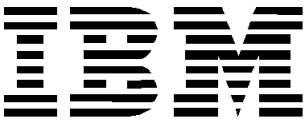
IBM® Storage

IBM Storage Solutions for IBM Cloud Private Blueprint

IBM Storage Team



Contents

About this document

This Blueprint is intended to facilitate the deployment of IBM Storage Solutions for IBM Cloud Private by using detailed hardware specifications to build a system and describe the associated parameters for configuring persistent storage within an IBM Cloud Private environment. To complete the tasks, you must have an understanding of IBM Cloud Private, IBM Spectrum Connect and IBM Storage Enabler for Containers.

Tip: Beginning with v3.4.0, IBM Spectrum Control™ Base Edition is now IBM Spectrum Connect. To learn more about the rebranding transition, see the IBM Knowledge Center at:

https://www.ibm.com/support/knowledgecenter/en/SS6JWS/landing/IBM_Spectrum_Connect_welcome_page.html

The information in this document is distributed on an “as is” basis without any warranty that is either expressed or implied. Support assistance for the use of this material is limited to situations where IBM Storwize and/or IBM FlashSystem storage devices are supported and entitled and where the issues are not specific to a blueprint implementation.

This edition of the document applies to IBM Storage Solutions for IBM Cloud Private Version 1 Release 5.0.

Support for the Blueprint and its configurations

The information in this document (referred to throughout as “the Blueprint”) is distributed on an “as is” basis without any warranty that is either expressed or implied. Support assistance for the use of this material is limited to situations where IBM Spectrum Connect support is entitled and where the issues are not specific to a blueprint implementation.

Support of the underlying IBM Spectrum Connect components is entitled and provided as an extension of the related Storage hardware and system software. Please refer to the hardware and system software documentation for more information on how to request assistance and support for the IBM Spectrum Connect components.

Prerequisites

This Blueprint assumes familiarity with and basic knowledge of:

- IBM Cloud Private 3.1.2
- Kubernetes container orchestrator
- VMware vSphere version 6.0 or later
- IBM PowerVC
- IBM Spectrum Connect and IBM Storage Enabler for Containers
- IBM Z®, x86 and IBM Power compute
- IBM z/VM® hypervisor
- IBM Storwize and IBM FlashSystem all-flash storage arrays
- IBM Spectrum Scale™

What’s new in Version 1 Release 5.0

The documentation for the Blueprint configuration, hardware and software requirements has been updated for:

- Support for Helm chart installation of Storage Enabler for Containers
- IBM Spectrum Connect 3.6.1
- IBM Storage Enabler for Containers 2.1

Executive summary

Overview: Microservice provision is an emerging architectural design pattern that brings agility and scalability to enterprise applications.

- Challenge: A challenge of implementing microservices has been the need for infrastructure to dynamically spin up and connect these services.
- Solution: The dynamic nature of cloud platforms is a key enabler in the shift to a microservices strategy. The end-to-end private cloud solution described in this Blueprint documents the essential private cloud service fabric for developing and managing on-premises containerized applications.

To gain a competitive advantage for your business, you need a reliable, secure and flexible IT environment—one that enables modern application enterprise workloads by scaling as necessary to fit your needs, and that gives access to users, no matter what kind of endpoint device they're using. Further, it should allow orchestration, both to suit your resource consumption requirements and to minimize downtime. Your environment must provide reliable platform-as-a service capabilities with flexible infrastructure. This means deploying a cloud-service fabric to reliably deliver containerized applications to your endpoints of choice, to meet or exceed service-level expectations.

Additionally, organizations must protect data, whether for highly regulated industries or when building mission-critical applications. Getting to market quickly, iterating, and attracting new customers are top of mind for executives around the world, and even though cloud computing is a major force in business innovation, challenges are everywhere. Your cloud is only as private and secure as the technology that that protects it allows. As organizations implement modern application platforms, they are leveraging technologies to deliver cloud-native workloads, provide stateful data services, and deliver enterprise-critical capabilities from artificial intelligence and messaging to blockchain applications, DevOps, analytics and high-performance computing.

To this end, the full-stack IBM Cloud Private cloud solution documented in this Blueprint delivers a private cloud service fabric for building and managing on-premises, containerized applications that can deliver scale, performance, security and data-protection, and that can extend across hybrid and multicloud environments to fill your most critical application requirements. The possibilities are endless, and real-time decision-making is within reach.

Scope

This Blueprint provides:

- A solutions architecture and the related storage endpoint capabilities that interact with the following software and hardware components:
  - IBM Cloud Private 3.1.2
  - A flexible choice of compute and storage resources
  - IBM Spectrum Connect
  - IBM Storage Enabler for Containers
- Detailed technical configuration steps for building an end-to-end private cloud solution

This Blueprint does not:

- Provide performance analysis or metrics for end-user consumption
- Replace any official manuals and documents issued by IBM for related products
- Explain installation and configuration of VMware vSphere

Getting started: End-to-end private cloud solution building blocks

This section describes the end-to-end private cloud solution architecture to facilitate a smooth deployment experience.

Helm

Helm, the Kubernetes native package management system, is used for application management inside an IBM Cloud Private cluster. The Helm GitHub community curates and continuously expands a set of tested and preconfigured Kubernetes applications. You can add items from this catalog of stable applications to your cluster from the management console.

Helm charts describe even the most complex applications; provide repeatable application installation, and serve as a single point of authority. Helm charts are easy to update with in-place upgrades and custom hooks. Charts are also easy to version, share, and host on public or private servers. You can use *helm rollback* to roll back to an older version of a release with ease.

IBM Cloud Private 3.1.2

IBM Cloud Private is an application platform for developing and managing on-premises, containerized applications. It is an integrated environment for managing containers that includes the container orchestrator Kubernetes, a private image registry, a management console and monitoring frameworks.

IBM Cloud Private delivers a customer-managed container solution for enterprises. It is also available in a community edition, IBM Cloud Private-CE, which provides a limited offering that is available at no charge and ideal for test environments.

For the best experience in using IBM Cloud Private, you must understand how Kubernetes, Docker and Helm work. These open-source components are fundamental to the IBM Cloud Private platform. Kubernetes manages the deployment of application instances, which are built into Helm charts that reference Docker images. The Helm charts contain the details about your application, and the Docker images contain all the software packages that your applications need to run.

As of release 1.5, this Blueprint describes a set of additional software packages and middleware support that are currently available, as shown in Table 1.

Table 1 Operating systems supported by IBM Cloud Private

Vendor	Operating system
Red Hat	Enterprise Linux (RHEL) 7.4, 7.5 and 7.6 (64-bit)
Canonical	Ubuntu 18.04 LTS and 16.04 LTS
SUSE	Linux Enterprise Server (SLES) 12 SP3

IBM Spectrum Connect 3.6.1

IBM Spectrum Connect is a centralized server system that consolidates a range of storage provisioning, automation and monitoring solutions through a unified server platform. It provides a single-server back-end location and enables centralized management of storage resources for different virtualization and cloud platforms.

IBM Spectrum Connect facilitates the integration of IBM storage system resources by taking advantage of features provided by independent software vendor (ISV) products and solutions. IBM Spectrum Connect version 3.6 supports different IBM storage systems, including the IBM DS8880 family; IBM Storwize family; IBM FlashSystem 9100, IBM FlashSystem A9000, IBM FlashSystem A9000R and IBM FlashSystem V9000; IBM Spectrum Accelerate and IBM XIV; and IBM Spectrum Virtualize.

For a complete list of the supported IBM Storage Systems and respective microcode levels, refer to the IBM Knowledge Center for IBM Spectrum Connect at:

https://www.ibm.com/support/knowledgecenter/en/SS6JWS/landing/IBM_Spectrum_Connect_welcome_page.html

IBM Spectrum Connect policy-based automated block storage provisioning can now be applied to container environments, VMware environments and mixed environments as shown in Table 2.

Table 2 Operating systems supported by IBM Spectrum Connect

Vendor	Operating system
Red Hat	RHEL 6.3-6.9, 7.0-7.5 (64-bit)
CentOS	CentOS 7.x

Compute platforms support

Table 3 notes the compute environment support documented as of release 1.4 of this Blueprint.

Table 3 Compute environment support

Vendor	Model	Architecture	Virtual machine (VM) environment
IBM	z14 ZR1	s390x	z/VM
IBM	IBM z13®	s390x	z/VM
IBM	[a]IBM POWER8® (S8xxL/E8xx)	ppc64le	IBM PowerVM®
IBM	[a]POWER8 (S8xxLC)	ppc64le	KVM
IBM	[a]POWER9 (S9xxL)	ppc64le	PowerVM
IBM	[a]POWER9 (LC921/LC922)	ppc64le	KVM
VersaStack	UCS B-Series	x86	VMware
Other	x86-64 servers (bare metal)	x86	None

a. Note: Reference “Appendix C. IBM PowerVC FlexVolume driver setup on IBM Cloud Private” when configuring Power server environments using IBM PowerVC.

Block storage platforms support

Table 4 notes the IBM Storage platform support documented as of release 1.4 of this Blueprint.

Table 4 IBM Storage platform support

Vendor	Model	Product family
IBM	IBM Storwize® V5000 family	IBM Spectrum Virtualize
IBM	IBM Storwize V7000 family	IBM Spectrum Virtualize
IBM	IBM FlashSystem V9000	IBM Spectrum Virtualize
IBM	IBM FlashSystem 9100	IBM Spectrum Virtualize
IBM	IBM FlashSystem A9000	IBM Spectrum Accelerate
IBM	IBM FlashSystem A9000R	IBM Spectrum Accelerate
IBM	IBM XIV® Gen 3	IBM Spectrum Accelerate
IBM	IBM DS8880	(none)

The Blueprint provides a foundation for deploying and scaling small, medium or large environments that can take advantage of flexible compute or storage resources to meet myriad workload demands that are being delivered through the IBM Cloud Private catalog.

When deciding which solution to deploy, you should consider capacity, performance, data management criteria and cost aspects for the total solution.

IBM Spectrum Scale

IBM Spectrum Scale is a parallel, scale-out, high-performance solution consolidating traditional file-based and new-era workloads to support artificial intelligence, data lake and object storage, Hadoop, Spark and analytics use cases. IBM Spectrum Scale helps clients optimize for cost and performance using intelligent data management that automates movement of data to the optimal storage tier without end-user impact. IBM Spectrum Scale is known for performance and reliability, providing data storage for some of the largest compute clusters in the world.

IBM Spectrum Scale v5.0 is required on all IBM Spectrum Scale nodes in the Kubernetes cluster (all nodes that run the IBM Storage Enabler for Containers code). Refer to the IBM Spectrum Scale Software Version Recommendation Preventive Service Planning for recommendations on the exact IBM Spectrum Scale v5.0 level to use: http://www-01.ibm.com/support/docview.wss?uid=ssg1S1009703

Table 5 notes the software requirements for the solution lab test environment as configured.

Table 5 Software requirements

Software solution requirements	Version
IBM Spectrum Scale	v5.0+
IBM Elastic Storage™ Server[a]	v5.3.0+

a. IBM Elastic Storage Server is shown to illustrate storage compatibility with IBM Spectrum Scale and IBM Storage Enabler for Containers. However, please note that IBM Storage Enabler for Containers, Kubernetes, or IBM Cloud Private code cannot be installed directly upon the IBM Elastic Storage Server Elastic Management Server or IBM Elastic Storage Server I/O nodes. IBM Elastic Storage Server will be managed by the larger IBM Spectrum Scale cluster, and IBM Storage Enabler for Containers, Kubernetes, or IBM Cloud Private code will be installed on IBM Spectrum Scale nodes in the larger cluster.

IBM Storage Enabler for Containers

IBM Storage Enabler for Containers allows IBM storage systems to be used as persistent volumes for stateful applications running in IBM Cloud Private clusters. IBM Storage Enabler for Containers v2.1 extends IBM Spectrum Connect v3.6 for IBM block storage and IBM Spectrum Scale for file storage, respectively, to Kubernetes-orchestrated container environments. IBM Storage Enabler for Containers currently supports only using block storage or IBM Spectrum Scale for dynamic storage provisioning within a single cluster. Refer to IBM Storage Enabler for Containers Release Notes for supported operating systems tables.

Multicloud solution reference architecture

The architecture for this multicloud solution contains the following elements:

- Software
  - IBM Cloud Private (version 3.1.2)
  - VMware vSphere 6.5
  - IBM Spectrum Connect 3.6.1
  - IBM Storage Enabler for Containers 2.1
- Hardware
  - VersaStack Solution infrastructure
- Network
  - 16 Gbps Fibre Channel
  - 40 GB Ethernet

This Blueprint uses the end-to-end private cloud solution architecture illustrated in Figure 1.

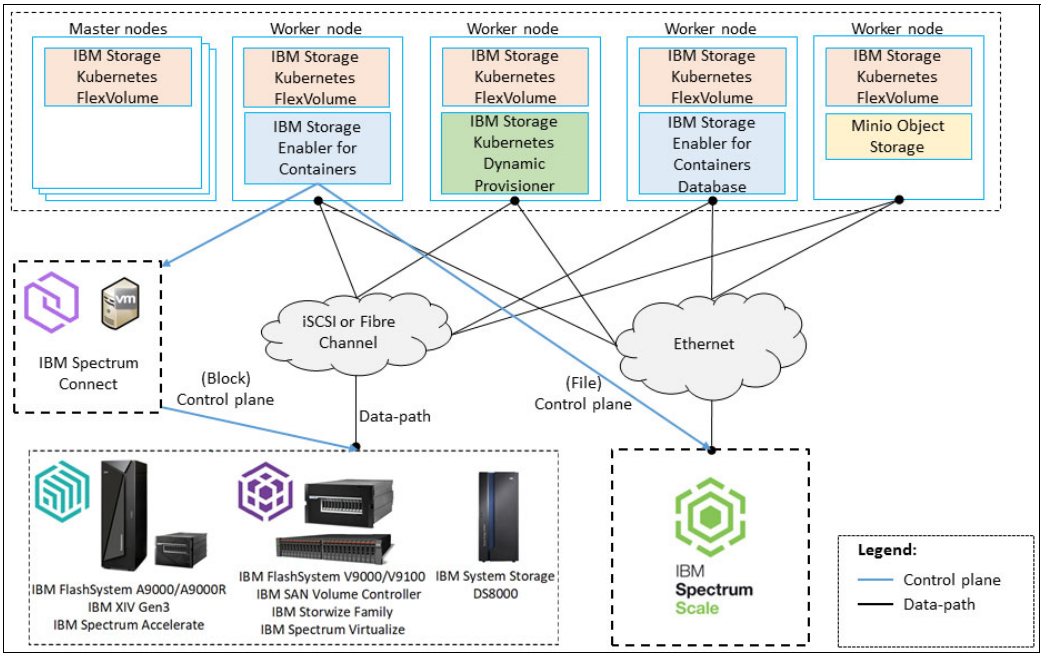


Figure 1 End-to-end private cloud solution

In this test environment, VMware vSphere 6.5 is installed on Cisco UCS 5108 blade servers. The Cisco UCS blade servers are connected over 40 Gb Ethernet to Cisco UCS 6248 series fabric interconnect switch modules. The IBM Storwize V5020 storage controller is connected to the network by 1 Gbps Ethernet. There is also an IBM Storwize V7000F system connected to Cisco Nexus 93180YC with 10 GbE to provide iSCSI volumes for the persistent volumes. All the switch modules are configured with a cluster link to ensure maximum availability.

Solution architecture control and data paths

The control and data paths of the solution architecture described in this Blueprint are shown in Figure 2.

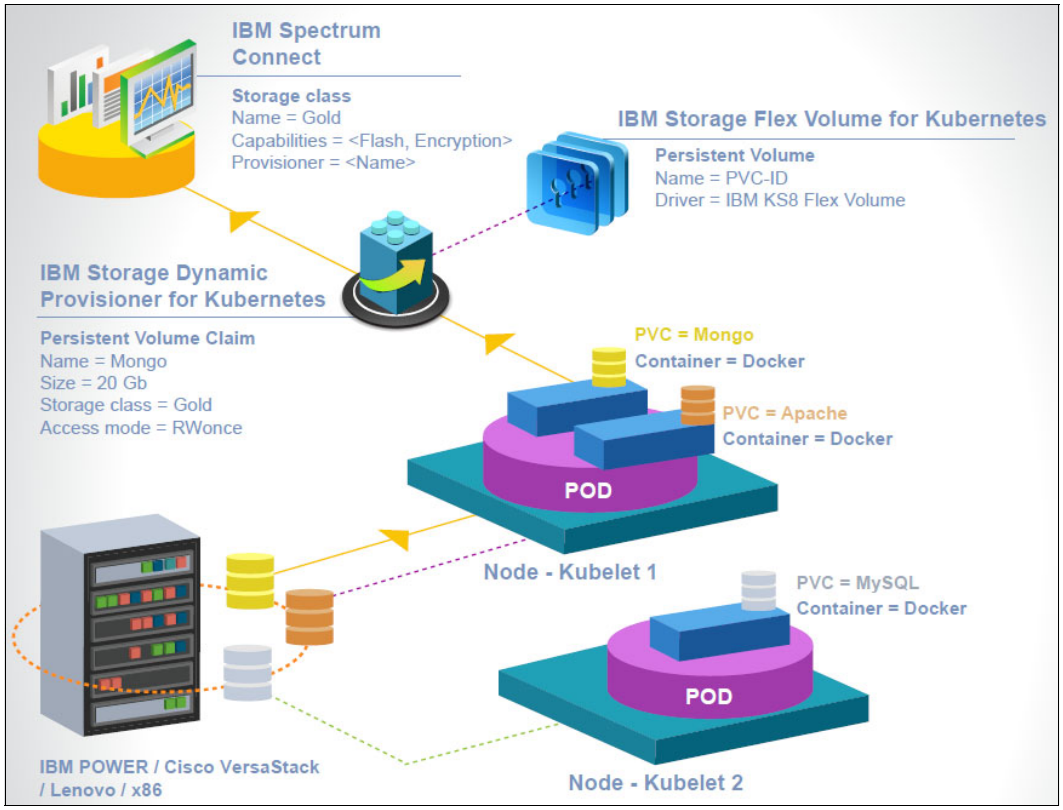


Figure 2 The control and data paths

In Figure 2, three storage classes such as ibmc-block-gold, ibmc-block-silver and ibmc-block-bronze types of volumes are created in VersaStack. These storage volumes have established connection with IBM Cloud Private worker nodes using iSCSI configurations. IBM Storage Enabler for Containers dynamic provisioner (IBM Storage Kubernetes Dynamic Provisioner) creates a persistent volume (PV) based on a persistent volume claim (PVC), in this case a MongoDB container as shown in Figure 2 with correct storage class definitions provided in IBM Spectrum Connect with the correct type of storage class volume. The IBM Storage Enabler for Containers flex volume driver then attaches or detaches the persistent volume with the MongoDB container.

Note: IBM Storage Enabler for Containers enables a stateful container on IBM Storage by provisioning the Kubernetes flex volume driver and dynamic provisioner. It integrates the Kubernetes flex volume and dynamic provisioner APIs.

In the solution proof of concept (POC) test setup, two VMs are created and configured with Red Hat Enterprise Linux 7 (64-bit) to install IBM Spectrum Connect version 3.6 in a high-availability configuration.

Five VMs are created and configured with Red Hat Enterprise Linux 7 (64-bit) to install IBM Cloud Private configured with three master and two worker nodes.

