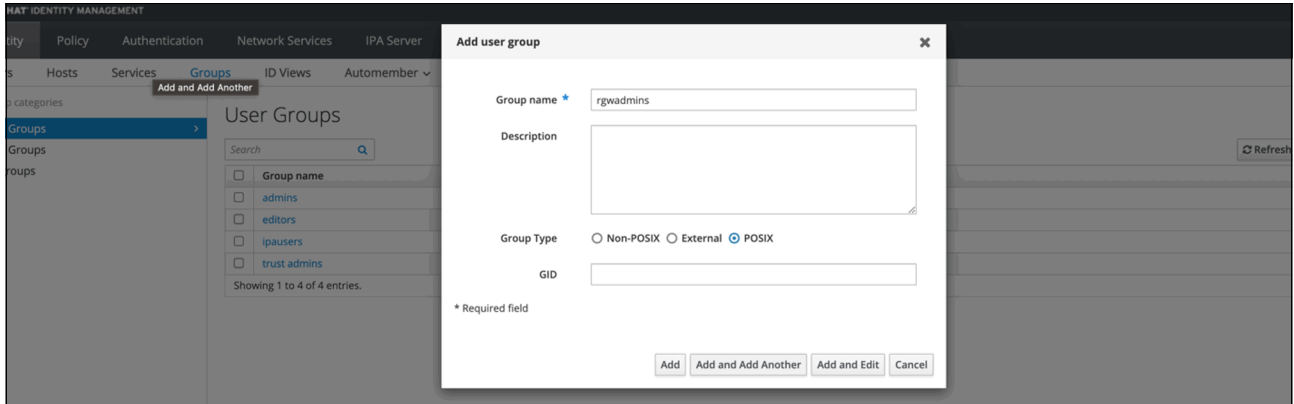


Figure 7-10 Add user groups

2. After creating groups, navigate 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. See Figure 7-11 on page 133.

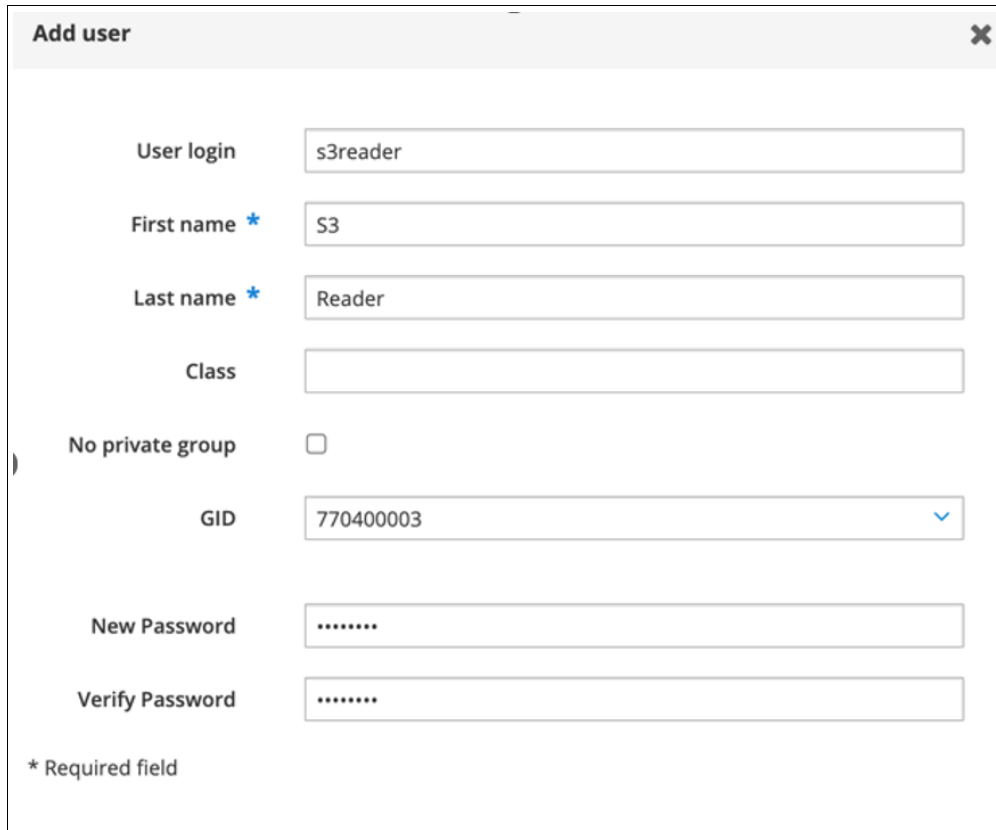


Figure 7-11 Add user

3. After successful user creations (s3admin, s3writer and s3reader) with their respective groups, they should look like the Figure 7-12 on page 134 (The picture shows 3 images combined into one).

