Red Hat OpenStack Platform 15

Google Cloud Backup Guide

Configuring OpenStack Block Storage Backups to Use Google Cloud Storage

Last Updated: 2020-05-13

Red Hat OpenStack Platform 15 Google Cloud Backup Guide

Configuring OpenStack Block Storage Backups to Use Google Cloud Storage

OpenStack Team
rhos-docs@redhat.com

Legal Notice

Copyright © 2020 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This document describes how to deploy the OpenStack Block Storage Backup Service to use Google Cloud Storage as a back end. These instructions are specific to an overcloud deployment. Backup integration with Google Cloud Storage is offered in this release as a Technology Preview. Do not deploy Technology Preview features in production environments. For more information about Technology Previews, see:

Table of Contents

CHAPTER 1. INTRODUCTION TO CONFIGURING BLOCK STORAGE BACKUPS TO USE GOOGLE CLOUD STORAGE

The Red Hat OpenStack Platform director is a toolset to install and manage a complete Red Hat OpenStack environment. It is based primarily on the OpenStack project TripleO (*OpenStack-on-OpenStack*). The primary objective of the director is to orchestrate a functional, Enterprise-grade OpenStack deployment with minimal manual configuration. Director addresses many of the issues inherent in configuring individual OpenStack components manually.

The OpenStack environment that you deploy with the director is called the *overcloud*. The overcloud contains all of the components that provide services to end users, including Block Storage. This document provides guidance on how to deploy custom back ends to the Block Storage service on the overcloud.

Red Hat OpenStack Platform supports several third-party services, devices, and applications for functions ranging from block storage back ends to data processing. You can configure the Block Storage service to use Google Cloud as a backup storage service. This Google Cloud integration is available in this release as a technology preview feature.

> **IMPORTANT**
>
> *Technology Preview* features are not fully supported by Red Hat. The deployment scenario described in this document should only be used for testing, and should not be deployed in a production environment.
>
> For more information about Technology Preview features, see Scope of Coverage Details.

This document presents a test scenario where the Block Storage service on an overcloud deployment is configured to back up volumes to Google Cloud storage. This test scenario requires the following prerequisites:

- The overcloud has already been deployed through director. For more information about director deployment, see the Director Installation and Usage guide.
- You have the username and password of an account with elevated privileges. You can use the same account that you create to deploy the overcloud. If you follow the steps in the Director Installation and Usage guide, you create a user named **stack** for this purpose.
- The Block Storage service is installed on the Controller nodes, or, on every Controller node in an HA deployment.
- You have a Google account with access to Google Cloud Platform. The Block Storage service uses this account to access and use Google Cloud to store backups.

1.1. UNDERSTANDING THE CONFIGURATION PROCESS

Configuring the Block Storage service to use Google Cloud as a backup service involves the following steps:

1. Creating and downloading the service account credentials of your Google account. For more information, see (Chapter 2, *Creating and downloading the GCS credentials file*).

2. Creating an environment file to define the Block Storage settings that you require. For more information, see (Chapter 3, *Create the Block Storage environment file*). This environment file also uses the service account credentials that you created in the previous step.