For more details, refer to the Hardware requirements and recommendations for IBM Cloud Private 3.1.2 topic in the IBM Knowledge Center at:

https://www.ibm.com/support/knowledgecenter/SSBS6K_3.1.2/kc_welcome_containers.html

Table 6 provides the VMs configured for the solution lab test environment.

Table 6 Configured VMs

Solution module	Number of VMware vSphere VMs	Operating system
IBM Spectrum Connect	2	Red Hat Enterprise Linux 7 (64-bit)
IBM Cloud Private	5	Red Hat Enterprise Linux 7 (64-bit)

IBM Cloud Private 3.1.2

IBM Cloud Private is an application platform for developing and managing on-premises, containerized applications. It is an integrated environment for managing containers that includes the Kubernetes container orchestrator, a private image repository, a management console and monitoring frameworks.

IBM Cloud Private also includes a graphical user interface (GUI), which provides a centralized location from which enterprises can deploy, manage, monitor and scale applications.

IBM Cloud Private is available in a community edition as IBM Cloud Private-CE, which provides a no-charge, limited offering that is ideal for test environments. IBM Cloud Private is available for purchase with other IBM products, including IBM middleware and other software products. IBM Cloud Private bundles include the Cloud Native (PN: D1URTLL & D1US3LL) and Enterprise (PN: D1VXCLL & D1VXSLL) editions that contain the core IBM Cloud Private platform and featured applications (available at no charge) that you can access through the catalog.

Each bundle also contains different entitled software that you can install separately or add to the catalog after you install the IBM Cloud Private platform. IBM Cloud Private can be purchased via IBM Passport Advantage® or an IBM Business Partner.

For more information about IBM Cloud Private, refer to the IBM Knowledge Center at:

https://www.ibm.com/support/knowledgecenter/SSBS6K_3.1.2/kc_welcome_containers.html

Installing IBM Cloud Private Enterprise 3.1.2 HA environment

This section provides installation instructions to set up high-availability (HA) IBM Cloud Private 3.1.2 on master, worker and proxy nodes in a cluster.

In the POC solution lab test environment, five RHEL 7 (64-bit) VMs are configured with the hardware specifications shown in Table 7 for master, proxy, worker and management roles for HA nodes.

Table 7 provides the system configuration for IBM Cloud Private.

Table 7 System configuration

Operating system	Processor	Memory and storage capacity
RHEL 7 (64-bit)	64-bit dual-core ?2.4 GHz	16 GB RAM; 200 GB disk space

For detailed hardware requirements and recommendations for IBM Cloud Private 3.1.2 server nodes, refer to the IBM Knowledge Center at:

https://www.ibm.com/support/knowledgecenter/SSBS6K_3.1.2/kc_welcome_containers.html

In the lab test environment described, three nodes are configured for master, proxy and management roles, respectively, and two nodes are configured as worker nodes.

Configuring cluster nodes

For detailed IBM Cloud Private 3.1.2 cluster node configuration instructions, refer to the IBM Knowledge Center at:

https://www.ibm.com/support/knowledgecenter/SSBS6K_3.1.2/kc_welcome_containers.html

Steps performed at the solution lab environment:

1. Configure the **/etc/hosts** file for each cluster node (as shown in Example 1).

Example 1 /etc/hosts file sample

```
127.0.0.1   localhost localhost.localdomain localhost4
localhost4.localdomain4
::1         localhost localhost.localdomain localhost6
localhost6.localdomain6

<master_node_1_IP_address> icp-master-1 #master1
<master_node_2_IP_address> icp-master-2 #master2
<master_node_3_IP_address> icp-master-3 #master3
<worker_node_1_IP_address> icp-worker-1 #worker1
<worker_node_2_IP_address> icp-worker-2 #worker2
```

2. Share the Secure Shell (SSH) keys among cluster nodes.
3. For each node in your cluster, confirm that a supported version of Python is installed. Python versions 2.6 to 2.9.x are supported.
4. Install Docker or configure your nodes for the automatic installation of Docker. IBM Cloud Private requires Docker.

   For detailed information about Docker installation, refer to the IBM Knowledge Center at:

   https://www.ibm.com/support/knowledgecenter/SSBS6K_3.1.2/installing/install_docker.html
5. On each RHEL node in your cluster, including the boot node, disable SELinux using the following command:

   setenforce 0

   Edit **/etc/sysconfig/selinux** and set **permissive** to **disabled**, then save the file
6. Optionally, on RHEL, disable the firewall with the following commands:

   systemctl disable firewalld

   systemctl stop firewalld
7. Download the installation files for IBM Cloud Private. You must download the correct file or files for the type of nodes in your cluster. You can obtain these files from the Passport Advantage website at:

   https://www-01.ibm.com/software/passportadvantage/pao_customer.html
8. Extract the images and load them into Docker using the following command. Extracting the images may take a few minutes.

   For Linux on 64-bit x86 processors, run this command:

   tar xf ibm-cloud-private-x86_64-3.1.2.tar.gz -0 | sudo docker load

   For Linux on Power (ppc64le), run this command:

   tar xf ibm-cloud-private-ppc64le-3.1.2.tar.gz -0 | sudo docker load

9. Create an installation directory to store the IBM Cloud Private configuration files.

   ```
   mkdir /opt/ibm-cloud-private-3.1.2
   ```

10. Navigate to your installation directory.

    ```
    cd /opt/ibm-cloud-private-3.1.2
    ```

11. Extract the sample configuration file from the installer image.

    For Linux on 64-bit x86 processors, run this command:

    ```
    docker run -v $(pwd):/data -e LICENSE=accept
    ibmcom/icp-inception:3.1.2-amd64-ee cp -r cluster /data
    ```

    For Linux on 64-bit Power (ppc64le) processors, run this command:

    ```
    docker run -v $(pwd):/data -e LICENSE=accept
    ibmcom/icp-inception:3.1.2-ppc64le-ee cp -r cluster /data
    ```

12. A cluster directory is created inside your installation directory. For example, if your installation directory is **/opt/ibm-cloud-private-3.1.2**, the **/opt/ibm-cloud-private-3.1.2/cluster** folder is created.

    The cluster directory contains the following files:

    - **config.yaml**: The configuration settings that are used to install IBM Cloud Private to your cluster.
    - **hosts**: The definition of the nodes in your cluster.
    - **misc/storage_class**: A folder that contains the dynamic storage class definitions for your cluster.
    - **ssh_key**: A placeholder file for the SSH private key that is used to communicate with other nodes in the cluster.

13. Modify the **/opt/ibm-cloud-private-3.1.2/cluster/hosts** file as shown in Example 2.

    Example 2 Hosts file

    ```
    [master]
    <master_node_1_IP_address>
    <master_node_2_IP_address>
    <master_node_3_IP_address>
    [worker]
    <worker_node_1_IP_address>
    <worker_node_2_IP_address>
    [proxy]
    <proxy_node_1_IP_address>
    <proxy_node_2_IP_address>
    <proxy_node_3_IP_address>
    ```

Note: In the solution lab environment master, proxy and management nodes are configured in the same server nodes.

14. If you use SSH keys to secure your cluster, in the /opt/ibm-cloud-private-3.1.2/cluster folder, replace the ssh_key file with the private key file that is used to communicate with the other cluster nodes.

    ```
    cp ~/.ssh/id_rsa /opt/ibm-cloud-private-3.1.2/cluster/ssh_key
    ```

15. Move the image files for your cluster to the **/opt/ibm-cloud-private-3.1.2/cluster/images** folder.

    For Linux on 64-bit x86 processors, run this command:

```
mkdir -p cluster/images; \
mv <path_to_installation_file>/ibm-cloud-private-x86_64-3.1.2.tar.gz
cluster/images/
```

For Linux on Power (ppc64le), run this command:

```
mkdir -p cluster/images; \
mv <path_to_installation_file>/ibm-cloud-private-ppc64le-3.1.2.tar.gz
cluster/images/
```

16. Customize the **config.yaml** file.

    a. Navigate to the cluster folder in your installation directory.

    ```
    cd cluster
    ```

    For more information, refer to the IBM Knowledge Center at:

    https://www.ibm.com/support/knowledgecenter/SSBS6K_3.1.2/installing/config_yaml.html

    b. Set the *kubelet_nodename* variable to use hostname instead of the default IP address (if required):

    ```
    kubelet_nodename: hostname
    ```

IMPORTANT: IBM Storage Enabler for Containers requires this configuration step when connected to IBM Spectrum Virtualize products.

    c. Refer to the important network configuration settings in the **config.yaml** file shown in Example 3.

Example 3 config.yaml cluster/proxy settings

```
## High Availability Settings for master nodes
vip_iface: <interface>
cluster_vip: <cluster_access_ip>

## High Availability Settings for Proxy nodes
proxy_vip_iface: <interface>
proxy_vip: <proxy_access_ip>
```

For addition details, refer to the IBM Knowledge Center at:

https://www.ibm.com/support/knowledgecenter/SSBS6K_3.1.2/installing/config_yaml.html#network_setting

17. Deploy IBM Cloud Private 3.1.2 from the cluster folder installation directory:

For Linux on 64-bit x86 processors, run this command:

```
sudo docker run --net=host -t -e LICENSE=accept \
-v "$(pwd)":/installer/cluster ibmcom/icp-inception-amd64:3.1.2-ee install
```

For Linux on 64-bit Power (ppc64le) processors, run this command:

```
sudo docker run --net=host -t -e LICENSE=accept \
-v "$(pwd)":/installer/cluster ibmcom/icp-inception-ppc64le:3.1.2-ee install
```

Access the IBM Cloud Private 3.1.2 cluster using the management console

Obtain the cluster management console URL and default credentials. The URL is *https://master_ip:8443*, where *master_ip* is the IP or cluster access VIP address of the master node of the IBM Cloud Private cluster. For reference, the IBM Cloud Private dashboard is shown in Figure 3.

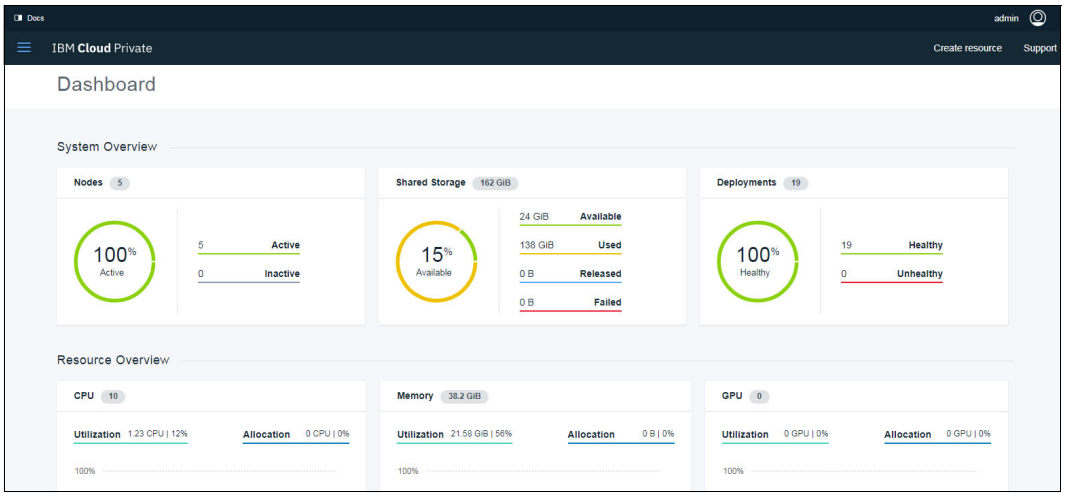


Figure 3 IBM Cloud Private dashboard

Performing first-time installation of Spectrum Connect 3.6.1

Table 8 provides the solution lab testing environment and the hardware specification of the VM on which IBM Spectrum Connect is installed.

Table 8 Solution lab testing environment and the hardware specification of the VM

Operating system	Processor	Memory and storage capacity
RHEL 7 (64-bit)	64-bit dual-core	4 GB RAM; 16 GB free disk space

IMPORTANT: IBM Spectrum Connect only runs on x86-based platforms on supported Linux operating systems.

For further details about the installation, refer to the IBM Spectrum Connect user guide for more information at:

https://www.ibm.com/support/knowledgecenter/en/SS6JWS/landing/IBM_Spectrum_Connect_welcome_page.html

Adding IBM Storage to IBM Spectrum Connect 3.6.1

Perform the following steps to add IBM Storage to IBM Spectrum Connect 3.6.1:

1. Enter the web address (URL) of the Linux host on which IBM Spectrum Control Base is installed. Use the following format:

   https://*[Spectrum Connect IP address]*:8440

Log in using the appropriate IBM Spectrum Connect credentials.

2. After successful login, the Storage Services and Storage Systems panes are displayed (as shown in Figure 4).

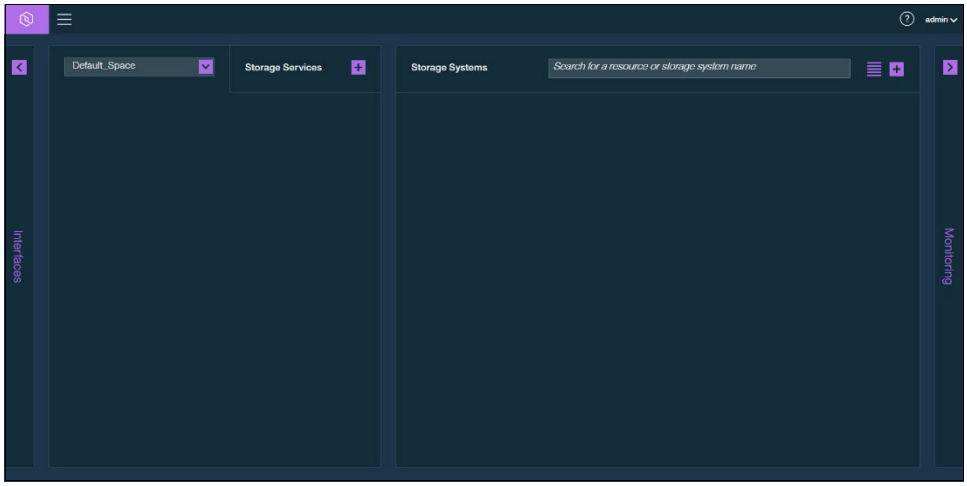


Figure 4 Spaces/Storage Services and Storage Systems panes

3. Click the **Settings** button and click **Storage credentials**. See Figure 5.

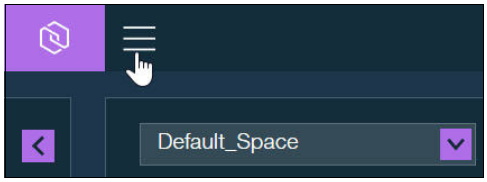


Figure 5 The Settings button

4. Enter the user name and password of the storage admin user who was defined on all your IBM storage systems. See Figure 6.

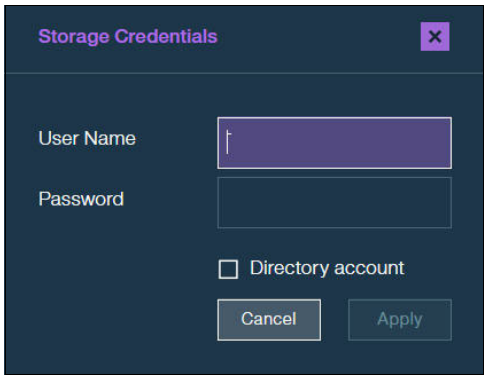


Figure 6 Current storage system user name (for all storage systems)

5. If the storage admin user account is defined on a directory server, select the **Directory account** check box. If the storage admin user account is locally defined on the storage system, clear the check box.
6. Click **Apply**.

7. Click the **Add a new IBM storage system** button (the “+” sign in the purple box) on the Storage Systems pane as shown in Figure 7.

Figure 7 Adding a new IBM storage system

8. Enter the management IP address or host name of the array.
9. Click **Add**. If the credentials are correct and the IP connection is established, the storage system is added to the Storage Systems pane, as shown in Figure 8.

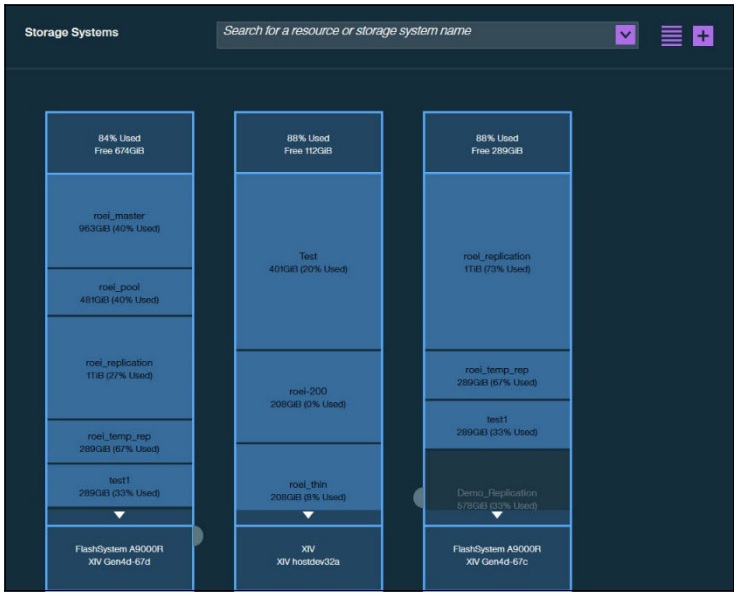


Figure 8 Storage Systems pane, bar view

Note: IBM Spectrum Connect fetches information about storage resources every 10 minutes by default. You can refresh the storage resource information immediately by right-clicking a system’s name, then clicking **Refresh**.

10. Click the **Table View** button to display the existing storage systems as a table.
11. Add the **Enabler for Containers** interface as shown in Figure 9.

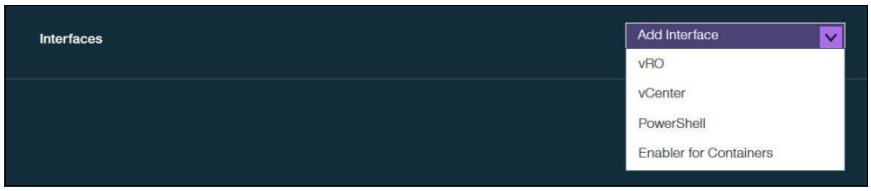


Figure 9 Selecting the Enabler for Containers interface

Log in with valid user credentials (as shown in Figure 10) and click **Apply**.