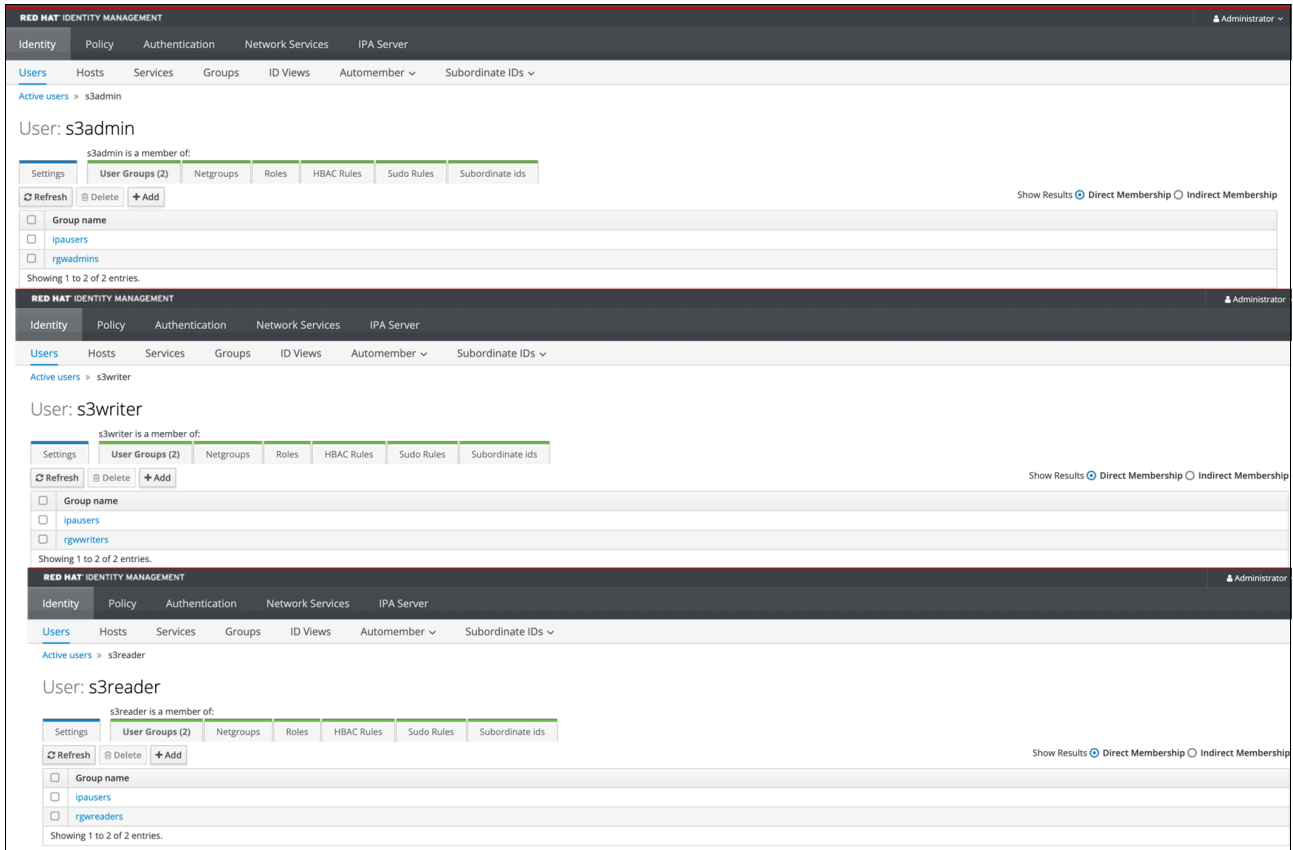


Figure 7-12 Successful creation of users

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

Perform the following steps to configure Red Hat Single Sign-On (Keycloak).

1. Sign in to your RH SSO instance as an admin user and create a new realm with the name Ceph. See Figure 7-13.

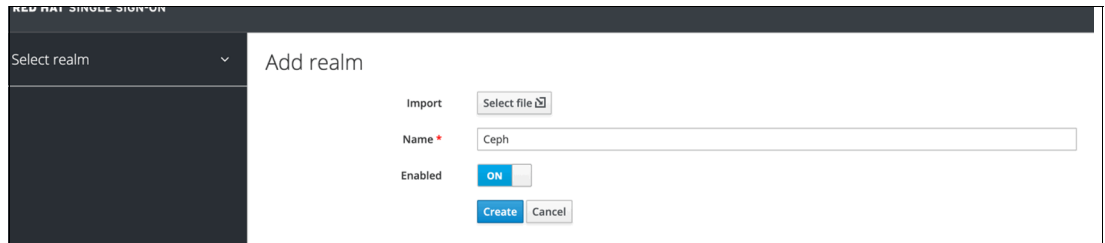


Figure 7-13 Add realm

2. Navigate to **User Federation** from the left side menu and add an LDAP provider. Example settings are shown in Figure 7-14 on page 135.

RED HAT SINGLE SIGN-ON

Ceph

Configure

- Realm Settings
- Clients
- Client Scopes
- Roles
- Identity Providers
- User Federation
- Authentication

Manage

- Groups
- Users
- Sessions
- Events
- Import
- Export

User Federation > Ldap

Ldap

Settings Mappers

Required Settings

Provider ID: a9b4f900-d6fd-4aab-a760-9055664c0b1d

Enabled:  ON

Console Display Name: ldap

Priority: 0

Import Users:  ON

Edit Mode: READ\_ONLY

Sync Registrations:  OFF

Vendor: Red Hat Directory Server

Username LDAP attribute: uid

RDN LDAP attribute: uid

UUID LDAP attribute: nsuniqueid

User Object Classes: inetOrgPerson, organizationalPerson

Connection URL: ldap://idm.stg.local

Users DN: cn=users,cn=accounts,dc=stg,dc=local

Custom User LDAP Filter: LDAP Filter

Search Scope: One Level

Bind Type: simple

Bind DN: uid=admin,cn=users,cn=accounts,dc=stg,dc=local

Bind Credential: .....

Test connection

Test authentication

> Advanced Settings

> Connection Pooling

> Kerberos Integration

> Sync Settings

> Cache Settings

Save Cancel Synchronize changed users Synchronize all users Remove imported Unlink users

Figure 7-14 Add an LDAP provider

3. Enter the admin password to the Bind Credential field and test both connection and authentication separately. You do not need to change other settings below the **Required Settings**.
4. After successfully creating an LDAP provider, save settings, synchronize users, and navigate to the **Mappers** section. We need to create an LDAP Group mapper to map the users to their respective groups during the import process to RH SSO.
5. Click the **Create** button to start adding your LDAP Group Mapper. You can find an example configuration in Figure 7-15 on page 136.