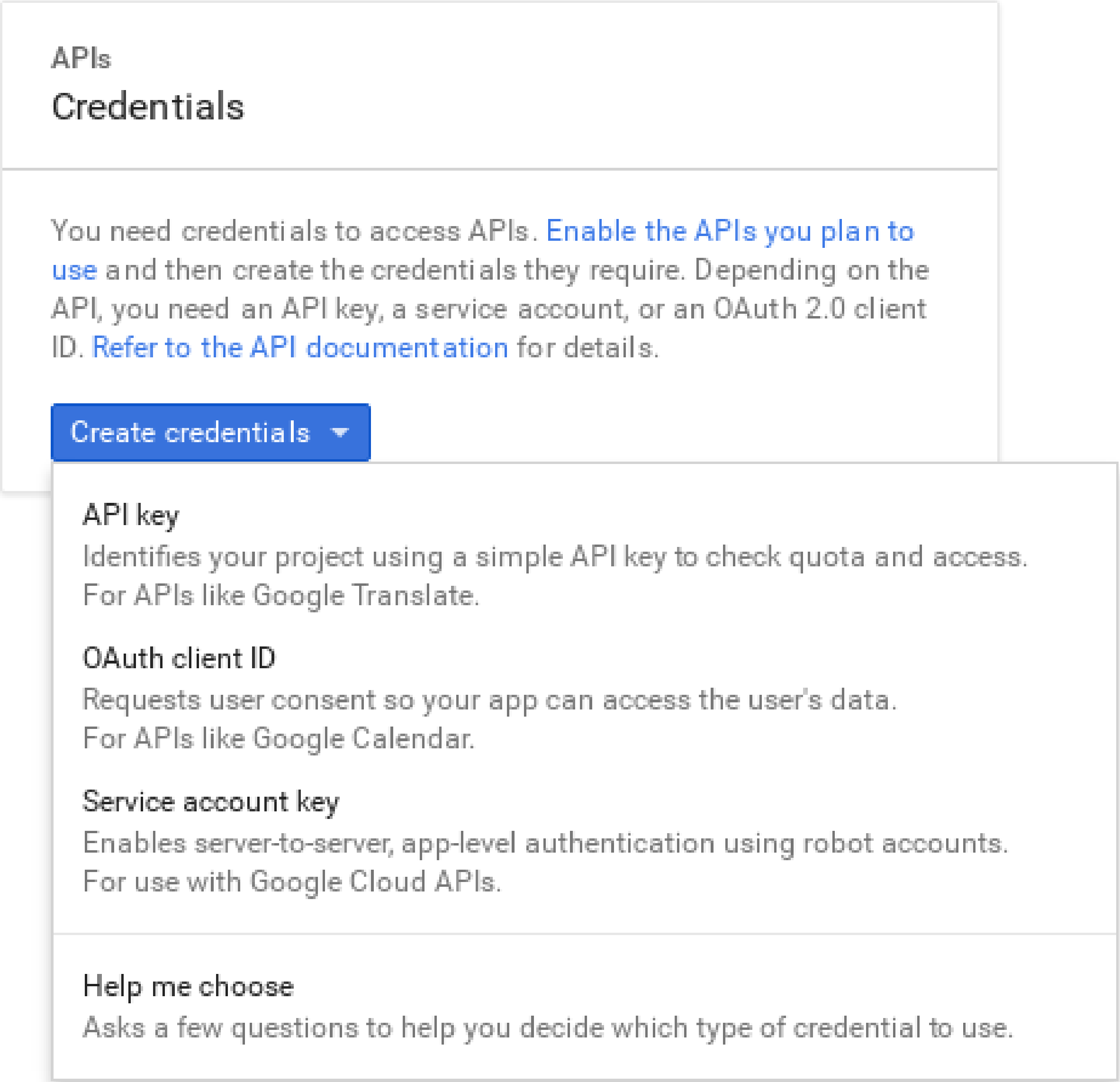
3. Re-deploying the overcloud using the new environment file. For more information, see (Chapter 4, *Re-deploying the overcloud*).

CHAPTER 2. CREATING AND DOWNLOADING THE GCS CREDENTIALS FILE

The Block Storage service needs your Google credentials in order to access and use Google Cloud for backups. You can provide these credentials to Block Storage by creating a service account key:

1. Log in to the Google developer console (http://console.developers.google.com) using your Google account.

2. Click the **Credentials** tab, click **Create credentials**, and select **Service account key** from the **Create credentials** drop-down list.



APIs

Credentials

You need credentials to access APIs. Enable the APIs you plan to use and then create the credentials they require. Depending on the API, you need an API key, a service account, or an OAuth 2.0 client ID. Refer to the API documentation for details.

API key

Identifies your project using a simple API key to check quota and access.
For APIs like Google Translate.

OAuth client ID

Requests user consent so your app can access the user's data.
For APIs like Google Calendar.