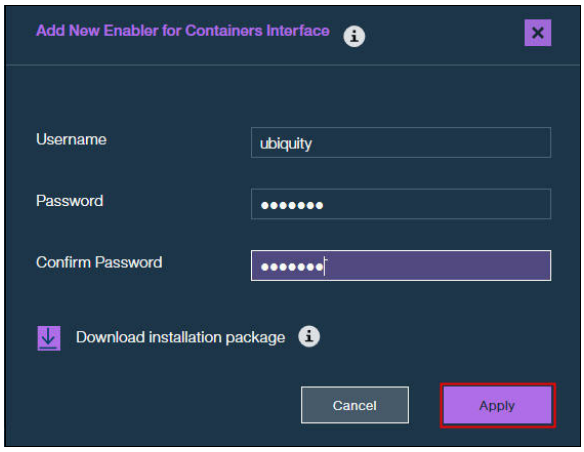


Figure 10 Add New Enabler for Containers Interface

12. Add a new storage service and provide its parameters, as shown in Figure 11, then click **Create**.

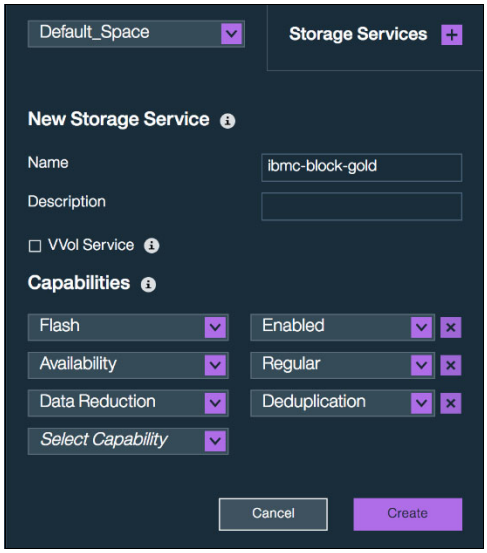


Figure 11 New storage service

Note: Storage services contain one or more physical storage pools. Storage services can represent distinct types or classes of storage pools depending on an organization's service level agreement (SLA): for example, ibmc-block-gold, ibmc-block-silver and ibmc-block-bronze representing solid-state drive (SSD), hard disk drive (HDD) and nearline SAS-based storage pools. In addition to specific storage pools' type and capacity, a storage service has a set of capabilities defining the storage quality, such as thin/thick provisioning, compression, encryption and so on.

13. Attach an appropriate storage resource with the required storage capacity from the storage pool (as shown in Figure 12) and click **Attach to ibmc-block-gold**.

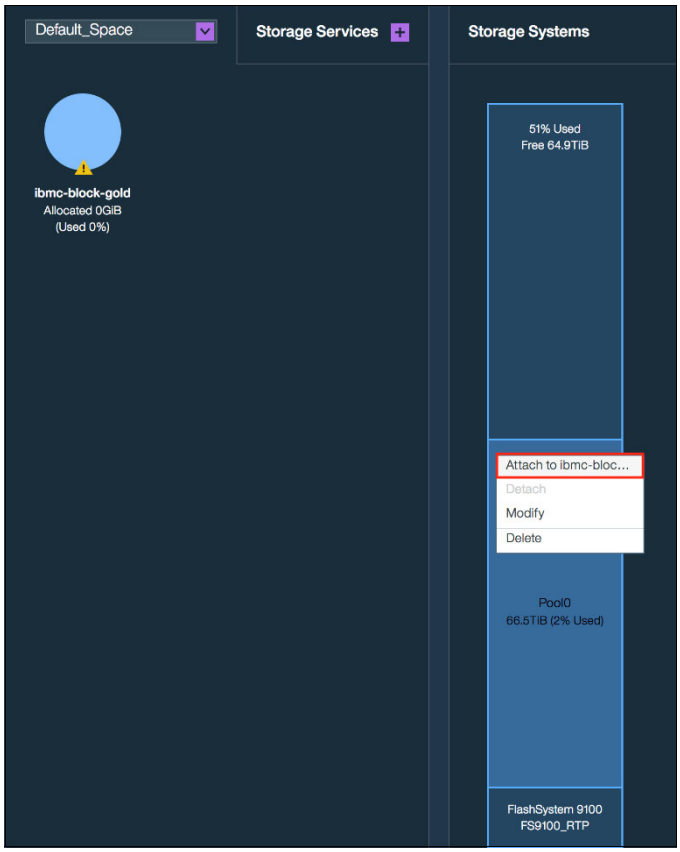


Figure 12 Attach storage pool

Note: The storage pools with the matching storage capabilities, as mentioned in step 13 on page 19, can be added as resources to a storage service.

14. Click **Delegate to ubiquity** to delegate a newly created storage service (as shown in Figure 13).

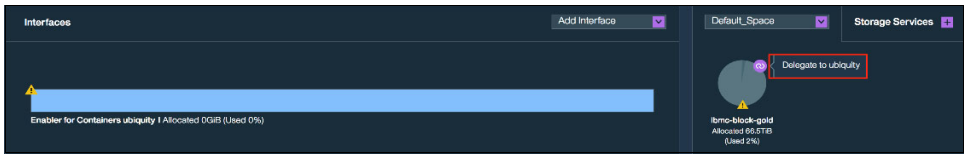


Figure 13 Delegate storage service

15. Notice that IBM Storage Enabler for Containers is successfully configured with IBM Spectrum Connect 3.6.1 (as shown in Figure 14 on page 20).

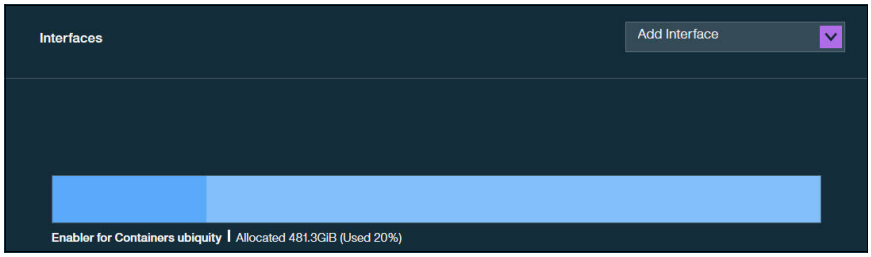


Figure 14 IBM Storage Enabler for Containers storage provision

Configuring iSCSI/Fibre Channel for IBM Cloud Private 3.1.2 worker nodes

IBM Cloud Private 3.1.2 worker nodes must be configured with the IBM storage system.

This section describes how to configure the storage system with the RHEL-based IBM Cloud Private 3.1.2 worker nodes.

The test team performed the following steps in the solution lab environment to install and configure iSCSI or Fibre Channel:

1. Install required packages.
   a. iSCSI configuration: Install **sg3_utils** utilities (which send SCSI commands), iSCSI initiator server daemon and the device mapper multipathing tool to configure multiple I/O paths between worker nodes and the storage array.

      ```
      yum install -y sg3_utils iscsi-initiator-utils device-mapper-multipath
      ```

   b. Fibre Channel configuration: Install **sg3_utils** utilities (which send SCSI commands) and the device mapper multipathing tool to configure multiple I/O paths between worker nodes and the storage array.

      ```
      yum install -y sg3_utils device-mapper-multipath
      ```

2. Multipath settings: The following settings as shown in Listing 4 are the preferred multipath settings for RHEL 7 and Storwize V7000. The **multipath.conf** file is copied at */etc/multipath.conf*.

   For IBM FlashSystem® A9000 systems, please reference the IBM Knowledge Center for details on the IBM Storage Host Attachment Kit:
   https://www.ibm.com/support/knowledgecenter/en/SSEPRF_2.9.0/UG/hak_ug_ch3_software_installation.html

   For IBM DS8880 systems, the default multipath configuration for the supported operating systems is sufficient. To create an */etc/multipath.conf* file with the default options, the following command can be run:

   ‘mpathconf –-enable’.

   More detailed information relating to your particular storage system can be found in the IBM Knowledge Center:

   https://www.ibm.com/support/knowledgecenter/en

   Example 4 shows */etc/multipath.conf*.

Example 4 /etc/multipath.conf

```
devices {
```

```
        device {
vendor “IBM”
product “2145”
path_grouping_policy “group_by_prio”
path_selector “round-robin 0”
prio “alua”
path_checker “tur”
failback “immediate”
no_path_retry 5
rr_weight uniform
rr_min_io_rq “1”
dev_loss_tmo 120
}
}
```

3. Configure and then start and verify the status of the multipath daemon service. Make sure that the multipath daemon service is in the "**`active (running)`**" state:

   `sudo modprobe dm-multipath`

   `systemctl start multipathd`

   `systemctl status multipathd`

4. If using iSCSI, update the iSCSI initiator name in the file "**`/etc/iscsi/initiatorname.iscsi`**" with worker node **`<hostname>`** inserted after **`InitiatorName=iqn.1994-05.com.redhat:<hostname>-<random generated number>`**.

   For example:

   `InitiatorName=iqn.1994-05.com.redhat:icp-worker1-74b436a728b6`

5. Add host definitions to the Storwize storage array by selecting hosts from the GUI console.

   IMPORTANT: The host definition name should exactly match the name of your Kubernetes node name as described in **`kubectl get nodes`**. In the solution lab environment, the host name is the same as the worker node name.

   Also, provide the iSCSI initiator name shown in the previous step in the file **`"/etc/iscsi/initiatorname.iscsi"`**. Click **Add** to add the host definition (as shown in Figure 15).

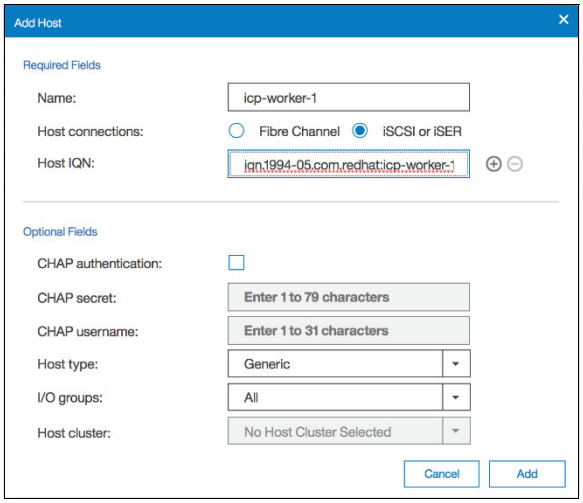


Figure 15 Add iSCSI host

For the iSCSI initd script startup, set a session to **automatic in /etc/iscsi/iscsid.conf: node.startup = automatic**

6. Discover the iSCSI target by using the **iscsiadm** CLI.

   ```
   iscsiadm -m discoverydb -t st -p <IP Address configured for iSCSI @ Storwize Storage Array>:3260 --discover
   ```

7. Log in to iSCSI target by using the **iscsiadm** CLI tool.

   ```
   iscsiadm -m node -p <IP Address configured for iSCSI @ Storwize Storage Array >:3260 –login
   ```

8. Verify the host using the Storwize GUI console (as shown in Figure 16).

Figure 16 iSCSI host status

Note: Host status will show as “degraded” for iSCSI hosts until a volume has been mapped to the host.

Configuring IBM Spectrum Scale for IBM Cloud Private 3.1.2. worker nodes

This section describes how to configure the storage system with the RHEL-based IBM Cloud Private 3.1.2 worker nodes.

First, set up your IBM Spectrum™ Scale cluster.

For more information, see “IBM Spectrum Scale cluster configurations” (https://www.ibm.com/support/knowledgecenter/STXKQY_5.0.0/com.ibm.spectrum.scale.v5r00.doc/bl1ins_lscfg.htm) and “Steps for establishing and starting your IBM Spectrum Scale cluster” (https://www.ibm.com/support/knowledgecenter/STXKQY_5.0.0/com.ibm.spectrum.scale.v5r00.doc/bl1ins_estart.htm).

You must add all IBM Cloud Private worker nodes as IBM Spectrum Scale client nodes.

For more information, see “Creating an IBM Spectrum Scale cluster” (https://www.ibm.com/support/knowledgecenter/STXKQY_5.0.0/com.ibm.spectrum.scale.v5r00.doc/bl1ins_clnodes.htm#clnodes).

Next, you must create a file system in your IBM Spectrum Scale cluster.

For more information about creating a file system, see “File system creation considerations” (https://www.ibm.com/support/knowledgecenter/STXKQY_5.0.0/com.ibm.spectrum.scale.v5r00.doc/bl1ins_plcrfs.htm).

For more information about the command to create a file system, see “mmcrfs command” (https://www.ibm.com/support/knowledgecenter/STXKQY_5.0.0/com.ibm.spectrum.scale.v5r00.doc/bl1adm_mmcrfs.htm).

Finally, mount the file system on all worker nodes in the IBM Cloud Private cluster.

For more information, see “Mounting a file system” (https://www.ibm.com/support/knowledgecenter/STXKQY_5.0.0/com.ibm.spectrum.scale.v5r00.doc/bl1adm_mount.htm).

Before using IBM Storage Enabler for Containers with IBM Spectrum Scale, please make note of the conditions described in “Appendix B. IBM Spectrum Scale usage restrictions”.

Installing IBM Storage Enabler for Containers

This section describes the details of installing IBM Storage Enabler for Containers using a helm chart. Perform the following steps to install IBM Storage Enabler for Containers:

1. Open the IBM Cloud Private Catalog, navigate to the ibm-storage-enabler-for-containers tile and select the tile.

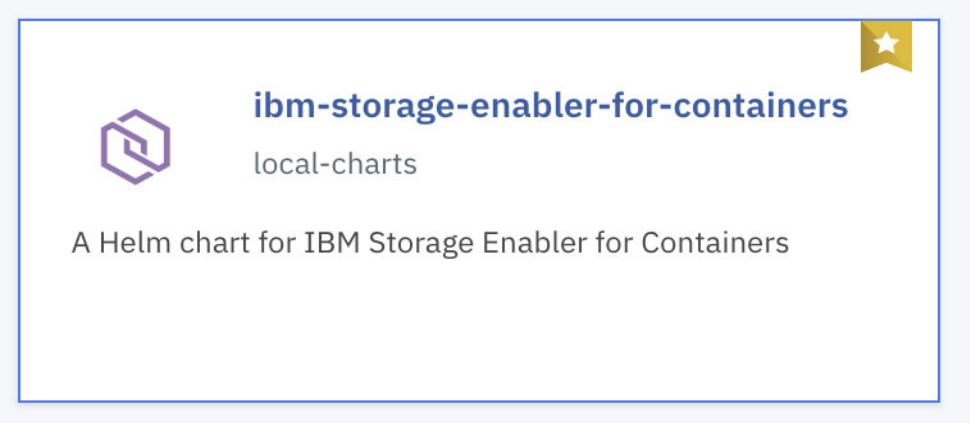


Figure 17 ibm-storage-enabler-for-containers tile in IBM Cloud Private catalog

2. Carefully read the README that is displayed in the helm chart for all Prerequisites and actions that need to be performed prior to installation of the chart.

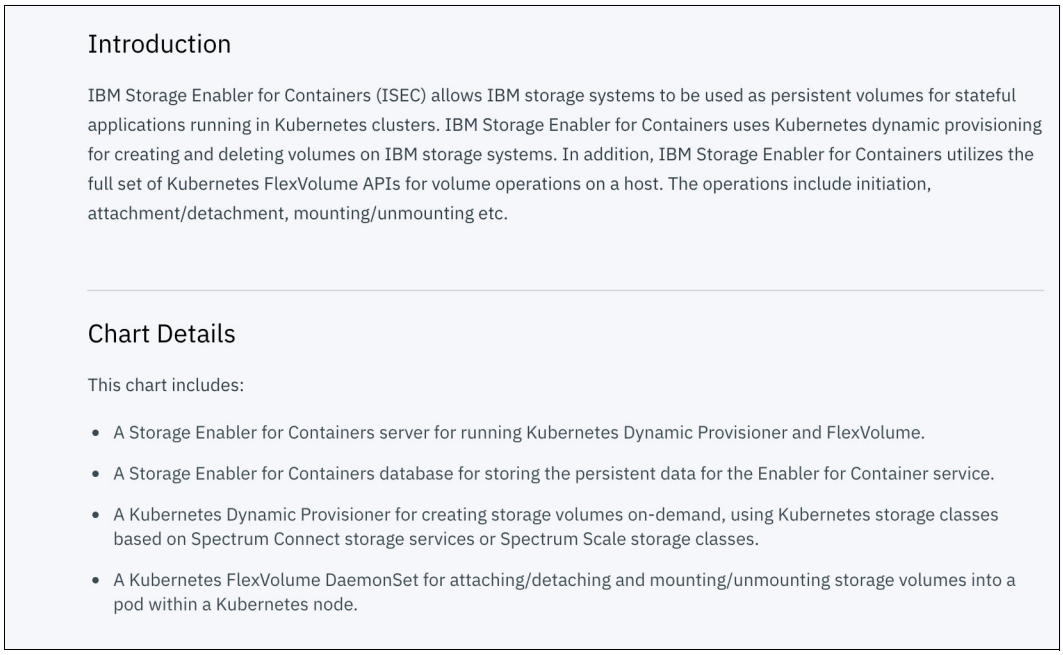


Figure 18 ibm-storage-enabler-for-containers README

3. Create a new Namespace that Storage Enabler for Containers will be installed in by navigating to **Menu** → **Manage** → **Namespaces** and select **Create Namespace**. Enter a Namepsace Name, ubiquity, and select "ibm-anyuid-hostpath-psp" as the Pod Security Policy.

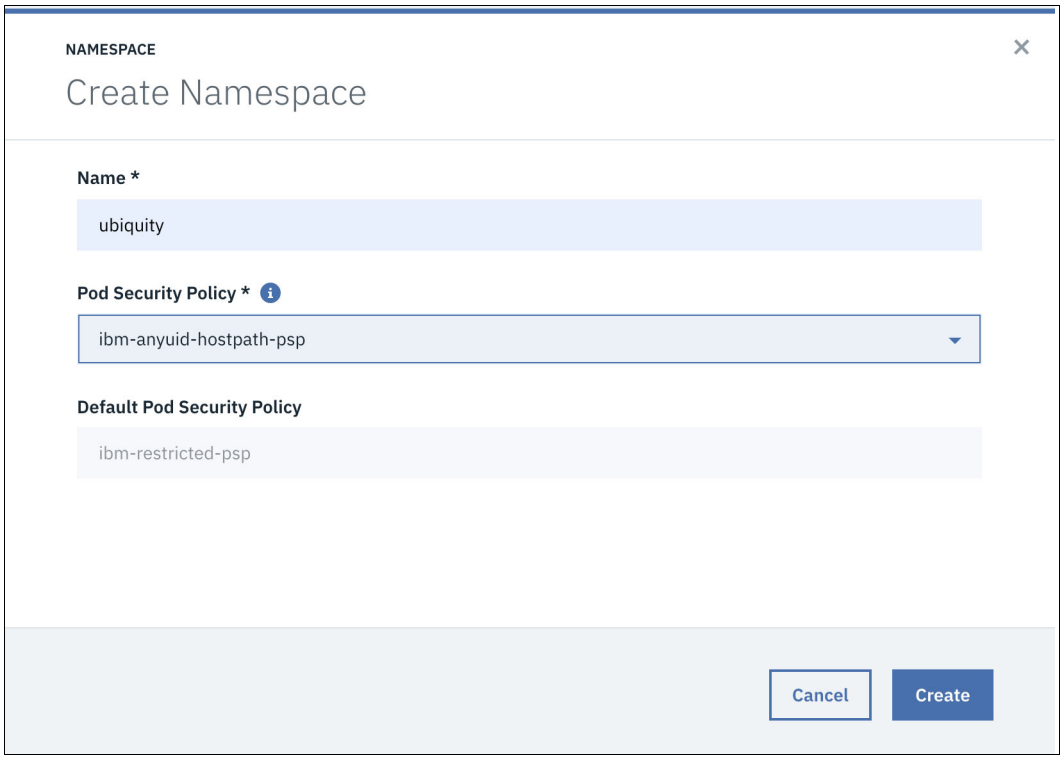


Figure 19 Create new Namespace for use by ibm-storage-enabler-for-containers helm chart

IMPORTANT: ibm-anyuid-hostpath-psp or equivalent PSP is required for Storage Enabler for Containers to install properly.
Refer to the ibm-storage-enabler-for-containers README if the use of a custom PSP is required.