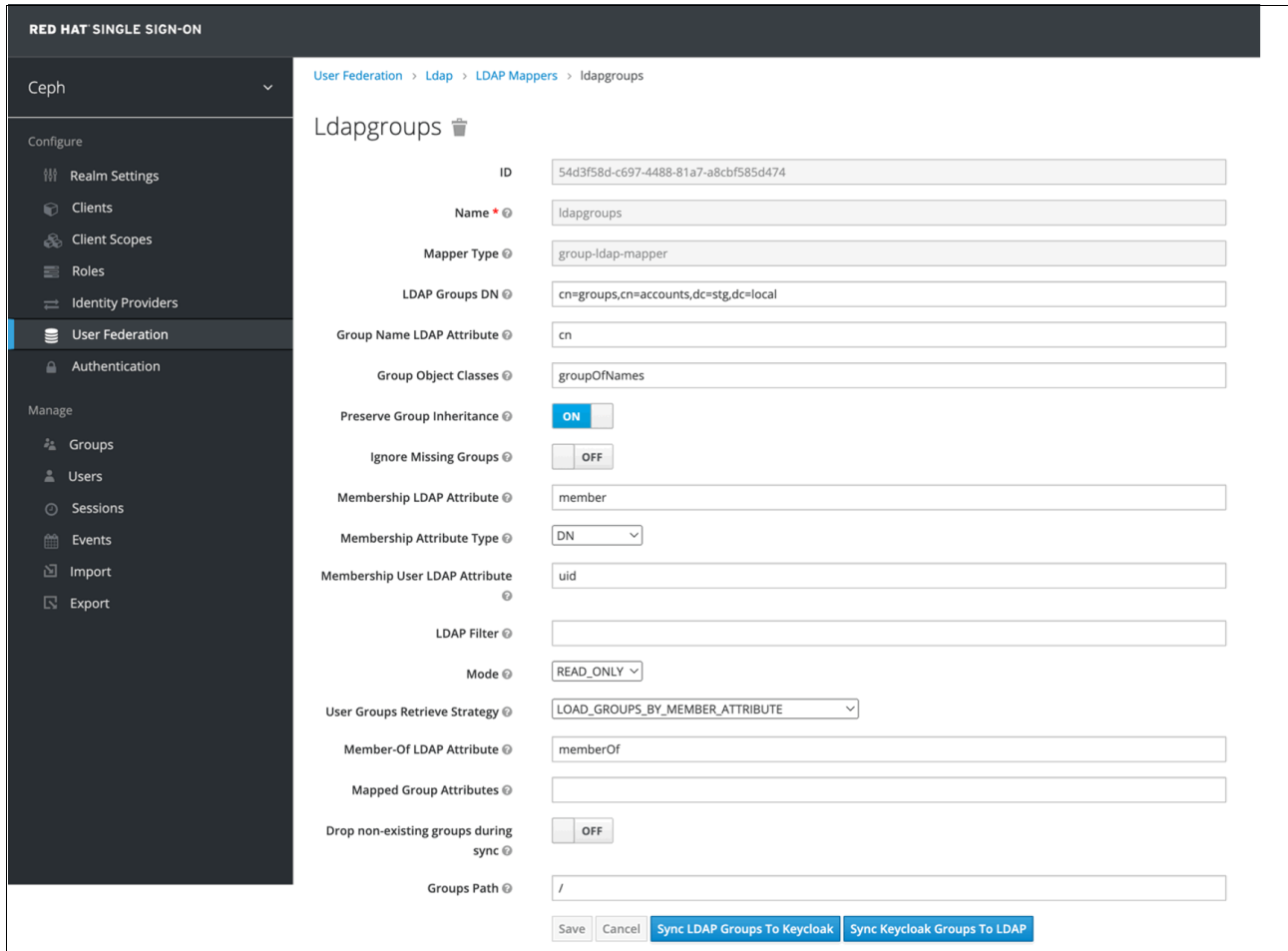


Figure 7-15 Adding the LDAP Group Mapper

6. Save your settings and click **Sync LDAP Groups to Keycloak** button. 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 side menu. See Figure 7-16 on page 137.

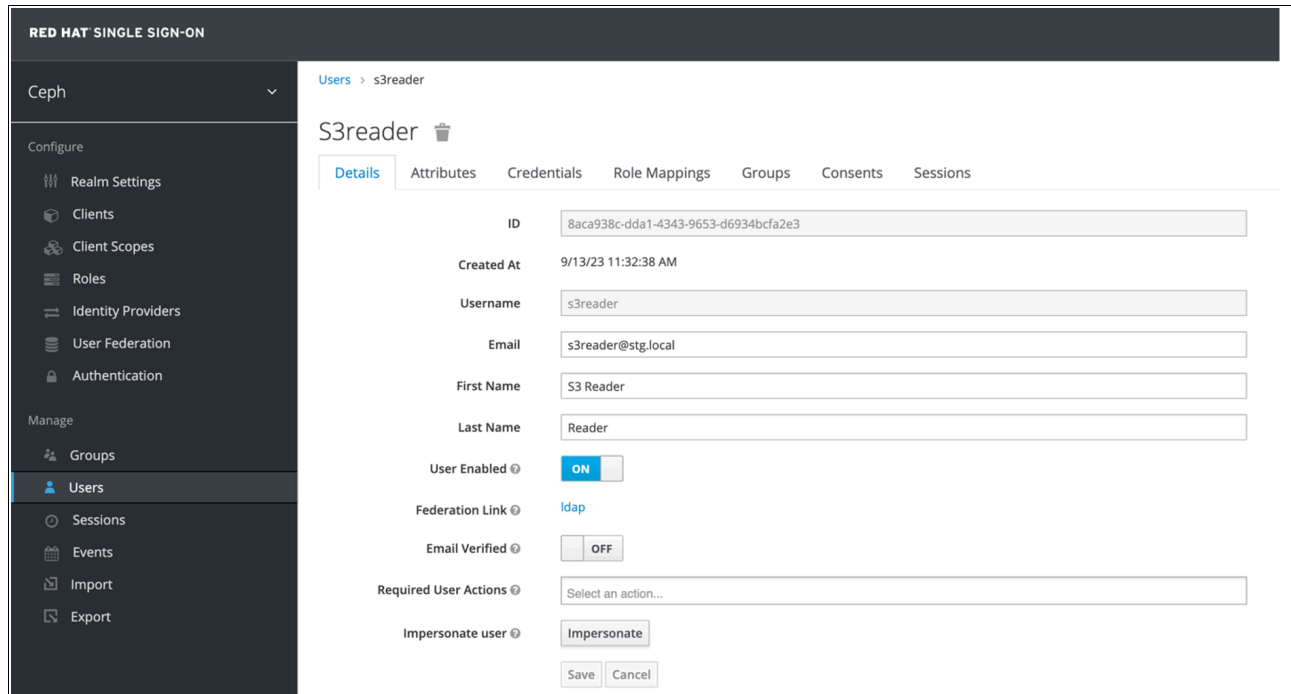


Figure 7-16 Validate the imported users

## 7.2.8 Client configuration

We need to add a client to our RH SSO (Keycloak) to allow connections from our IBM Storage Ceph instance. **Clients** are entities that can make requests to Keycloak for user authentication. It is recommended to use separate clients for each service or application that will be using Keycloak for authentication. We will add the client name to our Rados Gateway using the following steps.

1. To create a client on RH SSO (Keycloak), navigate to the **Clients** section from the left menu and click **Create**. We will be using `openid-connect` as the Client Protocol. See Figure 7-17.