4. Create a new Secret for use by Enabler for Containers DB by navigating to **Menu** → **Configuration** → **Secrets** and select **Create Secret.**

IMPORTANT: Data Values need to be encoded as base64 for entry into the IBM Cloud Private User Interface.

The output from base64 will be entered in Data, Values field in step c of the below example:

Command: **`echo -n ubiquity | base64`**

Output: **`dWJpcXVpdHk=`**

a. General information (refer to Figure 20):
   i. Name: `ubiquity-db-credentials`
   ii. Namespace: `ubiquity`
   iii. Type: `Opaque`

b. Data (refer to Figure 21 on page 26
   i. Name: `username`, Data: `dWJpcXVpdHk=`
   ii. Name: `password`, Data: `dWJpcXVpdHk=`
   iii. Name: `dbname`, Data: `dWJpcXVpdHk=`

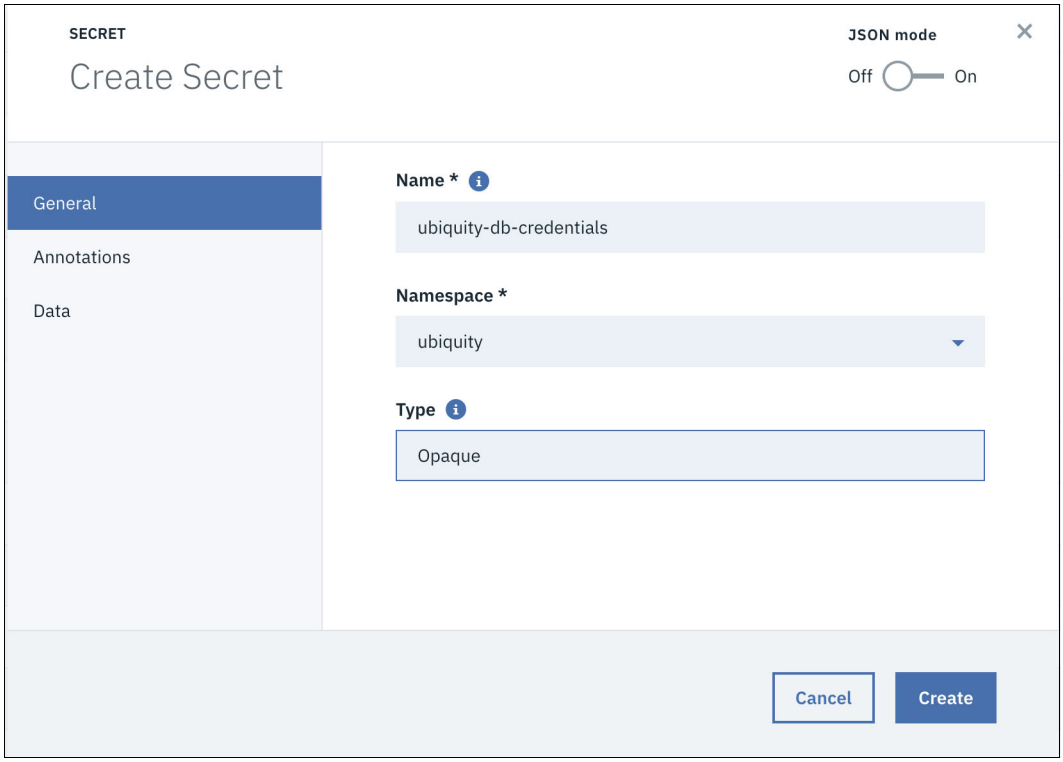


Figure 20 Enabler for Containers DB - Create secret, General Information

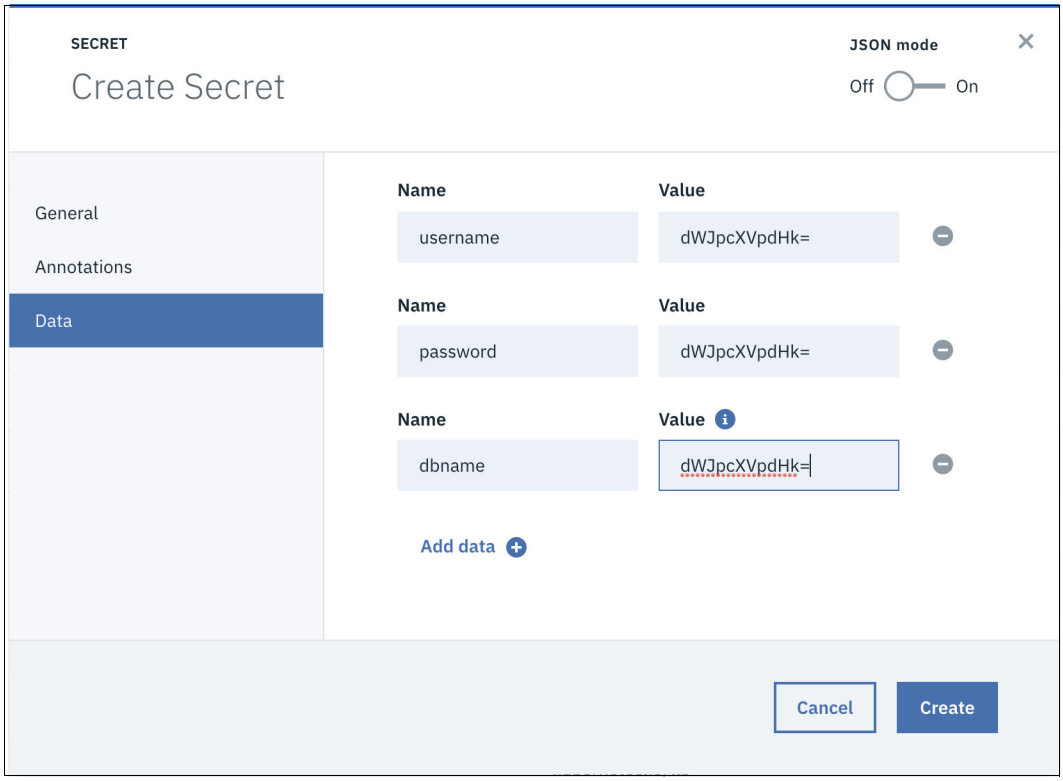


Figure 21 Enabler for Containers DB - Create secret, Data Information

5. Create new secret for Spectrum Connect or Spectrum Scale backends depending on the use of Block or File Storage, respectively.

IMPORTANT: Only one secret needs to be created for the correct backend configuration.

Spectrum Connect Secret configuration

a. General information (see Figure 22):

   i. Name: `scbe-credentials`
   ii. Namespace: `ubiquity`
   iii. Type: `Opaque`

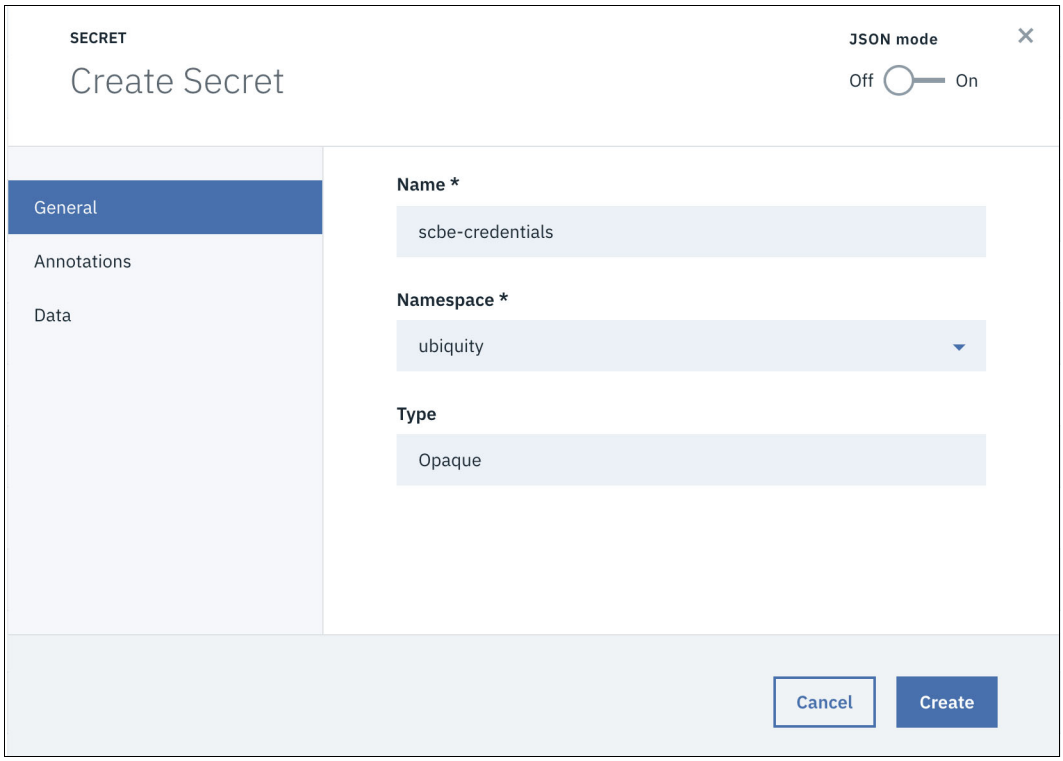


Figure 22 Spectrum Connect - Create namespace General Information

b. Data (see Figure 23):

   i. Name: `username`, Data: `dWJpcXVpdHk=`
   ii. Name: `password`, Data: `dWJpcXVpdHk=`
   iii. Name: `dbname`, Data: `dWJpcXVpdHk=`

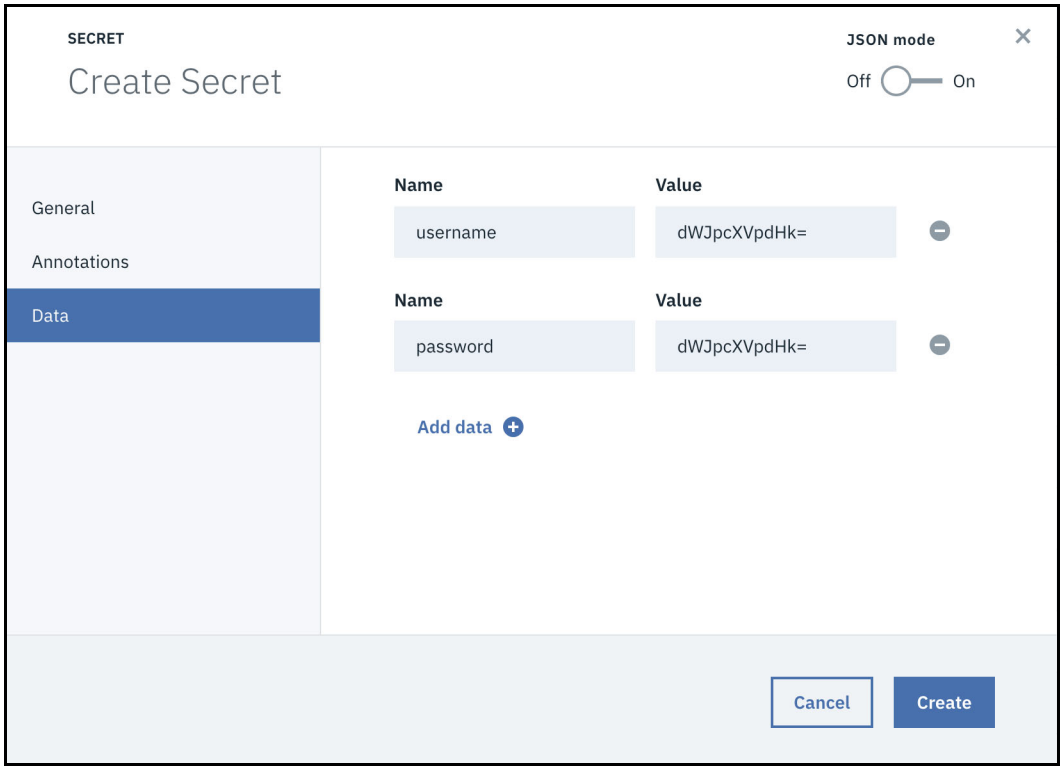


Figure 23 Spectrum Connect- Create namespace Data

Spectrum Scale Secret configuration

a. General information (see Figure 24):

   i. Name: `spectrumscale-credentials`
   ii. Namespace: `ubiquity`
   iii. Type: `Opaque`

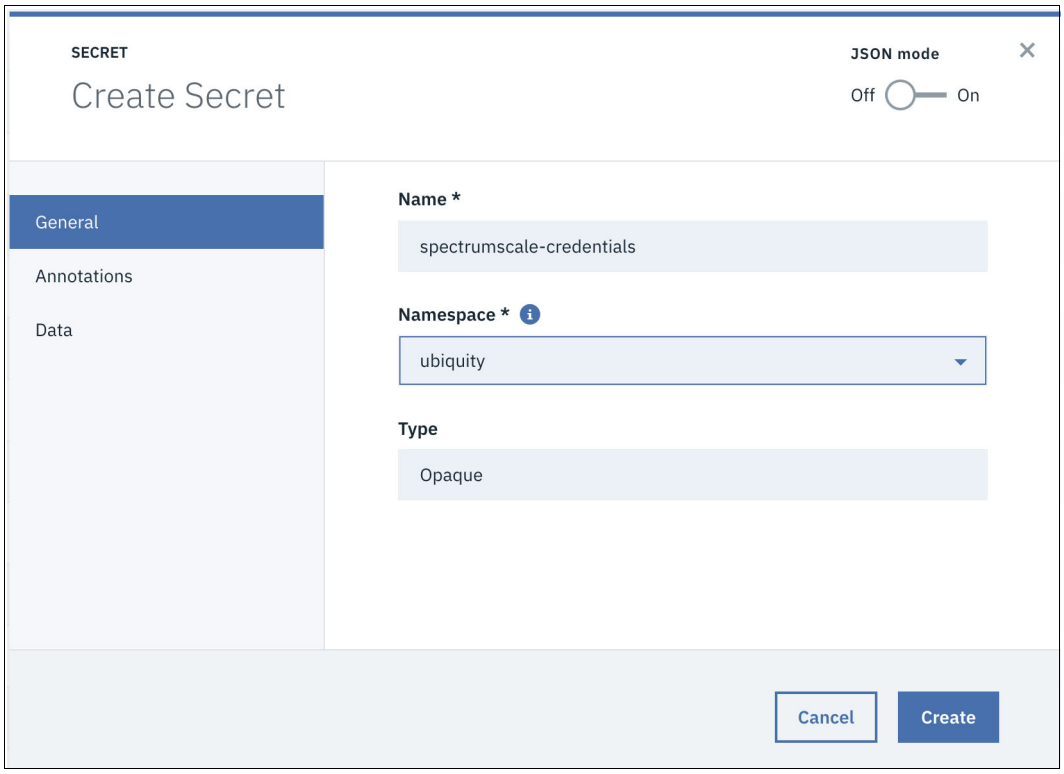


Figure 24 Spectrum Scale - Create secret, General Information

b. Data (see Figure 25):

   i. Name: `username`, Data: `dWJpcXVpdHk=`
   ii. Name: `password`, Data: `dWJpcXVpdHk=`
   iii. Name: `dbname`, Data: `dWJpcXVpdHk=`

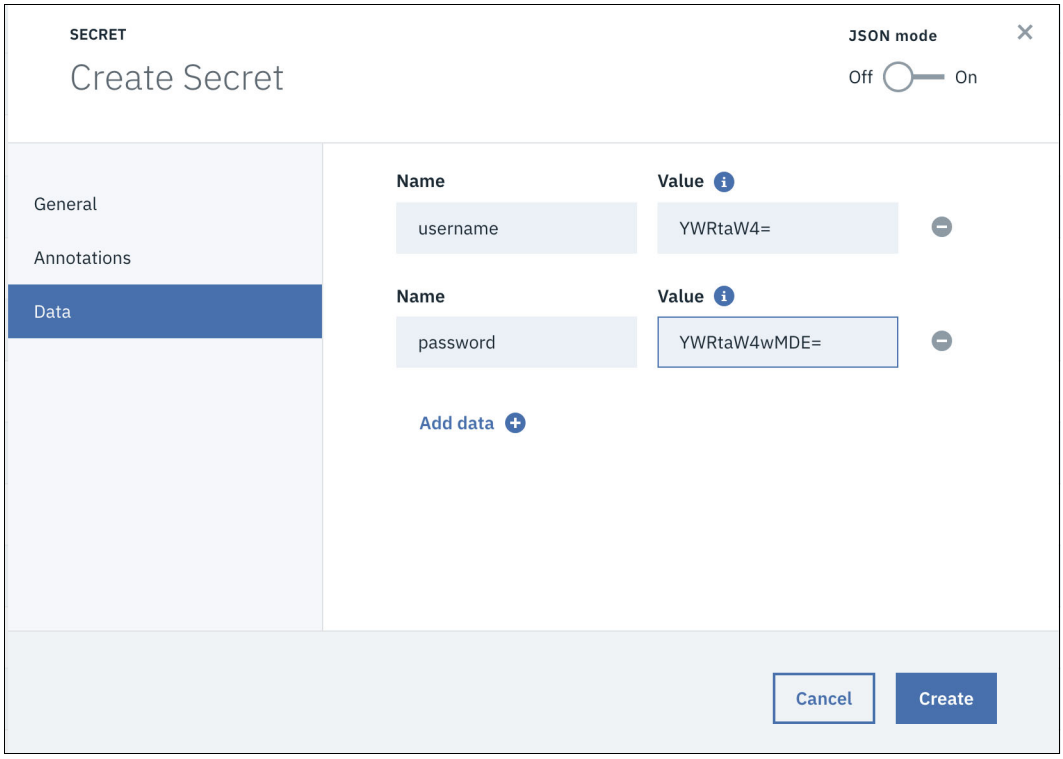


Figure 25 Spectrum Scale - Create secret, Data

6. Return to the ibm-storage-enabler-for-containers Helm Chart to complete the Configuration. Select Configure to continue.
7. Enter the release name, namespace and accept review of license agreements.

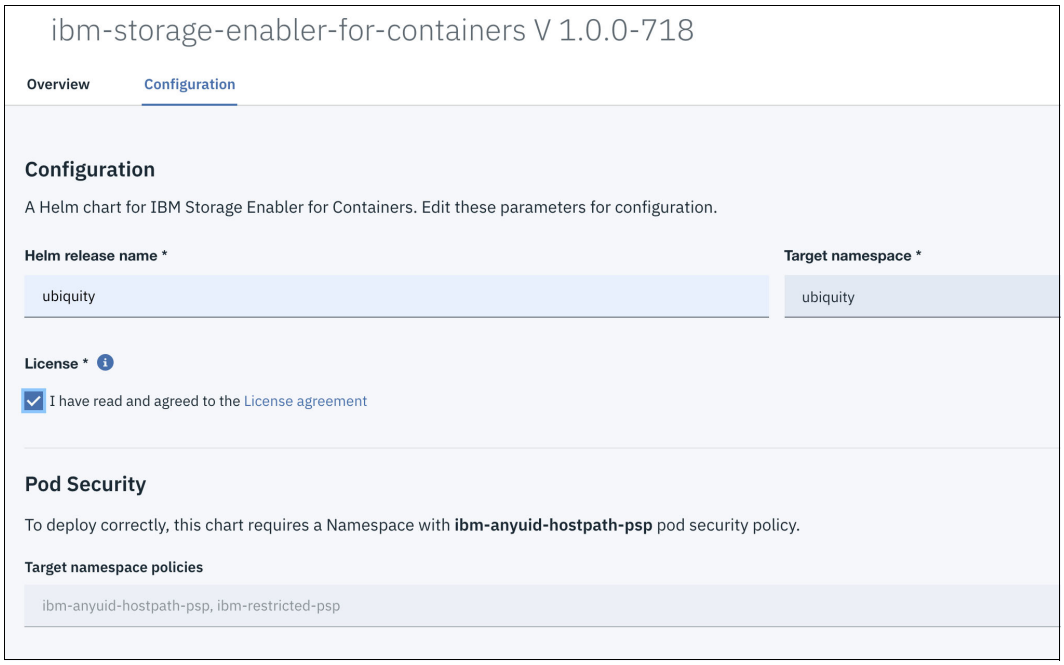


Figure 26 ibm-storage-enabler-for-beginning configuration

8. Select the correct Backend for the configuration you are running, Spectrum Connect (IBM Block Storage) or Spectrum Scale (IBM File Storage). The Backend determines which variables will be filled out in the Helm Chart. Follow Spectrum Connect configuration steps or Spectrum Scale configuration steps depending on required Backend configuration.

IMPORTANT: Values entered in images below are only to support solution architecture and your values will be different. Refer to the Storage Enabler for Containers Knowledge Center, README or syntactic feedback in the installer for additional information on each value.

Settings for Spectrum Connect

Enter the following as it applies to your environment:

a. IP or FQDN
b. Port
c. Secret for Spectrum Connect interface
d. Instance name
e. Default Storage Service
f. Default fstype of new volume
g. Default volume size (in GiB)
h. Storage service for Enabler for Containers DB storage class
i. fstype for Enabler for Containers DB storage class

Refer to Figure 27.

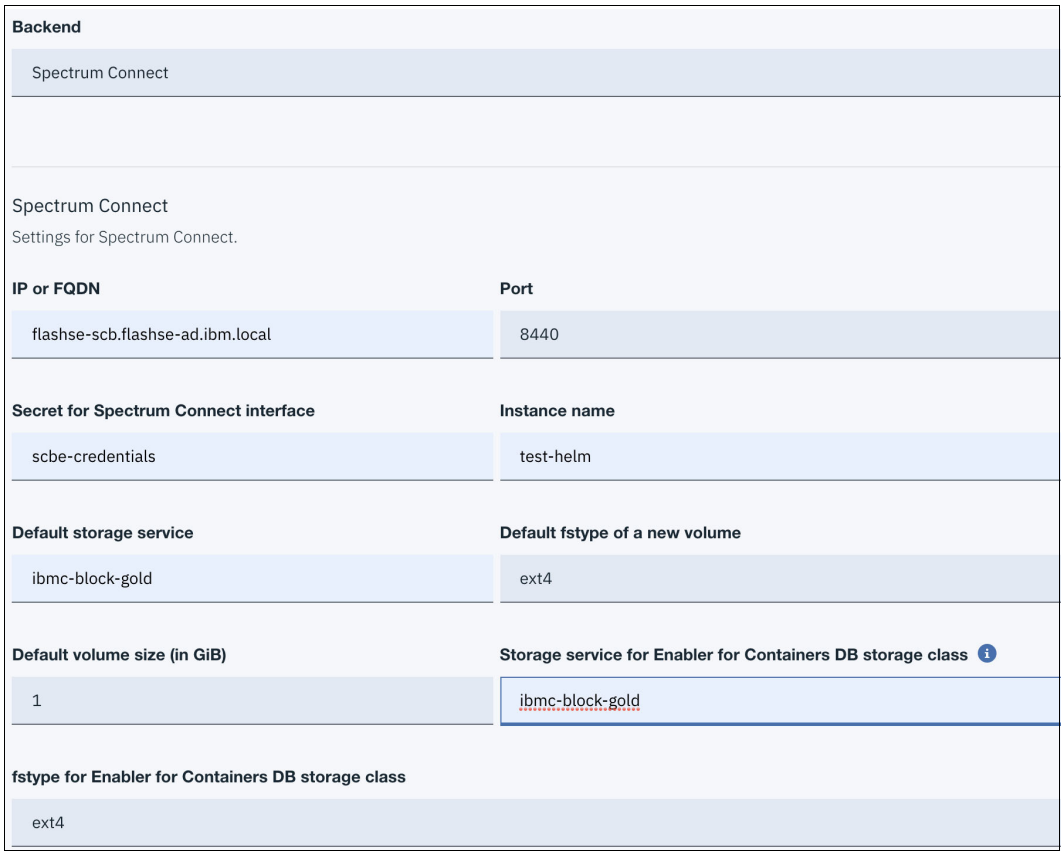


Figure 27 ibm-storage-enabler-for-containers Backend configuration

Spectrum Scale configuration steps

Enter the following as it applies to your environment:

a. IP or FQDN
b. Port
c. Secret for Spectrum Scale Management API (GUI) Server user credentials
d. Default filesystem

Refer to Figure 28.

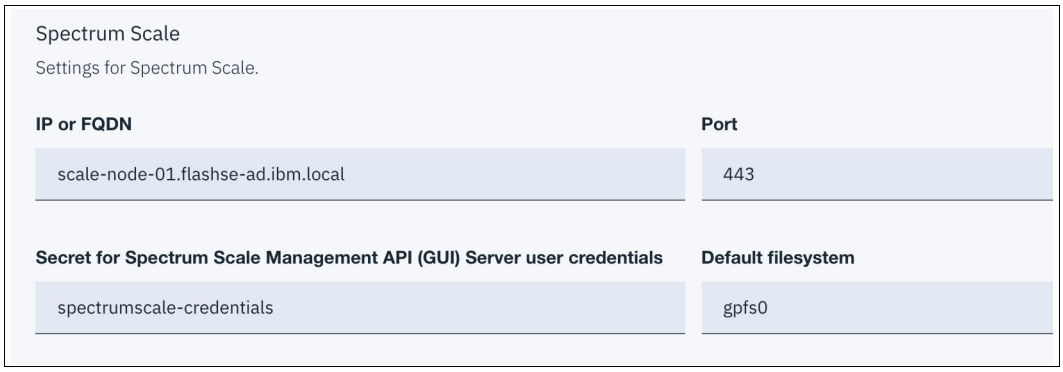


Figure 28 Spectrum Scale configuration steps

9. Enabler for Containers DB
   a. Secret for Enabler for Containers DB:
   b. Name of Enabler for Containers DB storage class: `ibmc-block-gold`
   c. Used as default storage class? Enable or Disable as needed

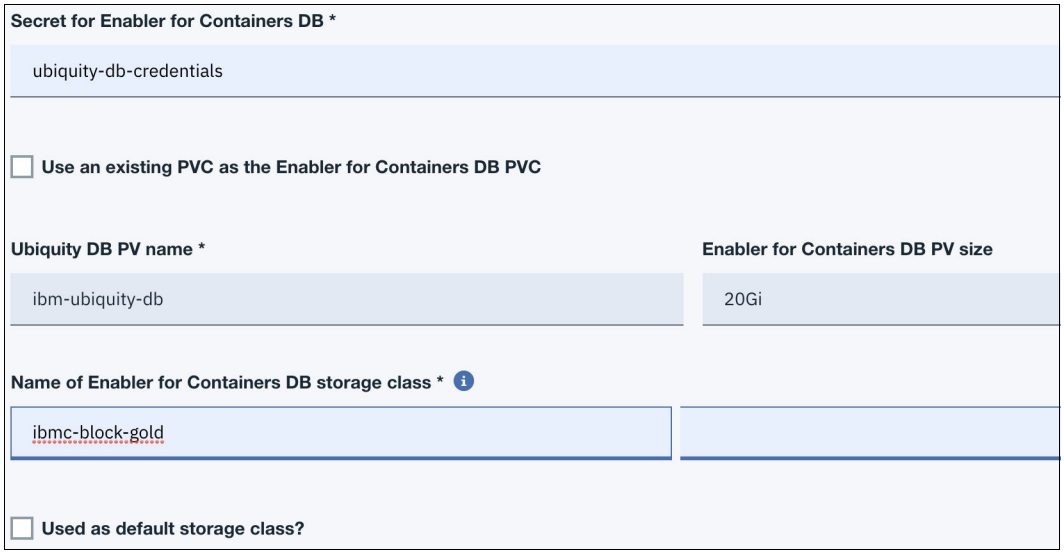


Figure 29 Enabler for Containers DB configuration steps

10. Default pod security policy

IMPORTANT: If this error (see Figure 30 on page 33) is shown after selecting a namespace and a custom Pod Security Policy is not being used, then this option would need to be Selected.

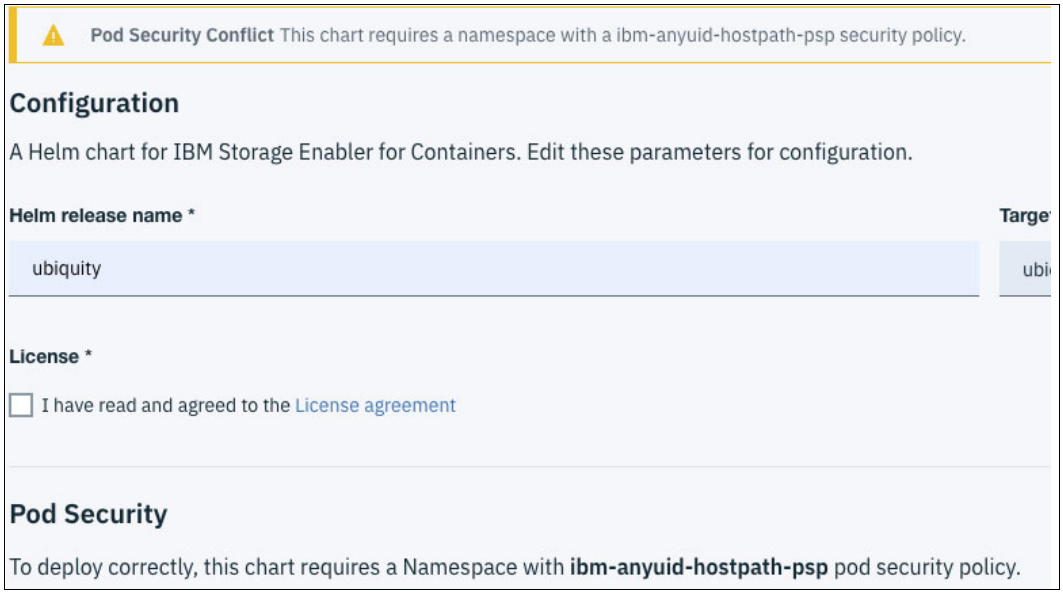


Figure 30 : Error message that shows changes required to Pod Security Policy

11. Global configuration

    Log level can be set to info (default), error or debug as required. If using SSL verification mode of verify-full, refer to the Storage Enabler for Containers Knowledge Center for configuration requirements.

Deploying a MongoDB instance using IBM Cloud Private and provisioning persistent storage to the MongoDB instance

This section provides detailed steps for deploying a MongoDB instance using the IBM Cloud Private catalog (Helm charts) and properly provisioning persistent storage to the MongoDB instance.

1. In the IBM Cloud Private 3.1.2 management web UI, select **catalog** from the menu bar and click **ibm-mongodb-dev** (as shown in Figure 31).

Figure 31 ibm-mongodb-dev

2. Click **Configure** and provide a proper release name in the Release name field, select the **Target namespace**, and tick the license agreement check box.
3. Continue in the same configuration web UI and, in the Data persistence configuration section, select **Use dynamic provisioning for persistent volume**. In the Data volume configuration section, provide the storage class name exactly as defined in one of the appropriate storageClass YAML files (as shown in Figure 32). For more information, refer to the "File storage class definitions" and "Block storage class definitions" sections.

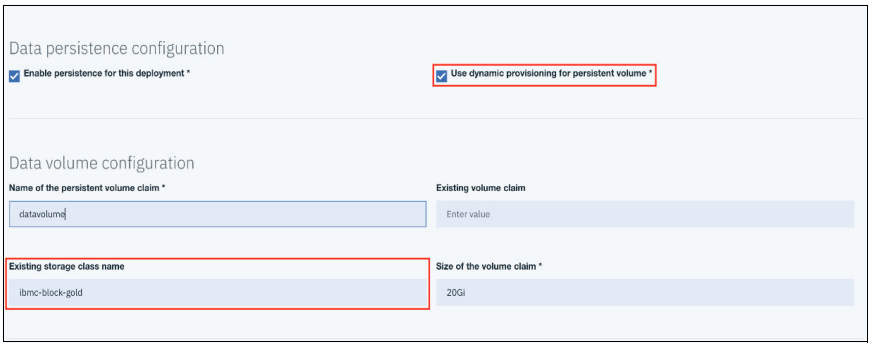


Figure 32 Configure data persistence for newly created MongoDB instance

Note: Provide the appropriate storage class based on the application needs.

4. In the Service configuration section of the same web UI, provide any additional MongoDB configuration parameters and click **Install**.

5. Validate the deployment by clicking **Workloads** → **Helm releases** in IBM Cloud Private (as shown in Figure 33).

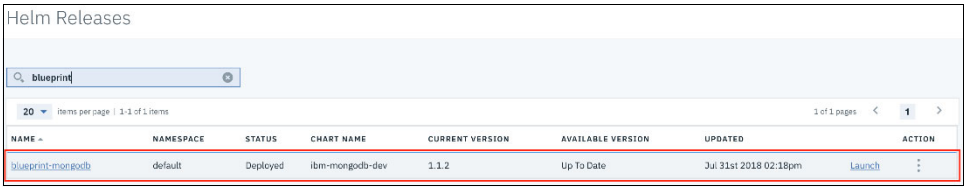


Figure 33 Helm releases

6. Verify **Persistent Volume Claim** by clicking the name of the MongoDB instance deployed and review the notes on how to access the MongoDB instance (as shown in Figure 34).

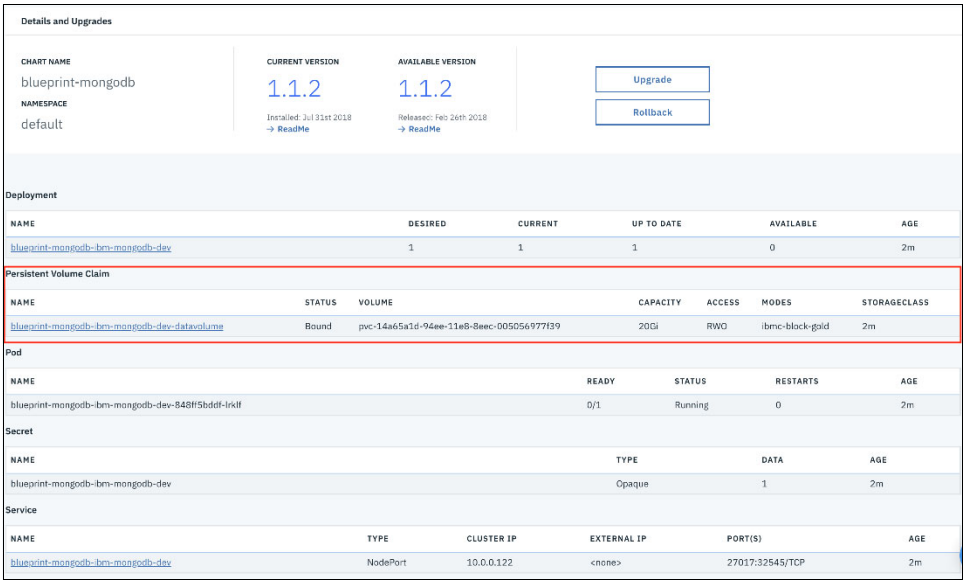


Figure 34 MongoDB instance

7. Verify the Storwize V7000F management web UI to validate the newly created persistent volume (as shown in Figure 35):

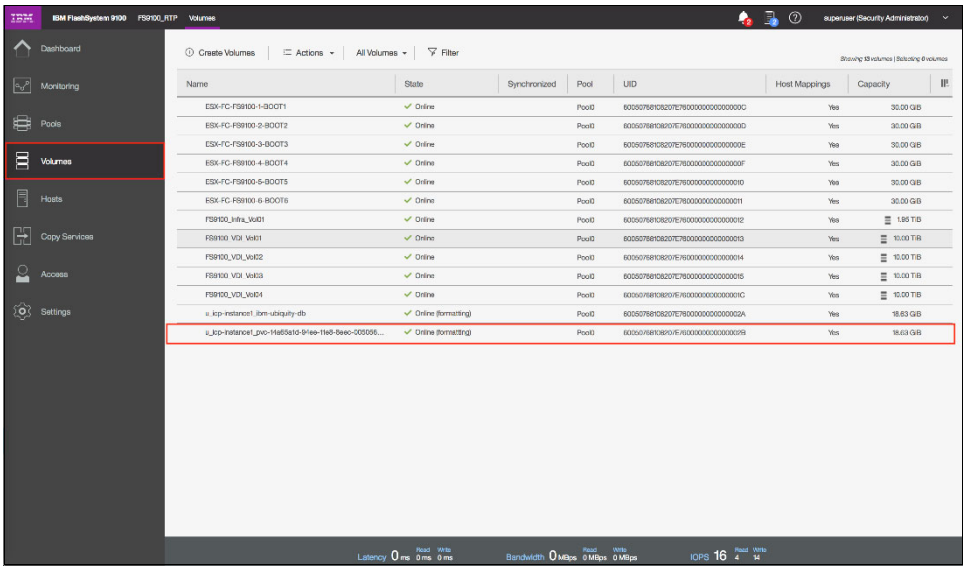


Figure 35 IBM Storwize V7000F management UI

Deploying Minio Object Storage using IBM Cloud Private

Minio is a high-performance distributed object storage server that can provide scalable object storage backed by IBM Storage, reference Table 4 on page 6 and Table 5 on page 7. Minio can run in a standalone or distributed mode.

IMPORTANT: Minio containers are scheduled as critical-pods and will be scheduled to start on the master nodes. Storage connectivity must be configured for the master nodes to service the Minio containers.

In order to deploy Minio in IBM Cloud Private, a secret object must be created that contains access and secret keys in base64 encoded form. For more information, see the ReadMe file included with the Helm chart. Use the following steps to create the secret object:

1. Encode accesskey and secretkey in base64 encoding:

```
echo -n "admin" | base64
YWRtaW4=

echo -n "admin1234" | base64
YWRtaW4xMjM0
```

2. Create the following secret object definition, after updating the <namespace>, as shown in Example 5:

Example 5 Create secret object definition

```
apiVersion: v1
kind: Secret
metadata:
  name: minio
```

```
  namespace: <namespace>
type: Opaque
data:
  accesskey: YWRtaW4=
  secretkey: YWRtaW4xMjM0
```

3. Run the following command to create the secret:

```
kubectl create -f secrets.yaml
```

Now it is possible to deploy Minio in IBM Cloud Private. To do so, use the following steps:

1. In the IBM Cloud Private Catalog UI, select **Catalog** from the menu bar and click **ibm-minio-objectstorage** (as shown in Figure 36).

Figure 36 The Minio object storage icon

2. Click **Configure** and provide a proper release name in the Release name field. Then select the **Target namespace** and tick the license agreement check box.
3. Select **Minio server mode** (standalone or distributed). Enter the **Access Secret** name you created manually.
4. Continue in the same configuration web UI and, in the Persistence configuration section, select **Enable Persistence** and **Use dynamic provisioning**. Provide the storage class name exactly as defined in one of the appropriate storageClass YAML files (as shown in Figure 37).