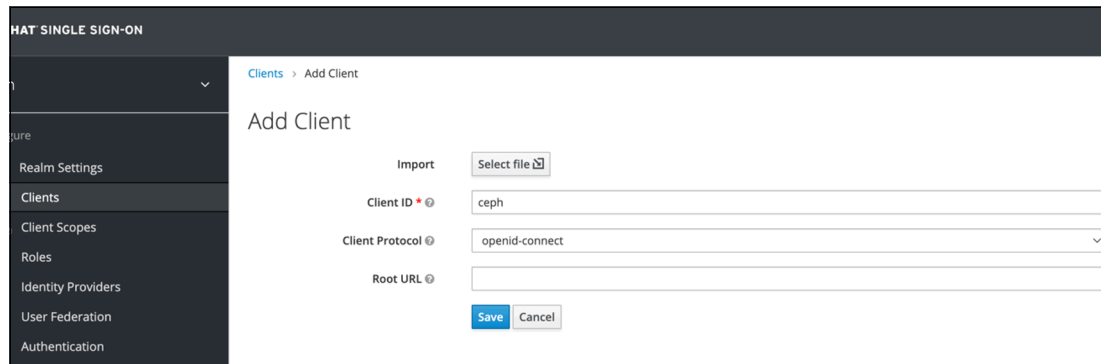


Figure 7-17 Add Client

2. An example Ceph client configuration is shown in Figure 7-18 on page 138. Save the settings after you enter the required values and navigate to the **Credentials** tab. *Take note of the client's secret, as this key will be used to authenticate our Ceph client.*

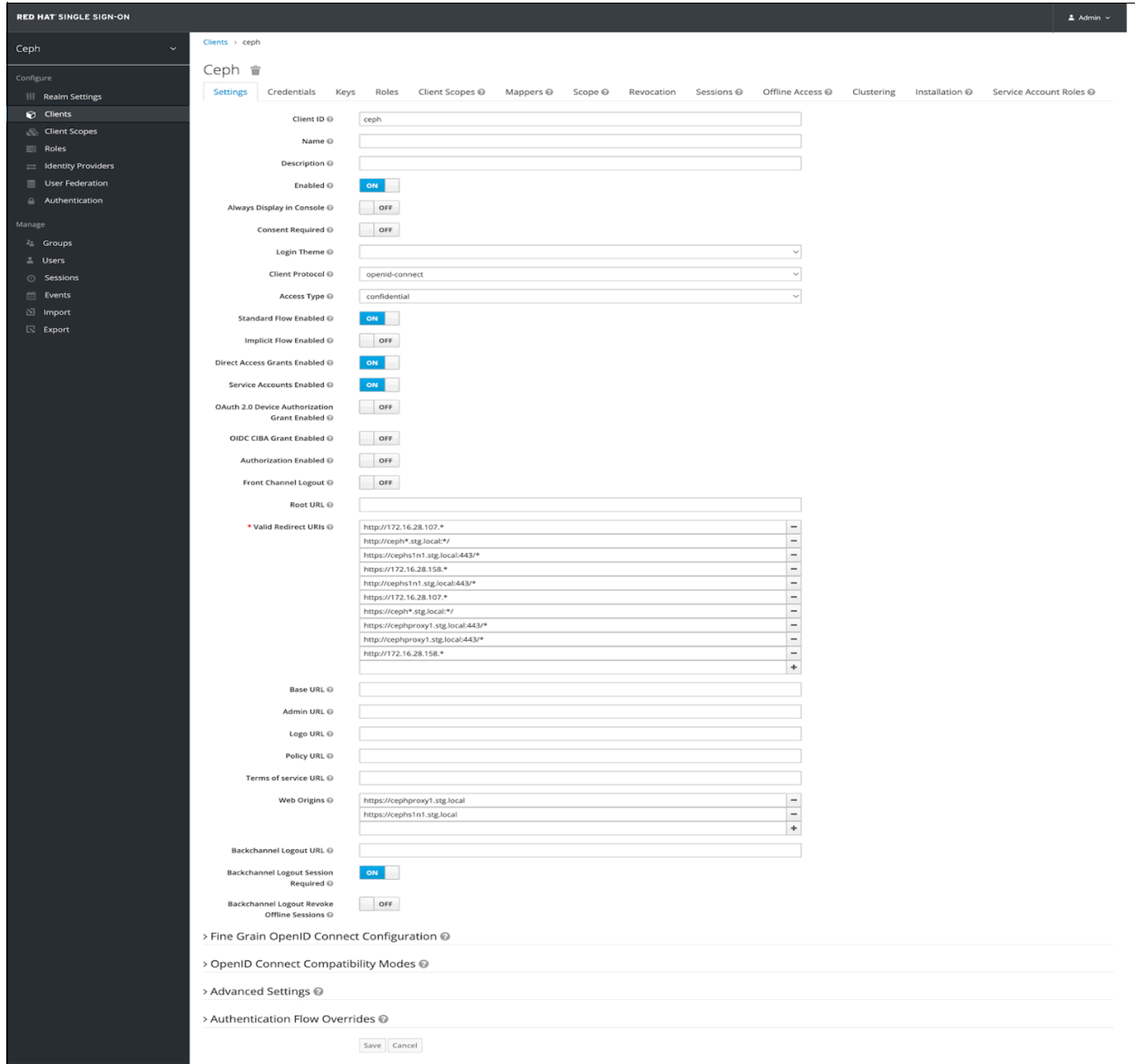


Figure 7-18 An example Ceph client configuration

3. In the following steps, you can always retrieve this value from this page. See Figure 7-19 on page 139.



Figure 7-19 Ceph client secret

We must create Ceph client mappers in types of User Property, Group Membership and Audience. These values will be included in the token we receive from RH SSO (Keycloak) during authentication and verification.

4. Firstly, we will create a User Property type of mapper named username. An example configuration is shown in Figure 7-20.

Figure 7-20 An example configuration - Username

5. Secondly, we need to create a Group Membership type of mapper with the name ceph-groups. An example configuration is shown in Figure 7-21 on page 140.

The screenshot shows the configuration page for a Ceph group named 'ceph-groups'. The page title is 'Ceph-groups' with a trash icon. The configuration fields are as follows:

- 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, there are 'Save' and 'Cancel' buttons.

Figure 7-21 An example configuration - Ceph-groups

6. Finally, we need to create an Audience type of mapper named ceph-mapper. An example configuration is shown in Figure 7-22 on page 140.