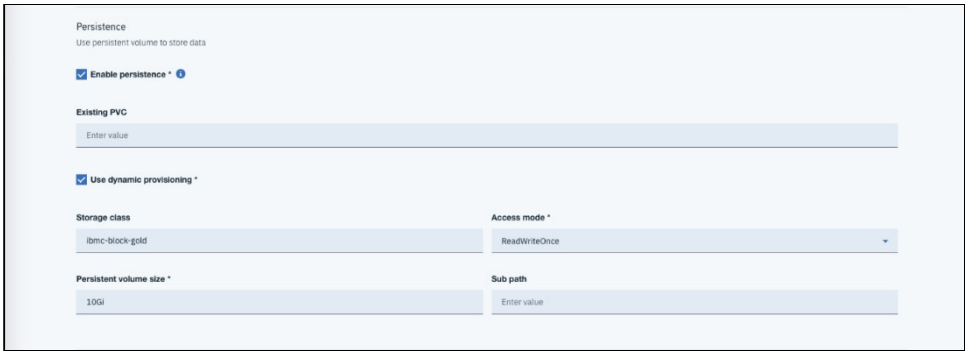


Figure 37 Minio storage properties

5. Provide any additional changes required to support your object storage and click **Install**.
6. Validate the deployment by clicking **Workloads** → **Helm** releases in IBM Cloud Private.

File storage class definitions

This section details how to configure Kubernetes storage classes.

First, define additional Kubernetes storage classes, if needed.

As the only storage class created during installation is used for the database, you might need additional storage classes for volume provisioning on IBM Spectrum Scale. A separate storage class must be created for each IBM Spectrum Scale filesystem to be used for creating persistent volumes. The template for setting storage classes is provided in the `./yamls/templates/storage-class-spectrumscale-template.yml` file as shown in Example 6.

Example 6 Storage class template

```
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
name: "<NAME>"
labels:
product: ibm-storage-enabler-for-containers
# annotations:
# storageclass.beta.kubernetes.io/is-default-class: "true"
#reclaimPolicy: "Retain" # Optional, Values: Delete[default] or
Retain
provisioner: "ubiquity/flex"
parameters:
backend: "spectrum-scale"
filesystem: "<filesystem name>"
type: "fileset"
# fileset-type: "<fileset type>" # Optional, Values:
Independent[default] or dependent
# uid: "<uid number>" # Optional
# gid: "<gid number>" # Optional
# inode-limit: "<no of inodes to be preallocated>" # Optional
```

You can configure the following parameters in the file, shown in Table 9:

Table 9 Configuration parameters in storage-class-template.yml

Parameter	Description
Name	Storage class name.
filesystem	IBM Spectrum Scale filesystem name for creating new volumes.
file- type	Optional parameter. Type of fileset to be created for volume. Permitted values: *independent* [default], *dependent*.

Uid	Optional parameter. Owner to be set on the fileset for newly created volume. User with specified uid/name must exist on IBM Spectrum Scale.
Gid	Optional parameter. Group owner to be set on the fileset for newly created volume. Must be specified along with uid. Group with specified gid/group must exist on IBM Spectrum Scale.
inode-limit	Optional parameter. Number of inodes to be pre-allocated for newly created fileset
isPreexisting	Optional parameter. Used to indicate whether to use existing fileset or create new fileset for volume. Permitted values: *false* [default], *true*. If *true* is specified, user must set **pv-name** parameter while creating PVC.
Type	Permanently set to *fileset*.
Product	Permanently set to *ibm-storage-enabler-forcontainers*.
provisioner	Permanently set to *ubiquity/flex*.
backend	Permanently set to *spectrum-scale*.

Block storage class definitions

You can define additional storage classes using the YAML configuration files. Refer to the sample configuration shown in Example 7.

Example 7 Configuration for storageclass-ibmc-block-gold.yaml

```
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: "ibmc-block-gold"
  labels:
    product: ibm-storage-enabler-for-containers

 annotations:
    storageclass.beta.kubernetes.io/is-default-class: "true"
provisioner: "ubiquity/flex"
parameters:
  profile: "ibmc-block-gold"
  fstype: "ext4"
  backend: "scbe"
```

Note: Example 7 shows the configuration for `storageclass-ibmc-block-gold.yaml`. To define silver or bronze storage classes, replace the `metadata: name and parameters: profile` for these profiles' yaml files with `ibmc-block-silver` and `ibmc-block-bronze`, respectively.

The `parameters: fstype` can be either `ext4` or `xfs`, depending on the required file system type.

Note: The following annotation, seen in Example 7, can be added to the configuration settings to make this storage class the default:

`storageclass.beta.kubernetes.io/is-default-class: "true"`

Also, `ubiquity_install.conf` uses `STORAGE_CLASS_NAME_VALUE` and

`STORAGE_CLASS_PROFILE_VALUE` parameters to configure the default storage class.

Use `kubectl` to create storage classes (as follows) from the IBM Cloud Private master node:

```
kubectl create -f storageclass-ibmc-block-gold.yaml
kubectl create -f storageclass-ibmc-block-silver.yaml
kubectl create -f storageclass-ibmc-block-bronze.yaml
```

Private cloud flexibility and data protection

As organizations adopt containers more broadly, data management and availability requirements have expanded. The IBM Spectrum Copy Data Management platform, in conjunction with IBM Spectrum Storage™ systems, enables critical use cases by providing in-place copy data management within existing infrastructure. The solution provides automated workflows that allow you to streamline the creation, management and use of copies of data, and to simplify copy management throughout the data lifecycle.

This section illustrates the creation of a persistent data volume, configuration of regular snapshots of that volume, and data restoration based on those snapshots, with an approach designed to be fast, flexible and familiar.

Configuration of IBM Cloud Private to support data protection

The IBM Cloud Private environment needs to be configured to support running kubectl and MongoDB client on all master nodes.

The included scripts use **`cloudctl`** to configure the kubectl environment prior to running the scripts. Update the required values to support your environment so the scripts will run successfully.

The version of MongoDB used at the time of release of this document is 3.6.0. Download and install the MongoDB client on all master nodes:

```
wget http://downloads.mongodb.org/linux/mongodb-linux-x86_64-rhel70-3.6.0.tgz
tar xzvf mongodb-linux-x86_64-rhel70-3.6.0.tgz
cp mongodb-linux-x86_64-rhel70-3.6.0/bin/mongo /usr/local/bin/
```

Installation and configuration of IBM Spectrum Copy Data Management

IBM Spectrum Copy Data Management can be deployed within a VMware infrastructure. On the vSphere server, specify the location of the appropriate `.ova` file and select the host and network to run the appliance.

IBM Spectrum Copy Data Management comes pre-packaged with all the required software, and once powered on, the console screen points to the portal link.

Proceed as follows:

1. Log in to the portal using a web browser: https://<hostname>:8443/portal/
2. Register the storage objects and Application Server. This is a one-time, agentless registration process. To register a new object, right-click the object to bring up an options menu (as shown in Figure 38):

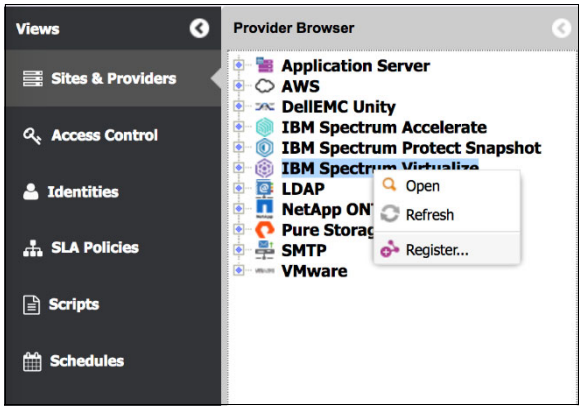


Figure 38 Registering storage provider and Application Server

Then click **Register** and fill in the details (as shown in Figure 39):

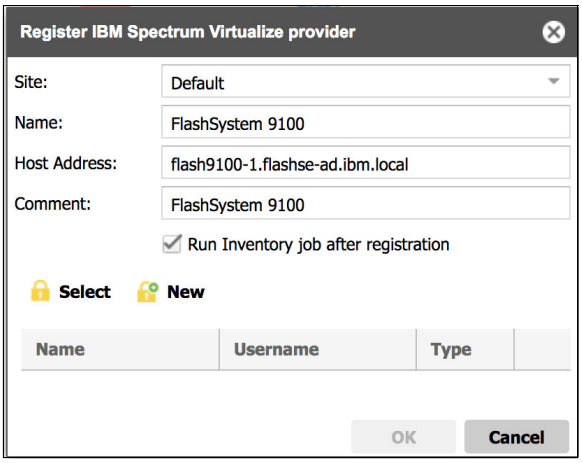


Figure 39 Registration of IBM Spectrum Virtualize™ provider

IBM Spectrum Copy Data Management uses the concept of sites to identify resource locations. The registration dialog box accepts the credentials and site selection.

3. Similarly, register IBM Cloud Private as an Application Server (File System) by selecting the server type (as shown in Figure 40):

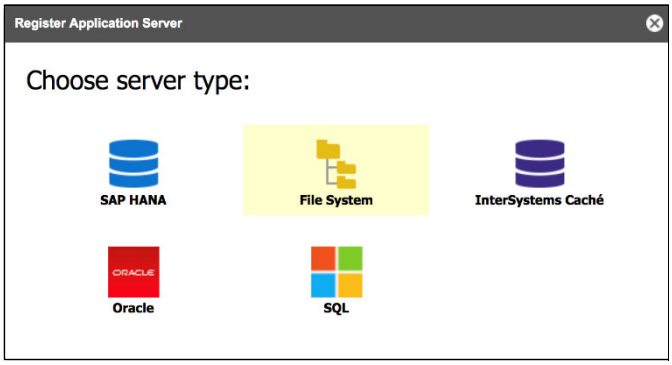


Figure 40 The Register Application Server screen

Then enter the registration details (as shown in Figure 41):

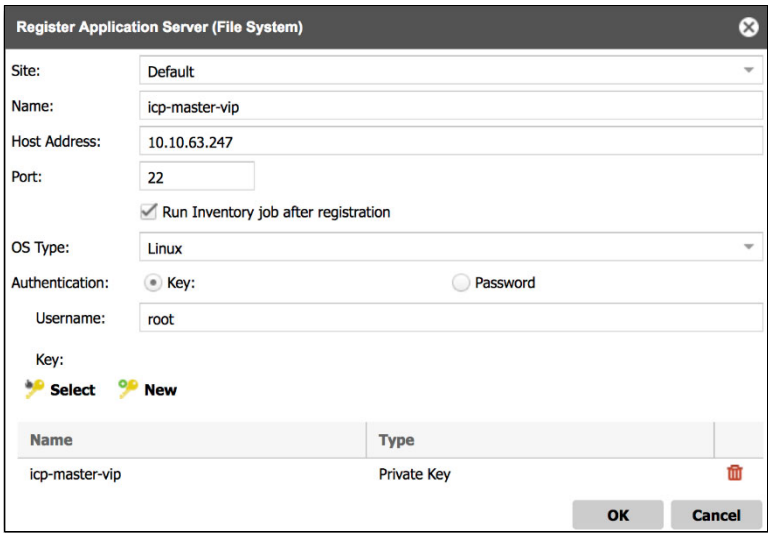


Figure 41 Registering a file system

4. Under Identities, provide the IBM Cloud Private user name and password or SSH keys.

After the resources are registered, IBM Spectrum Copy Data Management automatically creates a default catalog policy. This catalog policy can discover high-level objects, such as storage volumes and MDisk information in storage arrays.

IBM Spectrum Copy Data Management can run scripts before or after backup and restore jobs run, both at a job-level and before or after snapshots are captured. The example script included in "Appendix D: Sample scripts for MongoDB and Db2 database backup", uses **`db.fsyncLock()`** to force the MongoDB daemon (*mongod*) to flush all pending write operations to disk and lock the entire MongoDB instance to prevent additional writes until the user releases the lock with a corresponding **`db.fysncUnlock()`** command.

Upload the example script or your own (as shown in Figure 42) for use before and after snapshot creation to create crash-consistent snapshots:

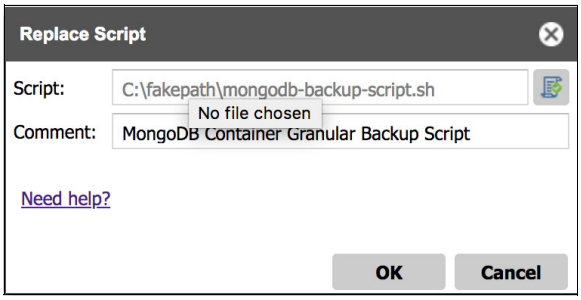


Figure 42 Replacing the MongoDB backup script

IBM Spectrum Copy Data Management policy creation

After registering the resources, the next step is to create a Service Level Agreement (SLA) policy, which will establish a storage workflow for Global Mirror and IBM FlashCopy®. Proceed as follows:

1. To create a policy, click on **Configure** → **SLA Policies** → **New SLA Policy** → **IBM Spectrum Virtualize**.
2. Click **Add FlashCopy**, and follow the storage workflow wizard (as shown in Figure 43):

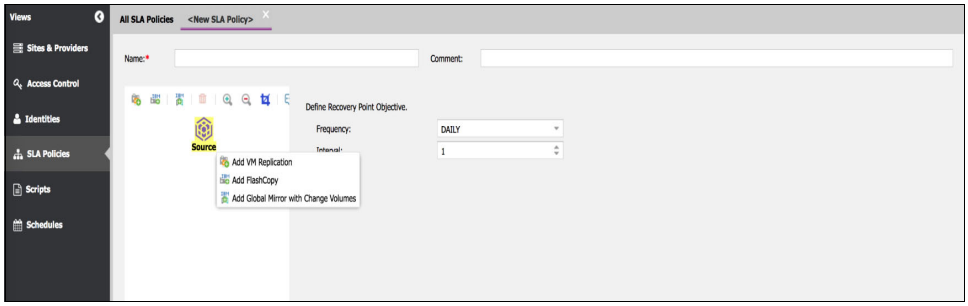


Figure 43 Configure FlashCopy

The SLA policy used in this Blueprint is detailed in Figure 44.

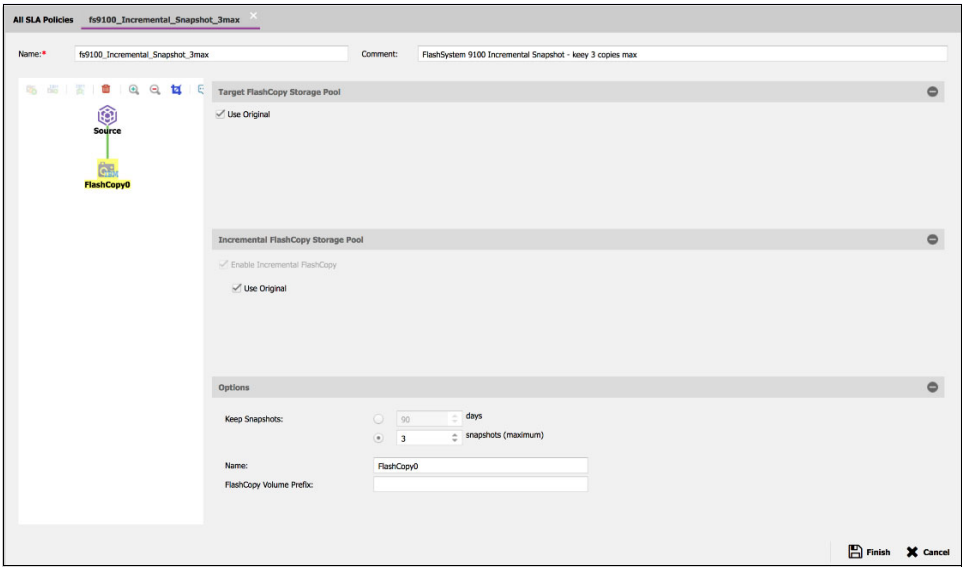


Figure 44 Solution lab setup SLA policy

Similarly, you can create a **Global Mirror with Change Volumes** SLA policy, if required.

IBM Spectrum Copy Data Management backup job creation

For the backup job creation, proceed as follows:

1. To create the backup policy for FlashCopy with IBM Spectrum Copy Data Management, click → **Jobs** → **Storage Controller** → **IBM Spectrum Virtualize** → **Backup** (as shown in Figure 45):

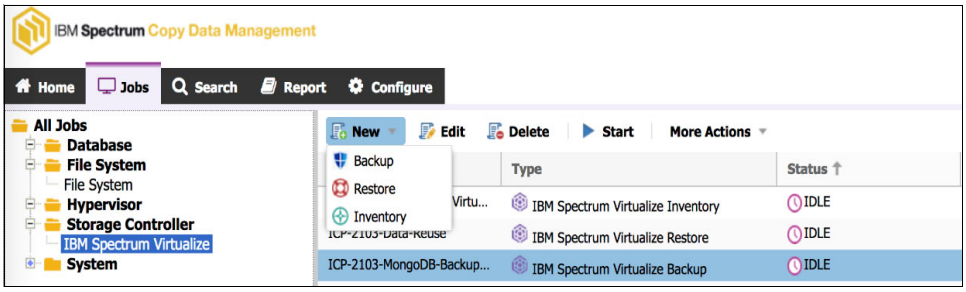


Figure 45 FlashCopy with IBM Spectrum Copy Data Management backup policy creation

The MongoDB instance that was deployed as part of the "Deploying a MongoDB instance using IBM Cloud Private and provisioning persistent storage to the MongoDB instance" section and shown in Figure 34 on page 35, is used in this example.

To determine the source volume, you can reference the PV ID associated with the PVC and prepend "**u_<InstanceID>**" to the PV ID. Alternatively, the following kubectl command will display its name:

```
kubectl get pv/$(kubectl get pvc/<PVC Name>  -o
jsonpath='{.spec.volumeName}') -o jsonpath='{.spec.flexVolume.options.Name}'
```

2. Follow the wizard, select the source volume, and associate the SLA policy created for FlashCopy with the backup job (as shown in Figure 46):

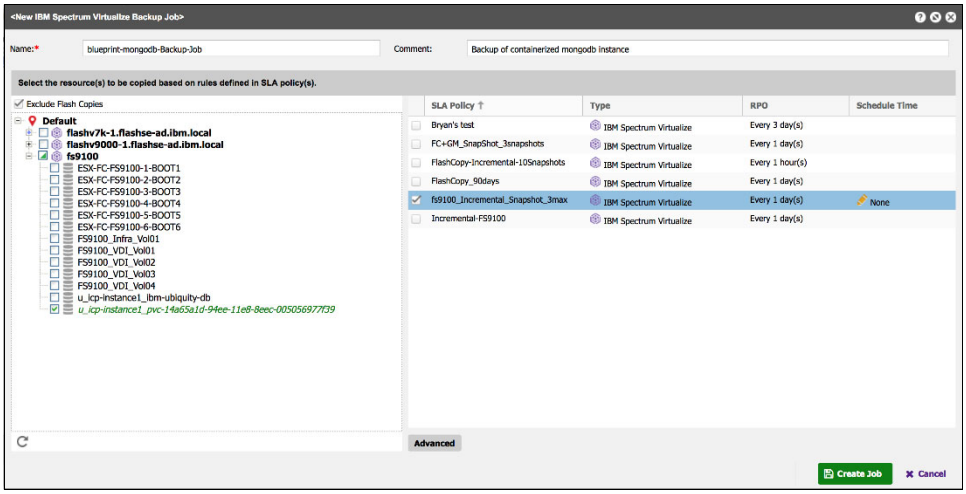


Figure 46 Select source volume and SLA policy

3. In Advanced Options, configure the job-level scripts for pre-script and post-script to lock and unlock the database using the provided example script, as shown in Figure 47.