The screenshot shows the configuration page for a Ceph mapper named 'ceph-mapper'. The breadcrumb navigation is 'Clients > ceph > Mappers > ceph-mapper'. The page title is 'Ceph-mapper' with a trash icon. The configuration fields are as follows:

- Protocol: openid-connect
- ID: e8403170-a16d-4473-9580-3c3ed0730120
- Name: ceph-mapper
- Mapper Type: Audience
- Included Client Audience: ceph
- Included Custom Audience: (empty)
- Add to ID token: OFF
- Add to access token: ON

At the bottom, there are 'Save' and 'Cancel' buttons.

Figure 7-22 An example configuration - Ceph-mapper

## 7.2.9 Configuring 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, navigate to the **Realm Settings** on the Keycloak interface and click **rsa-generated** on the provider column. Update the Priority value to 105 and click **Save**. See Figure 7-23.

The screenshot shows the Keycloak console interface for the 'Ceph' realm. The 'Keys' tab is selected, and the 'Providers' sub-tab is active. The configuration for the 'rsa-generated' provider is displayed. The fields are as follows:

- 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 of the configuration form.

Figure 7-23 Configuring Ceph Realm settings

## 7.2.10 Verify KC configuration: JWT token retrieval and validation

After successfully implementing RH SSO and IdM, we should be able to retrieve a JWT token from Keycloak and verify it using the user IDs we have created on IdM.

1. First, we need to retrieve the token from Keycloak to do this. An example script for retrieving a token with the specific user can be found in Example 7-8.

*Example 7-8 Retrieve the token from Keycloak*

```
[root@cephp1n1 ~]# cat get_web_token.sh
curl -k -q -L -X POST
"https://keycloak-ss0.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"
```

2. You can run this command using one of the users and passwords. See Example 7-9 on page 141.

*Example 7-9 Output of the command*

```
[root@cephp1n1 ~]# ./get_web_token.sh s3admin passw0rd
{"access_token":"eyJhbGciOiJIUzU1IiwiaXNjaW50eSI6Imc3ZG90aW50eS1kaWUiaWia21kIiA6ICJLUXVMbnJ1MGh1U
UtzY19Ca0FiV2xhVXZ1c0diTWwWncxbURIVDRrU1ZRIj0.eyJ1eHAiOiJlE20TUxMTAyNzMsIm1hdCI6MTY
```



```

    "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) 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) using the OpenID Connect (OIDC) open authentication protocol with RadosGW Secure Token Service (STS) feature, so RGW S3 users can authenticate against the configured SSO to access S3 resources.

### 7.3.1 Registering OpenID Connect (OIDC) provider on IBM Storage Ceph

Since you have now completed the RH SSO (Keycloak) integration with IdM, you must register your OIDC provider to IBM Storage Ceph.

1. First, 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 Rados Gateway, the following command can be used:

```
[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. Then, you need to assign the admin capabilities to this user with the following command:

```
[root@cephs1n1 ~]# radosgw-admin caps add --uid="oidc"
--caps="oidc-provider=*
```

3. Now, you need to obtain the thumbprint for the x5c certificate. You will use this thumbprint in the next step to register the OIDC client on RGW.

An example script for retrieving the x5c thumbprint can be seen in Example 7-12.

#### Example 7-12 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 as shown in Example 7-12 captures the first key in the array: “.keys[0].x5c[]”. Depending on your Keycloak configuration, you may 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, you need to use the key with the use: "sig" parameter. This is the key/cert used to sign JWT tokens, and the one you need to get the thumbprint from for configuring RGW. Example 7-13 shows an example output (This example is not an actual step to take, it is just an example of the warning in this Note).

#### Example 7-13 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_IQ4PvzxDQblyPiBc4cnsQ_b5m07ih0MfG_-nM_qu_kaiVz48ifzNRaCi0fK6H3Yi6u
5-vK70tLRT0Ycxwz4dw416QrKNTF8PKK98Qu_-5_-0yh5HAyGS6mMXhB0Esys1wUF2KCXAwwQ5HMi-QAU1
xiT1bLD7uPTOMAOnqvkoZHG8ZzRJS-X5QAGgXKMJ39mIBBkLjo27U0UTkZpK0QYeZxytQiN1PRrFIOPhsI
M1mm6ivb4vQ9U9w",
    "e": "AQAB",
    "x5c": [

"MIIC1zCCAX8CBgGKdRY7ozANBqkqhkiG9w0BAQsFADAMPQ0wCwYDVQQDDARjZXB0bM4XDTIzMDkwODEzN
TYONVoXDTMzMDkwODEzNTgyNVowDzENMAsGA1UEAwEY2VwaDCCASiWdQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAMyCj0aK3W5EjUX7YkgA/iWBKvo0yhpcCTLJ+ND81qlod7SDcVYSxs6wKIWIuUpN1U5KIaYAp
M1h0t31qjxqp7bTcVUirmztI/yEOD788Q0G9cJ4gXOHJ7EP2+Zt04odDHxv/pzP61P5Go1c+PIIn8zUWgot
Hyuh92IuruFryuzrSOUzmHMcM+HcONekK5DUxfDyivfELv/uf/tMoeRwMhkupjF4QdBLMntcFBdig1wMME
ORzIvKAFJcYk5WYw+7j06DANJ6r5KGRxvGc0SUV1+UABoFyjCd/ZiAQZC46Nu1NFE5GaStEGHmccrUIjZT
OaxSND4bCDNZpuor2+LOPVPcCAwEAATANBgkqhkiG9w0BAQsFAA0CAQEAAwK2iM7r30IYjJc7XMHteeg1Ef
1Dsu+f4zCRVR6Yn1qDmU+UzWZAsEGu+cVxydvfNA3jEzGbs8US0nfw09grX6tDUA5dWaqmIqbQX1hLBJz7
AjuWUv0gh4Dy7r/JSq1ZmDV8kqDxotwJbmcGDI8Hm97U9tN3D2aj0Ympc31i1bif7qYK/1jkIOxQ4ihZfon
ij85/975LosUf/7ScceNkZnfrTr8sbw9UDuup/fXmTBvWf27/tjgd00K8VBDvMQ9rNJYwtXUHuYC1nX7q/
OX/GyENne08rj6w+1n18DKZm0wceevnrjcwH/tkeW309Pckop+NE/2rR4w2GjXFj9tCESIg=="
    ],
    "x5t": "A0nP1pfgsW3RPIaw_9wp1X5dJN8",
    "x5t#S256": "LWJyORSrq_VCKsY1tMZsvUmPFWUuxZFC4a11ULZ2X8k"
  },
  {
    "kid": "KQuLnre0heQKsc_BkAbW1aUvesGbmcpZw1mDHT4kSVQ",
    "kty": "RSA",
    "alg": "RS256",
    "use": "sig", <-- This is the correct x5c key/cert!
    "n":
"uJZsgG51iWUm_rLtkYgjDAIr8cew_7-aU1m7-XBd3vtNt6DGRPSht59BBd1cpt4mg4zAan7x4RUL8ed-
nVnB0cph8DS5VLG10N6CXm7N6FCpPx-eB_ssCjGFIyzLR9cxE3QuDUK3r0_AeEarn9mDw_fkV2edXxFLXC
x01iAI0bSfrAb69iq33LiZh73smhQDC8zHnml1Gs2x2UhjB2_Vfa79-rGHZLsVMoLXBn-hFLTU3Td-auZX
rJHF5rFVKTrgsV1s1a5Ek_hY8Yvhh1D9BzK0gvnp1otIEU42W-gApS4CFG96cJEN4AGYJJW01ScrBnUQ8
6e9HVbOK_eB3Nfw",
    "e": "AQAB",
    "x5c": [

"MIIC1zCCAX8CBgGKdRY6YjANBqkqhkiG9w0BAQsFADAMPQ0wCwYDVQQDDARjZXB0bM4XDTIzMDkwODEzN
TYONVoXDTMzMDkwODEzNTgyNVowDzENMAsGA1UEAwEY2VwaDCCASiWdQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBALiWbIBudY11Jv6y7ZGIiWwCK/HhsP+/m1JZu/1wXd77TbegxkT0oak+fQQxdXKbeJo0MwGp+
8eEVC/Hnfp7zW9HKYfA0uVSxtTjeg15uzehQqT8fngf7LAoxhSMsy0fXMRNOLg1Ct6zvwHhGq5/Zg8P35L
9nnV8RS1wsTtYgCNG0hawG+vYqt9y4mYe97JoUAWwMx55i5RrNsd1IYwdv1X2u/fqXh2S7FTKC1wZ/orS0
1N03fmmV6yRxeaxVS64LL5ebNWuRJP4WPGL4YZQ/QcYtIL56daLSH1ON1voAKUuAhRvenCRDeABmCSVj
pUnKwZ1EP0nr1W9Cv3gdzX8CAwEAATANBgkqhkiG9w0BAQsFAA0CAQEAAwK2iM7r30IYjJc7XMHteeg1Ef
AXu9odebGG14gAWZ00qu0eC4pdSemxENXN+0doNFpJjLs2VTXashORIo4Lx8Nw2P58so1GZec2uFS/Fpt
wk964eDwRcCt3bYbKKEmomsCPdjCCQDs/V3c1WRz3Z1pE+eqKZxN0s0r/9JEU8wcnDUeM2baXDACmvCtWH
tIXyFcyoLzBkEmHxuvKoa92C8VmuJAIzbN8Qjym6TBSotcZ9YbZp99Q/EnrOccuXrEMI/Z1nwURp1miA
LgBu/BZ84bmU2Kz07vcc0zdIoMi1NYfw4QVECIzk9ohzwo2LWa9cflXyF06qtj3pNYUkjQ=="
    ],
    "x5t": "phbucEw9qJthh_zOMBUNMOhHXU",

```

```

    "x5t#S256": "L1z7h1B3aWyuSnyi_wpq2GGH917u552a_FD3sW-Y99s"
  }
]
}

```

---

## 7.3.2 Configuring certificates

We will configure IBM Storage Ceph Object Storage (RGW) to use HTTPS/SSL. We will terminate SSL at the load balancer/Ingress service. With this configuration, the communications between the S3 clients and the load balancer will be encrypted, but the communications from the load balancer to the RGWs will be in the clear. In this example, we will use self-signed certificates, but you could also use trusted CA-signed SSL certificates.

First, you must create an SSL certificate on the Rados GW host. To do this, the following commands can be used.

1. Create a `certs` directory under `/root` and create the configuration file for the certificate with Subject Alternate Names (SAN). We can use the certificate with the domains defined in the SAN area. An example of a configuration file is shown in Example 7-14.

*Example 7-14 Create a `certs` directory under `/root` and create 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. Then, you need to create a key for the certificate. See Example 7-15.

*Example 7-15 Create a key for the certificate*

---

```

[root@cephproxy1 certs]# openssl genrsa 2048 > cert.key
[root@cephproxy1 certs]# chmod 400 cert.key

```

---

3. Finally, you can create your certificate file with the following command. See Example 7-16.



*Example 7-16 Create 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. You can verify the certificate SANs with the following command. See Example 7-17.

*Example 7-17 Verify 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, you must first transfer both cert.pem and cert.key files to /etc/pki/ca-trust/source/anchors directory under ceph1n1.stg.local host. This can be done using SCP:

*Example 7-18 Transfer cert.pem and cert.key files*


---

```
[root@cephproxy1 certs]# scp cert.pem cert.key
root@ceph1n1.stg.local:/etc/pki/ca-trust/source/anchors
```

---

6. After the transfer, you must import the certificate into the system. The following commands, as shown in Example 7-19, can be used to import the certificate to the system and update the ca-bundle:

*Example 7-19 Import the certificate into the system*


---