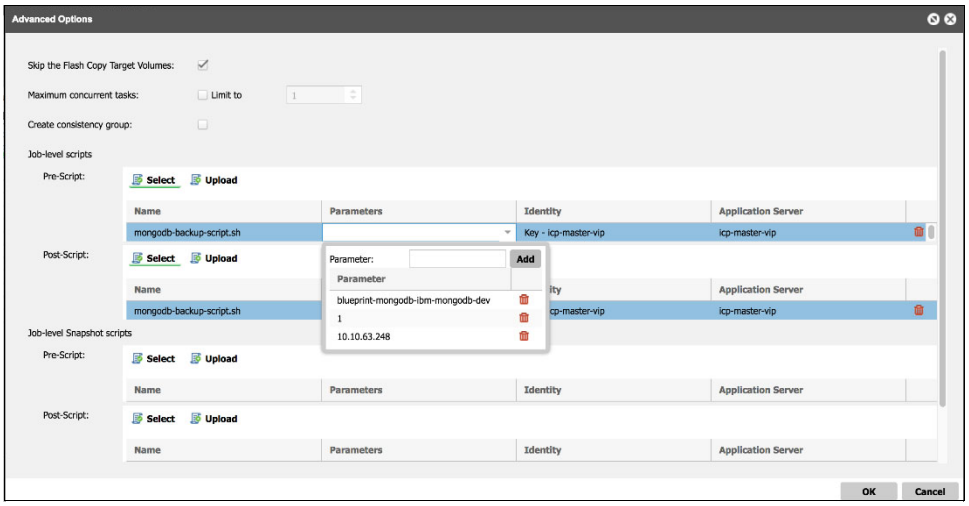


Figure 47 Configuring advanced options

As shown in Figure 47, the script is selected and the following parameters entered to be passed to the script:

a. Deployment Name of the Helm release as can be found in Figure 34 on page 35
b. 1 (to begin the backup) and 2 (to end the backup)
c. The master IP or proxy IP/virtual IP address, if configured to communicate with the pod

You must also specify the identity to use to communicate with the IBM Cloud Private cluster and the application server.

Once the job is executed successfully, IBM Spectrum Copy Data Management will create the target volume and start the snapshot (as shown in Figure 48).

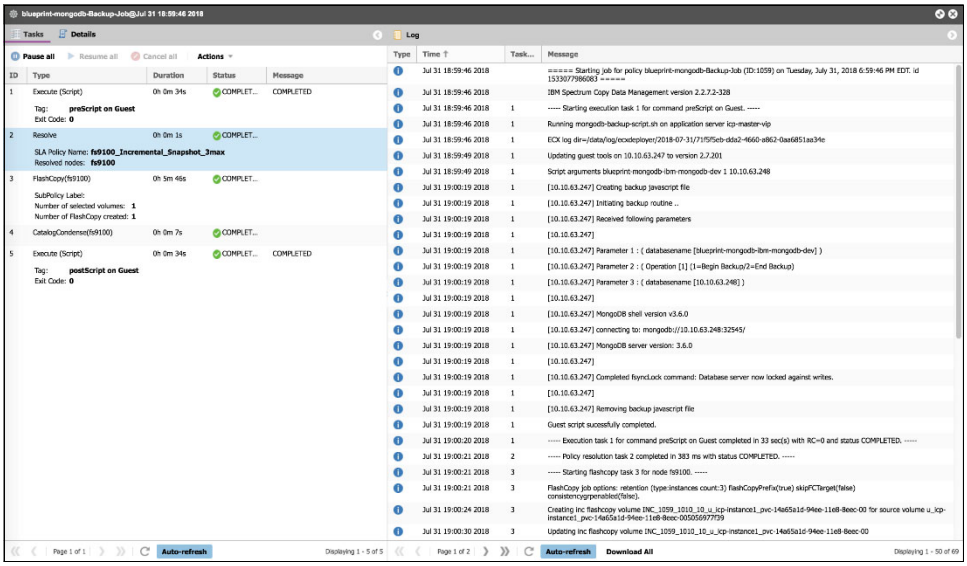


Figure 48 FlashCopy snapshot creation with IBM Spectrum Copy Data Management

IBM Spectrum Copy Data Management restore job creation

For the restore job creation, proceed as follows:

1. To create the restore policy for FlashCopy with IBM Spectrum Copy Data Management, click **Jobs** → **Storage Controller** → **IBM Spectrum Virtualize** → **Restore** (as shown in Figure 49):

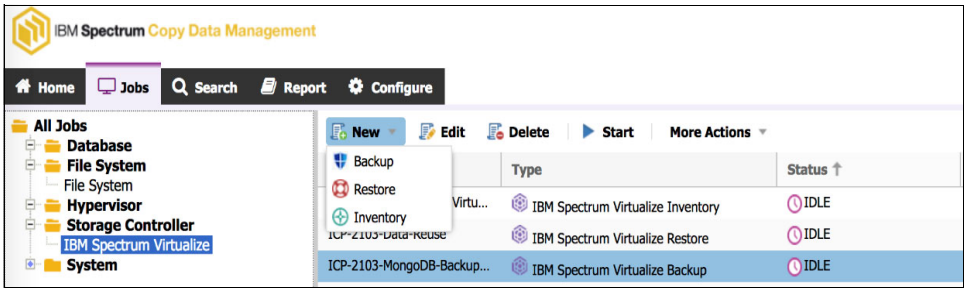


Figure 49 Restore policy creation for FlashCopy with IBM Spectrum Copy Data Management

2. Next, select **Restore Volumes**.
3. Select the source volume to be restored (as shown in Figure 50):

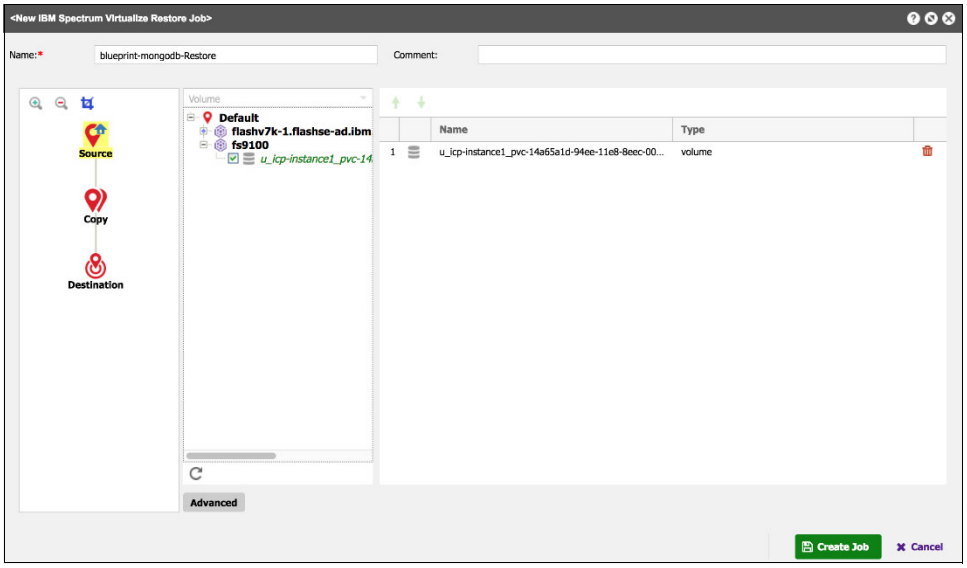


Figure 50 Source volume selection

4. Next, on the same page, click **Destination** and select an appropriate storage location.

   The version to be restored can be selected by clicking **Copy** and selecting the appropriate version, or you can leave the default selection, Use Latest (as shown in Figure 51).

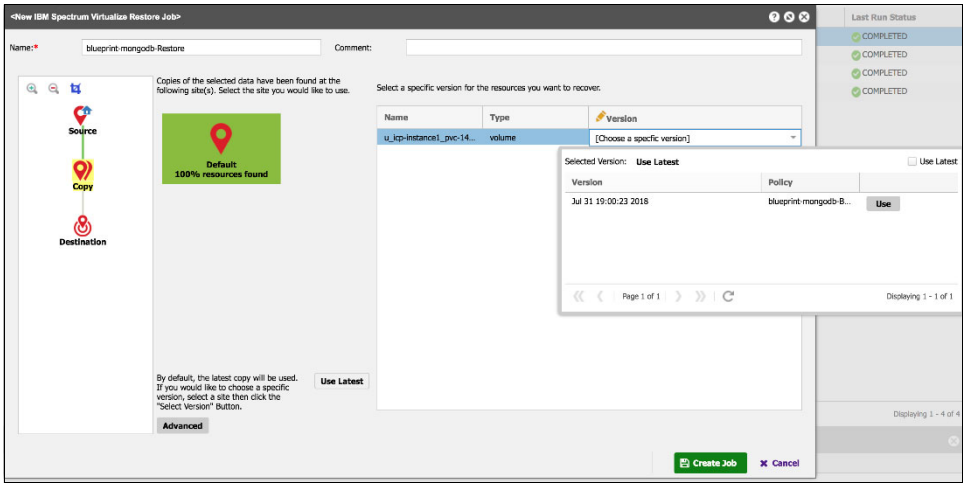


Figure 51 Selecting the version to be restored

5. The original volume must be offline prior to restoring the volume. This can be accomplished by using the Scale functionality of IBM Cloud Private to change the scale from **1** to **0**.

   a. To do this, go to **Workloads** → **Deployments**, click the **Action** menu on the left, and select **Scale**.

   b. Change the scale from **1** to **0**. The running pod will be stopped shortly.

6. Once the pod has stopped, run the restore job.

7. After completion of the restore job, scale the deployment from **0** to **1** and wait for it to start. The restore operation has completed.

IBM Spectrum Copy Data Management data reuse creation

Prepare the data reuse host for access to the IBM Storage System by following the "Configuring iSCSI/Fibre Channel for IBM Cloud Private 3.1.2 worker nodes" section of this Blueprint and installing the appropriate version of MongoDB server.

1. Download and install the MongoDB server on the data reuse host. The version of MongoDB used at the time of release of this document is 3.6.0:

```
wget http://downloads.mongodb.org/linux/mongodb-linux-x86_64-rhel70-3.6.0.tgz
tar xzvf mongodb-linux-x86_64-rhel70-3.6.0.tgz
cp mongodb-linux-x86_64-rhel70-3.6.0/bin/mongod /usr/local/bin/
```

2. To create the restore policy for FlashCopy with IBM Spectrum Copy Data Management, click **Jobs** → **Storage Controller** → **IBM Spectrum Virtualize** → **Restore** (as shown in Figure 52):

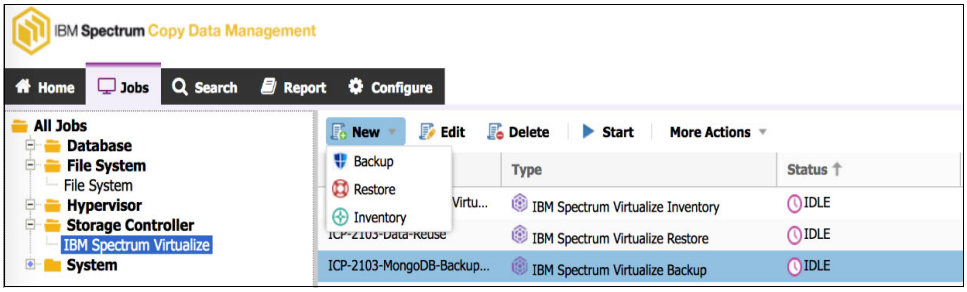


Figure 52 Restore policy creation for FlashCopy with IBM Spectrum Copy Data Management

3. From the available template options, select **Instant Disk Restore**.
4. Select the source volume to be restored (as shown in Figure 53):

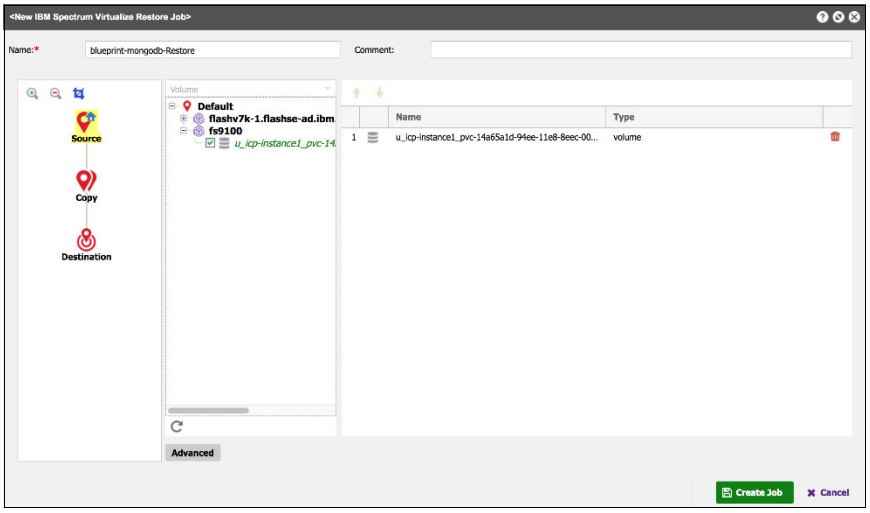


Figure 53 Selecting a source volume

5. Select the destination host that will mount the volume for data reuse.

   The version to be restored can be selected by clicking **Copy** and selecting the appropriate version, or you can leave the default selection, Use Latest (as shown in Figure 54):

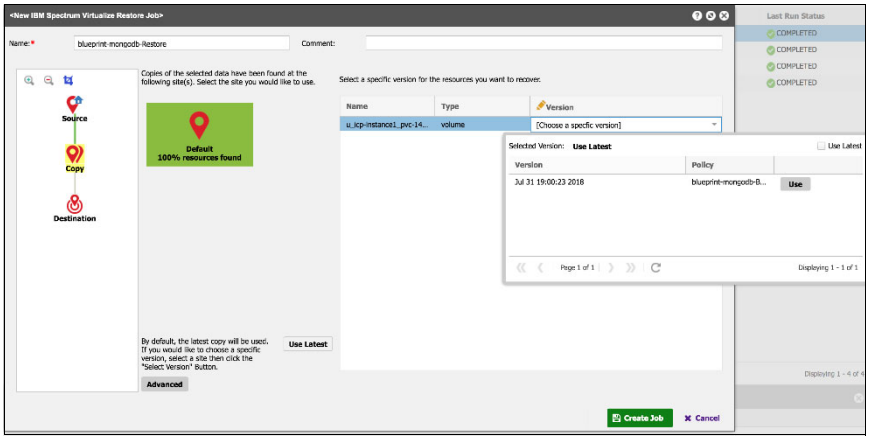


Figure 54 Selecting the version to be restored

6. Once the job is executed successfully, IBM Spectrum Copy Data Management will create a writable snapshot and map it to the destination host.

   The job activity will report the IBM IA® volume active message (as shown in Figure 55):

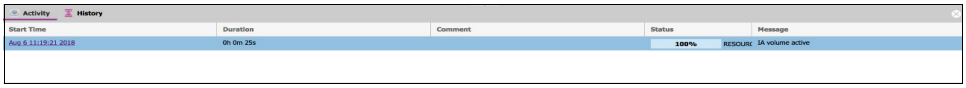


Figure 55 IA volume active message display

7. On the destination host, rescan the host bus adapters by running the following commands:

   For iSCSI hosts:

   ```
   iscsiadm -m session –rescan
   ```

   For both iSCSI and FC hosts:

   ```
   rescan-scsi-bus.sh -a
   ```

8. Once the volume has been discovered, determine the multipath mount point and volume ID by running the following command:

   ```
   multipath -ll
   ```

   Example 8 shows an example output of the command.

   Example 8 Multipath -11 example output

```
mpathe (360050768108207e7600000000000003e)
dm-2 IBM     ,2145

size=19G features='1 queue_if_no_path'
hwhandler='0' wp=rw

|-+- policy='service-time 0' prio=50
status=active

| `- 37:0:0:1 sdc 8:32 active ready running

`-+- policy='service-time 0' prio=10
status=enabled

  `- 38:0:0:1 sdb 8:16 active ready running
```

9. Next, create a mount point and point *mongod* to the existing database using the following commands:

   ```
   mkdir -p
   /datareuse/360050768108207e7600000000000003e


   mount /dev/mapper/mpathe
   /datareuse/360050768108207e7600000000000003e/


   /usr/local/bin/mongod --bind_ip_all --dbpath
   /datareuse/360050768108207e7600000000000003e/
   ```

   This will cause *mongod* to monitor all IP addresses of the destination host, listening on the default 27017 port. This will run *mongod* in the foreground, so the process can be ended by CTRL-C.

10. Once data reuse is no longer needed, run the following process to fully clean up the snapshot and release it from use. Right-click on the active session and select **End IA volume (Cleanup)** (as shown in Figure 56):

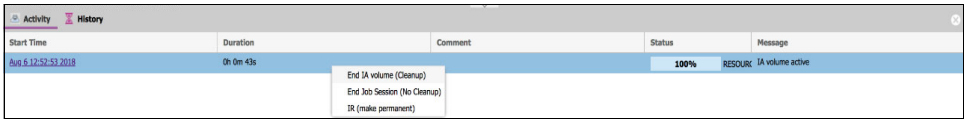


Figure 56 Cleaning up a snapshot after data reuse has completed

Summary

With this Blueprint, you can deliver a full stack of IBM® applications and middleware, from virtualization engines all the way up to the services catalog. Because the solution leverages open industry standards, clients are not locked into one flavor of application stack but instead can pick and choose the solution that is right for their environment. IBM Cloud™ Private provides clients an enterprise-grade self-service cloud stack that is enabled by IBM Spectrum Accelerate™ and the IBM Spectrum Virtualize storage infrastructure.

With this private cloud solution, clients can rest easy knowing that their data is within their control and that their solution allows them to meet stringent regulatory and compliance laws, deliver cloud native micro-services to extract meaning and value from their data, and manage operational expenses within the confines of their environment.

Get more information

How to get the benefits of cloud behind your firewall: IBM Cloud Private:

https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=KUW12527USEN

IBM FlashSystem 9100:

https://www.ibm.com/us-en/marketplace/flashsystem-9100

IBM Redbooks: IBM FlashSystem V9000 in a VersaStack Environment:

http://www.redbooks.ibm.com/redpapers/pdfs/redp5264.pdf

IBM Redbooks: Implementing the IBM Storwize V7000 and IBM Spectrum Virtualize V7.8:

https://www.redbooks.ibm.com/redbooks/pdfs/sg247938.pdf

IBM Redbooks: VersaStack Solution for File Storage Using IBM Storwize V5030 and Windows Server 2016:

http://www.redbooks.ibm.com/redpapers/pdfs/redp5442.pdf

IBM Spectrum Connect Version 3.6:

https://www.ibm.com/support/knowledgecenter/en/SS6JWS/landing/IBM_Spectrum_Connect_welcome_page.html

Appendix A. Using an existing fileset for volume creation

You might want to use the existing fileset for creating persistent volumes.