```
[root@ceph1n1 ~]# update-ca-trust
[root@ceph1n1 ~]# update-ca-trust enable
[root@ceph1n1 ~]# update-ca-trust extract
```

---

### 7.3.3 Configuring RGW in HA mode with load balancing and SSL

The ingress service allows you to create a highly available 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. The ingress service is deployed on N hosts, each of which has a haproxy daemon and a keepalived daemon. A virtual IP 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 will be 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 on page 148 shows the ingress service.

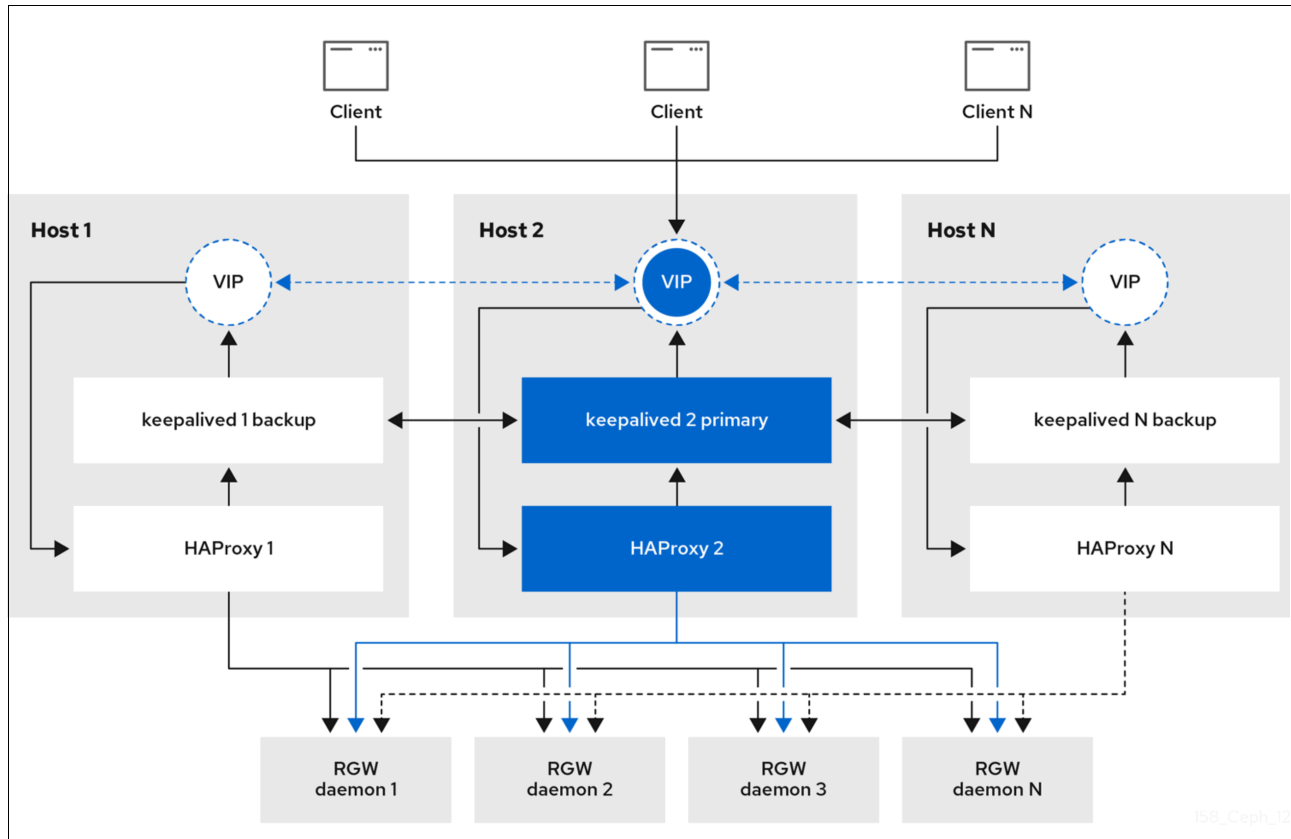


Figure 7-24 The ingress service

After a successful import of the certificate file, you need to configure the RGW with SSL ingress.

1. You must first label the RGW host with `rgws` to use it with the RGW configuration file.

The following command can label the `cephproxy1` node. See Example 7-20.

*Example 7-20 Label the `cephproxy1` node*

```
[root@cephs1n1 ~]# ceph orch host label add cephproxy1.stg.local rgws
Added label rgws to host cephproxy1.stg.local
```

2. You can verify the host label with the following command. See Example 7-21.

*Example 7-21 Verify 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 using port 8443. To do this, you will need to create a YAML file that includes the certificate and key file for the RGW. An example config file can be used to implement an RGW service with SSL

ingress. Note that this YAML file should be created on the RGW node and will be deployed from there. See Example 7-22.

*Example 7-22 Configure an RGW service with an SSL 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
```

---

- Note that the `/etc/pki` directory is bind mounted into the container that the RGW service will be running in. This ensures that the RGW service will always use the latest updates of the certificates imported into the RGW host.

### 7.3.4 Register the OIDC provider with RadosGW

Next, we need to register the OIDC provider with RadosGW.

- You can use the following Python code to register the OIDC provider to Rados GW. See Example 7-23.

*Example 7-23 Python code to register the OIDC provider to Rados GW*

---

```
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", required=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 following arguments. See Example 7-24.

*Example 7-24 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, you can verify that the OIDC provider has been added to Rados GW with the following command. See Example 7-25.

*Example 7-25 Verify that OIDC provider is added to the Rados GW*

---

```

[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

We now need to assign roles and policies to these roles according to the groups we created in previous steps. To do this, we will first create roles: `rgwadmins`, `rgwwriters`, and `rgwreaders`.

#### RGW Admins role

To start with we will create a role for `rgwadmins`.

First, create a JSON file defining the role properties. This JSON policy document will allow access to the `rgwadmins` role to any user who has been authenticated by the SSO (OIDC provider) and is part of the IDM/LDAP `rgwadmins` group. The condition checks for a match on the JWT token and will allow the user to assume the role if it finds a `StringLike` `rgwadmins` value in the JWT groups section of the token. See Example 7-26.

*Example 7-26 Create 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"
          ]
        }
      }
    }
  ]
}
```

---

Create an RGW role using the JSON file. See Example 7-27.

*Example 7-27 Create an RGW role 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":
  "{\\"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\\"]}}}}]"}"
}
```

---

**RGW Writers role**

Next, we need to create a role for rgwwriters. See Example 7-28.

*Example 7-28 Create a role for rgwwriters*


---

```
[root@cephs1n1 ~]# 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"
        }
      }
    }
  ]
}
```

---

Create a role for rgwwriters using the JSON file. See Example 7-29 on page 152.

*Example 7-29 Create a role for rgwwriters using the JSON file*


---

```
[root@cephs1n1 ~]# 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":
    "{ \"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\": \"rgwwriter s\" } } } ] }"
  }

```

---

## RGW Readers role

Finally, we need to create a role for rgwreaders. This JSON policy document will allow access to the rgwreaders role to any user who has been authenticated by the SSO (OIDC provider) and is part of the IDM/LDAP rgwreaders group. The condition checks for a match on the JWT token and will allow the user to assume the role if it finds a StringLike rgwreaders value in the JWT groups section of the token. See Example 7-30.

*Example 7-30 Create a role for rgwreaders*

---

```

[root@cephs1n1 ~]# 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"
        }
      }
    }
  ]
}

```

---

Create a role for rgwreaders using the JSON file. See Example 7-31.

*Example 7-31 Create a role for rgwreaders using the JSON file*

---

```

[root@cephs1n1 ~]# 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":
    "{ \"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\": \"rgwreader s\" } } } ] }"
  }

```

---

### 7.3.6 Attaching policies to roles

After successfully creating roles on RGW, we need to attach policies to define the capabilities of users within those groups.

1. To start with, we will create a role policy for rgwadmins. In this policy, we are allowing the user that has assumed the rgwadmins role to do any S3 action on any S3 resource. See Example 7-32.

*Example 7-32 Create 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 rgwadmins role. See Example 7-34.

*Example 7-33 Apply the role policy for 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. Next, you will create a role policy for rgwwriters. See Example 7-34.

*Example 7-34 Create 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. See Example 7-35.

*Example 7-35 Attach 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. Finally, we will create a policy for rgwreaders. See Example 7-36.

*Example 7-36 Create 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. See Example 7-37.

*Example 7-37 Apply 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
```

---

7. Note that you can define multiple policies at the same time for a specific group. An example of a multiple-policy definition is shown in Example 7-38. You can refer to [AWS documentation](#) for more examples.

*Example 7-38 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 RadosGW service

We need to enable STS on RadosGW service that we are going to use. To do this, the following commands can be used. See Example 7-39.

*Example 7-39 Enable STS on RadosGW 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 Keycloak certificate to RGW node

Since Keycloak is using an SSL connection for the requests and running on OpenShift, we need to import the certificate files of OpenShift Ingress Service into our RGW node. There are many ways to retrieve the Keycloak certificate chain, here is one example.

1. To import the OpenShift certificate, open a Firefox browser and navigate to the Keycloak URL, which is `https://keycloak-sso.apps.ocp.stg.local/` in our 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.

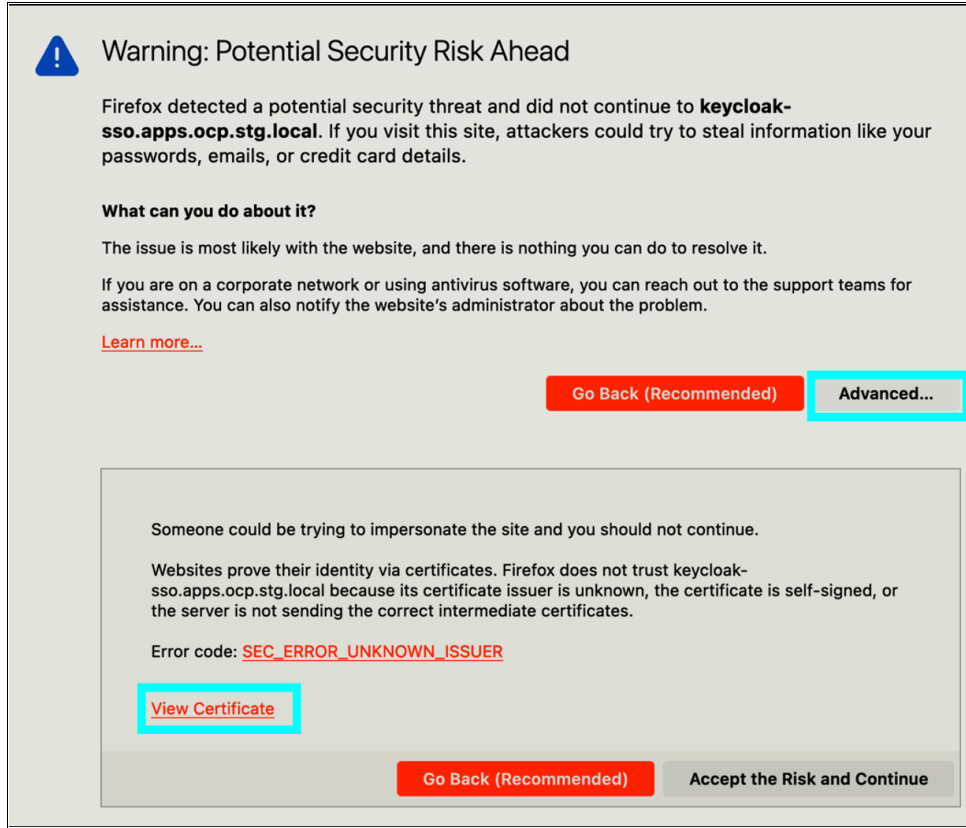


Figure 7-25 View Certificate

3. On the `*apps.ocp.stg.local` tab, navigate to **Miscellaneous** section and download the chain certificate file. See Figure 7-26.



Figure 7-26 Download 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 with the following commands. See Example 7-40 on page 158.

*Example 7-40 Update your system's certificate bundle and restart 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 to Keycloak's main page. See Example 7-41.

*Example 7-41 Verify 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 successful import of the certificates into the system, you can test the Assume Role With Web Identity functionality for role based access control of your IBM Storage Ceph cluster.

An example script for the test is shown in Example 7-42. Note that the `AWS_CA_BUNDLE` argument is pointing to the certificate that was created on the RGW node.

*Example 7-42 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 Assume role with Admin role

Running the example script in Example 7-43 with the admin role should give you the Admin role on buckets.

*Example 7-43 Running the example script with 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 we ran in the previous step takes care of exporting 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 provided by the RGW, as you can see on the next step.

Running the script with the `s3admin` user grants you admin privileges. This means you will have full control over the buckets and can perform any actions you desire. You can test the functionality with the following commands, as shown in Example 7-44.

*Example 7-44 Test the functionality*

```

[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 Assume role with Writer role

Running the script with s3writer role will give you the writer role on buckets. See Example 7-45.

*Example 7-45 Running the example script with 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 will only be able to write to the bucket. You can test the functionality with the following commands, as shown in Example 7-46.

*Example 7-46 Test the functionality*

---

```
[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 Assume role with Reader role

Running the script with s3reader role will give you the reader role on buckets. See Example 7-47.

*Example 7-47 Running the example script with 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 will only be able to list the bucket contents and retrieve objects from it. You can test the functionality with the following commands, as shown in Example 7-48.

*Example 7-48 Test the functionality*

---

```
[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
```

---





# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this paper.

## IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- ▶ *IBM Storage Ceph Concepts and Architecture Guide*, REDP-5721

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft 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 DocISBN

Printed in U.S.A.

Get connected