The following limitations exist for using existing fileset for volume creation:

- File set with pv-name specified in the PVC must exist in the filesystem specified in storage class and it should be linked.
- File set level quota must be enabled for the filesystem specified in the storage class.
- Storage-class parameters **uid**, **gid**, **inode-limit**, **fileset-type** are not valid for this functionality and must not be specified in storage class
- Quota on the file set must be equal or greater than storage requested in the PVC.
- If the **pv-name** is not specified in the PVC yaml configuration then a random PV name is generated, and the IBM Storage Enabler for Containers tries to look up the fileset with that random PV name. However, the random PV name most likely does not exist, and hence the PVC won't become available for use.
- If an existing fileset is used with the **reclaimPolicy** set to `retain`, then deleting the PVC does not delete the PV. The PV remains in a released state. If the released PV is deleted manually, and then you try to create a PVC with the same fileset name, the process will fail.

Use the following steps to create a volume using an existing fileset:

1. Create a new storage-class and set parameter **isPreexisting** to **true**, as shown in Example 9.

Example 9 Setting parameter in a new storage-class

```
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
name: "<NAME>"
labels:
product: ibm-storage-enabler-for-containers
# annotations:
# storageclass.beta.kubernetes.io/is-default-class: "true"
provisioner: "ubiquity/flex"
parameters:
backend: "spectrum-scale"
filesystem: "<filesystem name>"
type: "fileset"
isPreexisting: “true”
```

2. Create a PVC using the storage class created in step 1 and set the parameter **pv-name** to **<name of existing fileset>**, as shown in Example 10.

Example 10 Create a PVC and set parameter

```
kind: PersistentVolumeClaim apiVersion: v1
metadata:
name: "<PVC name>"
labels:
product: ibm-storage-enabler-for-containers
pv-name: "<name of existing fileset>"
spec:
```

```
storageClassName: <Storage Class Name>
accessModes:
- ReadWriteOnce # ReadWriteOnce and ReadWriteMany
resources:
requests:
storage: <Number>Gi
```

Appendix B. IBM Spectrum Scale usage restrictions

Make note of the following conditions before using IBM Storage Enabler for Containers with IBM Spectrum Scale:

- IBM Spectrum Scale storage tiering/pooling is not surfaced to the PVC/POD level. To gain a similar look and feel to the IBM Spectrum Scale storage tiering, it is advised to create a separate IBM Spectrum Scale file system on each type of desired storage (such as flash, SSD and SAS). After that file system is created, associate filesystem, fileset-type and storage classes in the **`storage-class-template.yml`** and **`pvc-template.yml`** files. This will allow multiple types of storage to be presented to the pods via multiple PVCs.
- IBM Spectrum Scale must be preinstalled along with the IBM Spectrum Scale GUI.
- IBM Storage Enabler for Containers is only supported on IBM Spectrum Scale 5.0.0 and above.
- At least one filesystem must exist and must be mounted on all the worker nodes.
- Quota must be enabled for all the filesystems being used for creating persistent volumes.
- All Kubernetes worker nodes must have the IBM Spectrum Scale client installed on them.
- IBM Cloud Private nodes should be configured to schedule pods after the IBM Spectrum Scale filesystem is mounted on worker node(s). This can be monitored by **`<mmlsmount all>`** and it is recommended to script IBM Cloud Private startup based on the return code/systemd results. If scripting off of this command, use the **`-Y`** parameter, as this is parseable and the formatting is consistent release to release.
- If the IBM Spectrum Scale filesystem is unmounted, or if there is an issue with IBM Spectrum Scale mounted on a particular node, then the applications in the containers that are using the PVC from IBM Spectrum Scale will throw an I/O error. IBM Storage Enabler for Containers does not monitor IBM Spectrum Scale and is unaware of any failure in the I/O path. Kubernetes also does not monitor IBM Spectrum Scale and is unaware of any failure in the I/O path. It is recommended to monitor IBM Spectrum Scale to avoid any issues. Monitoring can be accomplished via scripting, such as an IBM General Parallel File System (GPFS™) callback set to take action if the GPFS filesystem is unmounted or shut down (such as through script-based activation of cordon or drain node).
- RWX support: The same PVC cannot be attached or mounted on more than one pod on the same host.
- If a single PVC is used by multiple pods, then it is the application's responsibility to maintain data consistency.
- It is recommended to create the PVCs one after another. You can create a new PVC after all the earlier PVCs created using the SEC are in bound state.
- Creating a large number of PVCs in a single batch or deleting all of them simultaneously is not recommended. Such actions might result in overloading the IBM Spectrum Scale GUI node, which in turn might lead to the failure of creation and deletion of filesets on IBM Spectrum Scale.
- The **`uid`**, **`gid`**, **`inode-limit`**, and **`fileset-type`** parameters from the storage-classes are only allowed for new fileset creation.
- The volume for the **`ubiquity-db`** is created as a dependent fileset and changing it to an independent fileset is not supported.
- For each **`uid-gid`** combination, a new storage class needs to be defined.
- You must define a new storage class for each fileset while using the existing filesets for persistent volume.

- Advanced IBM Spectrum Scale functionality such as active file management, remote mount, encryption, and compression are not supported by IBM Storage Enabler for Containers.
- The persistent volumes created using IBM Storage Enabler for Containers with IBM Spectrum Scale as backend use the IBM Spectrum Scale quota to make sure that the users cannot use more storage space than the amount specified in the PVC. However, this does not guarantee that the storage specified in the PVC is actually available. It is up to the storage administrator to make sure that the required storage is available on the IBM Spectrum Scale filesystem.
- IBM Storage Enabler for Containers does not check the storage space available on the IBM Spectrum Scale filesystem before creating the PVC. You can use the Kubernetes storage resource quota to limit the number of PVCs or storage space.
- The file set created by the storage IBM Storage Enabler for Containers should not be unlinked or deleted from any other interface.
- The filesystem used for the persistent volume must be mounted on all the worker nodes at all times.
- IBM Cloud Private and IBM Spectrum Scale GUI uses the port 443.
- IBM Storage Enabler for Containers does not support volume expansion for storage class.
- The **`df`** command inside the container shows the full size of the IBM Spectrum Scale filesystem.

Appendix C. IBM PowerVC FlexVolume driver setup on IBM Cloud Private

Introduction

PowerVC can be used as the infrastructure-as-a-service (IaaS) layer that hosts the VMs on IBM Power Systems™ for the IBM Cloud Private master and worker nodes. With the IBM PowerVC FlexVolume driver (ibm-powervc-k8s-volume-driver) Helm chart, PowerVC can also be used to provision storage volumes and to mount storage for containers.

Requirements

1. PowerVC 1.4.1 or later (PowerVC 1.4.0 may be used if using Fibre Channel-attached storage).
2. IBM Cloud Private 2.0.1.2 or later (i.e., Kubernetes 1.9.1 or later).
3. FlexVolume driver directory mounted in the controller manager container. This is mounted by default starting in IBM Cloud Private 2.1.0.3.

Supported storage systems

See the “Storage capabilities” section at the following resource:

https://www.ibm.com/support/knowledgecenter/en/SSXK2N_1.4.1/com.ibm.powervc.standard.help.doc/powervc_features.html

Deployment architecture

Figure 57 displays the PowerVC deployment architecture.

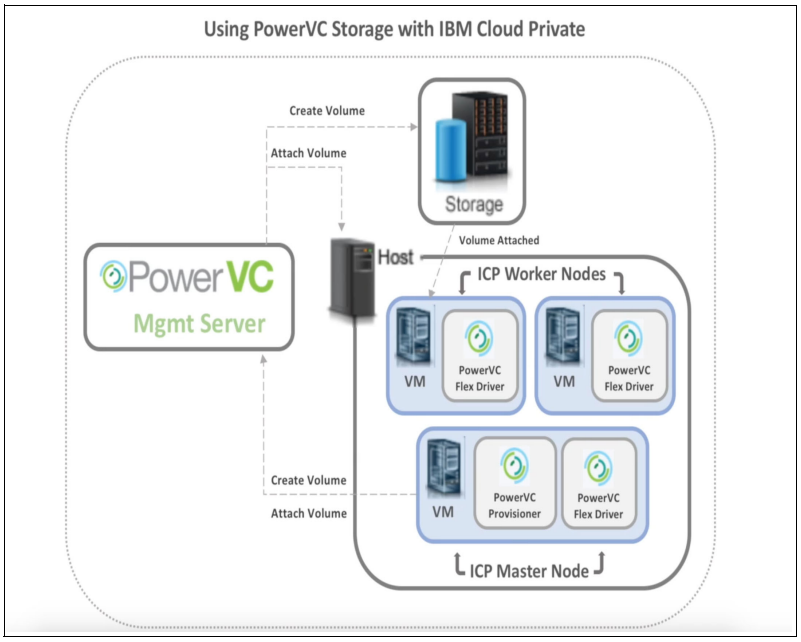


Figure 57 PowerVC deployment architecture

Installation steps

To use PowerVC with IBM Cloud Private, the user should install the PowerVC FlexVolume driver (ibm-powervc-k8s-volume-driver) Helm chart, available in the out-of-box IBM Cloud Private software catalog. The Helm chart will deploy the PowerVC FlexVolume driver on the IBM Cloud Private infrastructure as shown in the deployment architecture, see Figure 57 on page 57.

1. In the IBM Cloud Private user interface, navigate to the list of Helm charts and select ibm-powervc-k8s-volume-driver.
2. On clicking the chart, you will see the overview of the chart, including components of the PowerVC FlexVolume driver that the chart will create and deploy.

   The chart also lists the information to configure PowerVC as the infrastructure layer. This information needs to be provided once the user clicks **Configure** to configure the chart.
3. On clicking **Configure**, the user is taken to the following page to enter the PowerVC configuration details. The details required are for managing the PowerVC server that is managing the storage, which will be used for providing volumes to the containers running under IBM Cloud Private, see Figure 58.

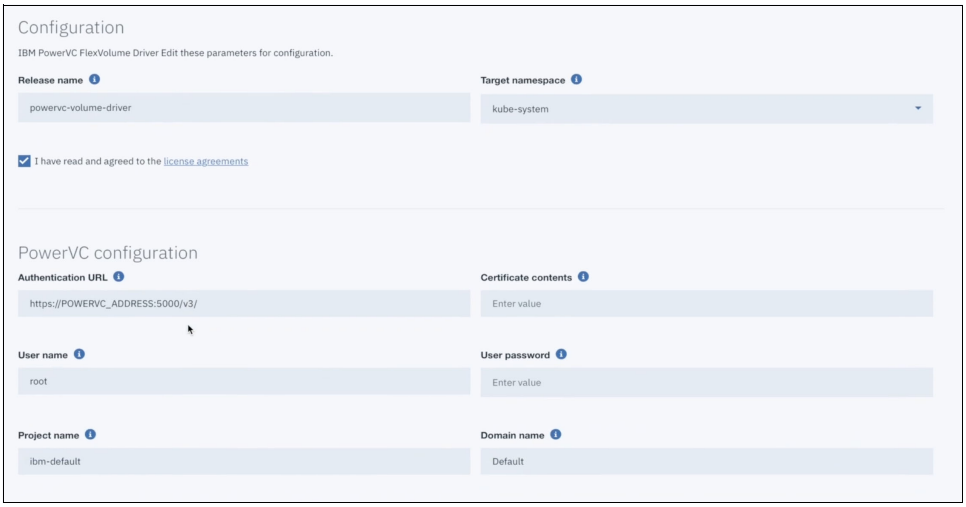


Figure 58 PowerVC FlexVolume configuration

4. Once the user has entered the required details for PowerVC server configuration on IBM Cloud Private, the Helm chart will be listed under Helm releases. On clicking on the configured chart, the chart will list all the components that have been configured on IBM Cloud Private, as follows:
   a. **Volume provisioner container**: This component is responsible for handling volume creation requests.
   b. **Secret**: Stores PowerVC credentials.
   c. **Config Map**: Stores PowerVC server configuration.
   d. **Storage Class**: The default storage class configured on IBM Cloud Private. Any pod requesting a persistent volume will now make use of this class, unless explicitly otherwise specified.
   e. **DaemonSet**: FlexVolume driver containers running on each of master and worker nodes that attach/detach volumes to the worker nodes.

5. Now IBM Cloud Private is configured with PowerVC server as its infrastructure layer. Any pod that is on the worker node and is managed by PowerVC can now request persistent volumes to be attached to it through persistent volume claims:
   a. When a pod is created and requests persistent volume, Kubernetes will use the volume provisioner to create a persistent volume on PowerVC. Then the FlexVolume driver will identify the worker node that is running the pod and attach the newly created volume to it. The FlexVolume driver will also mount the volume as a file system to the container.
   b. When a pod is deleted, the volume provisioner and FlexVolume driver will unmount, detach and delete the volume from the backend storage managed by PowerVC.

Appendix D: Sample scripts for MongoDB and Db2 database backup

These example scripts are provided as-is as a resource to help you write your own backup script.

MongoDB script

```
#!/usr/bin/bash
# ************************************************************************************************************** */
# This example script is provided as a resource to help you write your own backup script.  Use at your own risk */
# Input variables used by the script:
# $1 = DeploymentName as show in IBM Cloud Private
# $2 = (1=Begin Backup/2=End Backup)
# $3 = Proxy IP or Virtual IP Address used to communicate with the pod
# ************************************************************************************************************** */

echo "Creating backup javascript file"

cat <<EOF >$ECXSESSIONDIR/backup-script.$$.js
/* ************************************************************************************************************** */
/* This example script is provided as a resource to help you write your own backup script.  Use at your own risk */
/* ************************************************************************************************************** */

/*   checks if server is currently locked and returns true or false */
isLocked=function() {
   var lockCheck = db.currentOp();
   if ( lockCheck.hasOwnProperty("fsyncLock") && lockCheck.fsyncLock) return true;
   else return false;
}

runBeginBackup=function() {

   /* because of possibility of multiple lock/unlock pairs, check if the DB is locked now, so */
   /* that we don't have to worry about checking if it's unlocked when we are finished */
   var lockedAtStart = isLocked();
   if (lockedAtStart) {
      throw "Aborting backup since someone else already has server locked!";
   }

   /* Check that if this is a replica set that we are a secondary */
   var isMaster=db.isMaster();
   if ( isMaster.secondary ) {     /* We are connected to a primary */
      throw "This is a replica set. Primary is running on " + isMaster.primary + " host !";
   }

   /* Lock the server, check success */
   var lockResult = db.fsyncLock();
   if ( lockResult.ok != 1 ) {
      print("\nDidn't successfully lock the server. Returned status is " + lockResult.code + " " +
lockResult.errmsg);
      throw "Exiting after error locking";
   } else print("\nCompleted fsyncLock command: Database server now locked against writes.\n");
}

runEndBackup=function(){

   /* Now clean up */

   print("\nReleasing the database lock now.");

   /* we always run unLock regardless of success of backup! */
   var unlockResult = db.fsyncUnlock();
   if ( unlockResult.ok != 1 ) {
      if ( errmsg == "not locked" ) print("\nServer was already unlocked!");
      else {
          print("\nCouldn't successfully unlock the database. \nReturned status is " + tojson(unlockResult));

       /* Next section can be uncommented to keep trying to unlock if you want to guarantee the server is unlocked
          when the script is finished.  However, this is not safe, since other processes may not be paying
          attention to already locked state and queueing their lock requests anyway */
      /*  print("\tWill try again every 5 seconds!");

          while (unlockResult.ok!=1) {
              if (unlockResult.errmsg=='not locked') {
                   print("\t\tServer no longer locked");
```

```
                    break;
                }
                sleep(5000);
                unlockResult=db.fsyncUnlock();
            }
         */
        }
    }

    if (!isLocked() ) {
       print("\nServer is unlocked.  We are finished.\n");
    }
}

/*
 *  main()
 */

var dowhat;

switch( dowhat ){
       default: print("Received : [" + dowhat + "]\nExpecting 1 to begin the backup or 2 to end the backup"); break;
    case 1: runBeginBackup(); break;
    case 2: runEndBackup(); break;
}
EOF

echo "Initiating backup routine .."

echo "Received following parameters

Parameter 1 : ( DeploymentName [$1] )
Parameter 2 : ( Operation    [$2] (1=Begin Backup/2=End Backup)
Parameter 3 : ( ProxyIPAddress [$3] )
"

[[ -z ${1} ]] && echo "ERROR: Database name is not provided. Aborting !!" && exit 1
[[ -z ${2} ]] && echo "ERROR: Operation Begin/End (1/2) is not provided. Aborting !!" && exit 1
[[ -z ${3} ]] && echo "ERROR: Proxy IP address is not provided. Aborting !!" && exit 1
[[ ${2} -lt 1 || ${2} > 2 ]] && echo "ERROR: Operation type must be 1 or 2 " && exit 1

# Update cloudctl login information below with appropiate cluster address, username, password and namespace
echo "Logging in to IBM Cloud Private"
/usr/local/bin/cloudctl login -a https://mycluster.icp:8443 --skip-ssl-validation -u admin -p admin -n default

/usr/local/bin/mongo --host $3 --port $(kubectl get svc --namespace default $1 -o
jsonpath='{.spec.ports[0].nodePort}') --username mongo --password $(kubectl get secret --namespace default $1 -o
jsonpath="{.data.password}" | base64 --decode; echo) --authenticationDatabase "admin" --eval "var dowhat=$2"
$ECXSESSIONDIR/backup-script.$$.js

echo "Removing backup javascript file"
rm -rf $ECXSESSIONDIR/backup-script.$$.js
```

Db2 script

```
#!/usr/bin/bash
# ************************************************************************************************************** */
# This example script is provided as a resource to help you write your own backup script.  Use at your own risk */
# Default instance name of db2inst1 is assumed.
# Input variables used by the script:
# $1 = DB2 Pod as show in IBM Cloud Private
# $2 = (1=Begin Backup/2=End Backup)
# $3 = Proxy IP or Virtual IP Address used to communicate with the pod
# ************************************************************************************************************** */

echo "Initiating backup routine .."

echo "Received following parameters

Parameter 1 : ( PodName      [$1] )
Parameter 2 : ( Operation    [$2] (suspend/resume)
Parameter 3 : ( DatabaseName [$3] )
"

[[ -z ${1} ]] && echo "ERROR: DB2 Pod name is not provided. Aborting !!" && exit 1
[[ -z ${2} ]] && echo "ERROR: Operation suspend/resume is not provided. Aborting !!" && exit 1
[[ -z ${3} ]] && echo "ERROR: Database Name is not provided. Aborting !!" && exit 1
[[ ${2} != suspend && ${2} != resume ]] && echo "ERROR: Operation type must be 'suspend' or 'resume' " && exit 1

# Update cloudctl login information below with appropiate cluster address, username, password and namespace
echo "Logging in to IBM Cloud Private"
/usr/local/bin/cloudctl login -a https://mycluster.icp:8443 --skip-ssl-validation -u admin -p admin -n default

kubectl exec -it $1 -- /bin/su - db2inst1 -c "/opt/ibm/db2/V11.1/bin/db2 connect to $3; /opt/ibm/db2/V11.1/bin/db2
set write $2 for database"
```

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

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Privacy policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at http://www.ibm.com/privacy and IBM's Online Privacy Statement at http://www.ibm.com/privacy/details in the section entitled "Cookies, Web Beacons and Other Technologies," and the "IBM Software Products and Software-as-a-Service Privacy Statement" at http://www.ibm.com/software/info/product-privacy.

April 2019

Please recycle

ISBN 0738457647

REDP-5533-00