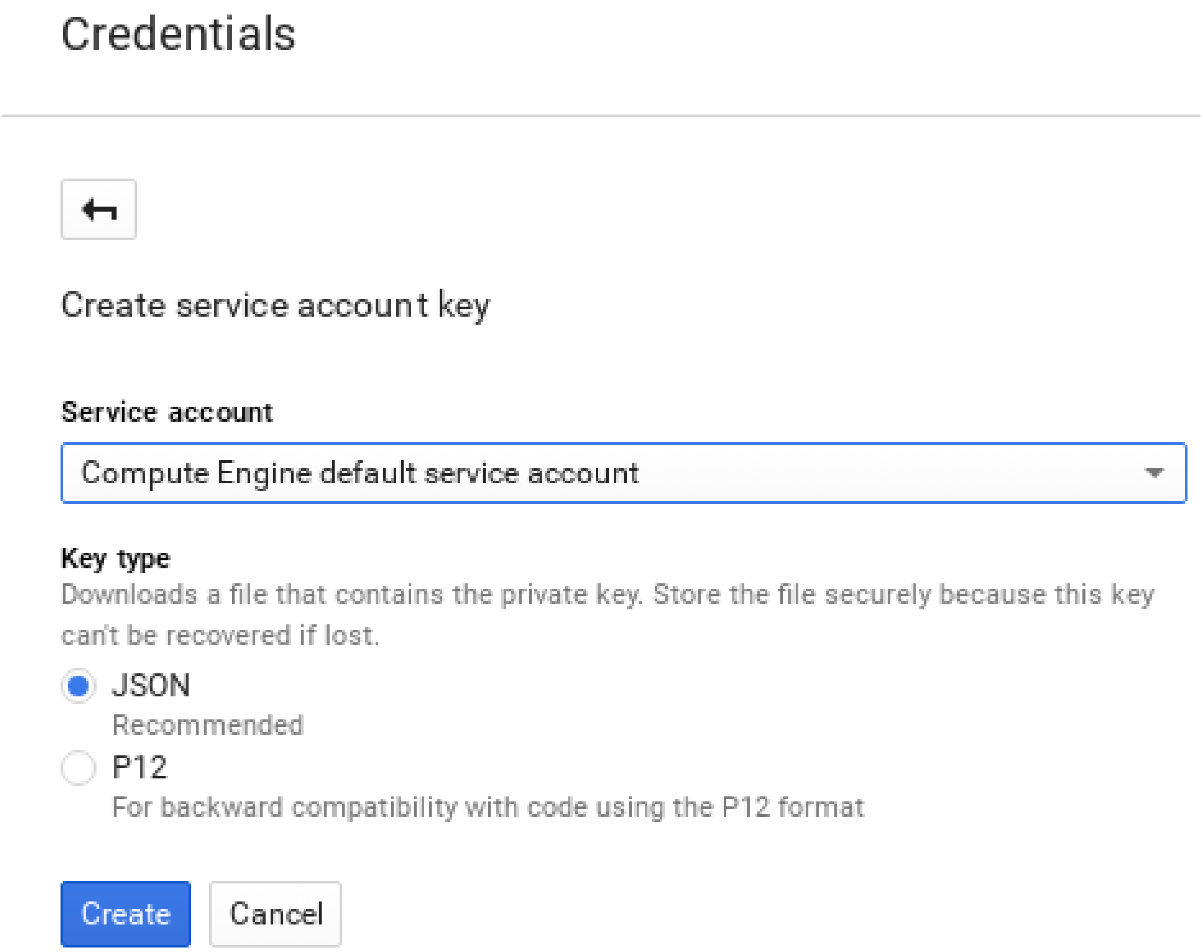
Service account key

Enables server-to-server, app-level authentication using robot accounts.
For use with Google Cloud APIs.

Help me choose

Asks a few questions to help you decide which type of credential to use.

3. On the **Create service account key** page, select the service account that you want the Block Storage service to use from the **Service account** drop-down list:



Credentials

Create service account key

Service account

Compute Engine default service account

Key type

Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

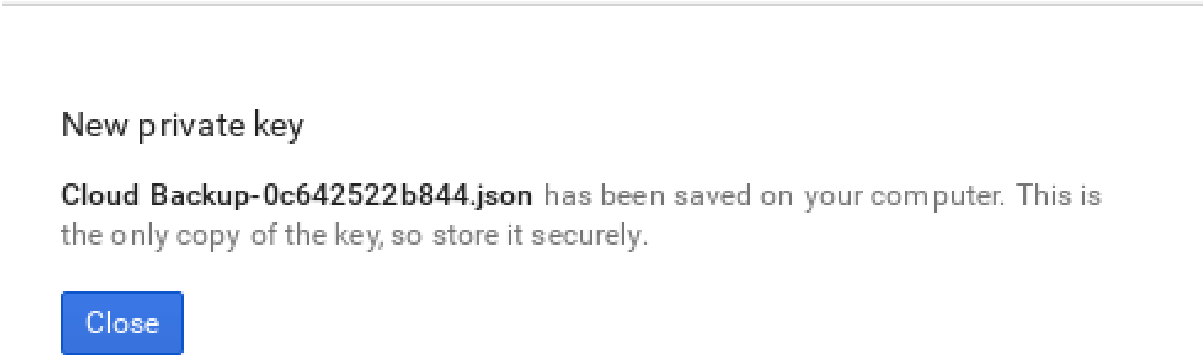
● JSON

Recommended

○ P12

For backward compatibility with code using the P12 format

4. On the **Create service account key** page, select the **JSON** radio button from the **Key type** section and click **Create**.
   The browser downloads the key to the default download location:



New private key

Cloud Backup-0c642522b844.json has been saved on your computer. This is the only copy of the key, so store it securely.

5. Open the file and note the value of the **project_id** parameter:

```
{
  "type": "service_account",
  "project_id": "cloud-backup-1370",
...
```

You use the */etc/cinder/Cloud-Backup.json* key in Chapter 3, *Create the Block Storage environment file* in particular, the value of **project_id**, and the absolute path to the file.

6. Copy the key file to */etc/cinder/* on any Controller node and change the user, group, and permissions of the key file to match that of */etc/cinder/cinder.conf*. This ensures that the Block Storage service can use the key file:

   ```
   # cp Cloud-Backup.json /etc/cinder/
   # chown cinder:cinder /etc/cinder/Cloud-Backup.json
   # chmod 0600 /etc/cinder/Cloud-Backup.json
   ```

7. Copy the key file to **/etc/cinder/Cloud-Backup.json_** on each Controller node. Use **rsync -a** to preserve the permissions and ownership settings:

   ```
   # rsync -a /etc/cinder/Cloud-Backup.json root@CONTROLLERHOST:/etc/cinder/
   ```

   Replace *CONTROLLERHOST* with the hostname of a target Controller.

CHAPTER 3. CREATE THE BLOCK STORAGE ENVIRONMENT FILE

The environment file contains the settings that you want to apply to the Block Storage service. In this case, configure the Block Storage service to store volume backups to Google Cloud. For more information about environment files, see the Director Installation and Usage guide.

Define each setting in the environment file in the following format:

Entry format

```
SECT/PARAM: # 1
  value: CONFIG # 2
```

1 All Block Storage settings are configured in the */etc/cinder/cinder.conf* file of the node that hosts the Block Storage service. This file is divided into different sections, making it easier to manage different settings. *PARAM* is the setting you want to apply, and *SECT* is the section to which it belongs.

2 *CONFIG* is the value you want to set to *PARAM*.

In this document, all parameters are declared in the **DEFAULT** section. The following table describes each setting required to configure Google Cloud Storage (GCS) as your backup service:

1. Google Cloud backup settings

PARAM	Default	*CONFIG* Description
backup_driver	cinder.backup.drivers.swift	The backup driver that the Block Storage server uses. For Google Cloud Storage, use **cinder.backup.drivers.gcs.GoogleBackupDriver**.
backup_gcs_credential_file		The absolute path to the service account key file you created earlier in Chapter 2, *Creating and downloading the GCS credentials file*.
backup_gcs_bucket		The GCS bucket, or object storage repository that you want to use, which may or may not exist. If you specify a non-existent bucket, the Google Cloud Storage backup driver creates one using the name you specify here. For more information, see Buckets and Bucket name requirements.

backup_gcs_bucket_location	US	The location of the GCS bucket. This value is only used if you specify a non-existent bucket in **backup_gcs_bucket**; in which case, the Google Cloud Storage backup driver specifies this as the GCS bucket location. For more information, see Bucket Locations.
backup_gcs_project_id		The project ID of the service account you are using, as noted in the **project_id** of the service account key from Chapter 2, *Creating and downloading the GCS credentials file*.
backup_gcs_object_size	52428800	The size (in bytes) of GCS backup objects.
backup_gcs_block_size	32768	The size (in bytes) that changes are tracked for incremental backups. This value must be a multiple of the **backup_gcs_object_size** value.
backup_gcs_user_agent	gcscinder	The HTTP user-agent string for the GCS API.
backup_gcs_reader_chunk_size	2097152	GCS objects are downloaded in chunks of this size (in bytes).
backup_gcs_writer_chunk_size	2097152	GCS objects are uploaded in chunks of this size (in bytes). To upload files as a single chunk instead, use the value **-1**.
backup_gcs_num_retries	3	Number of retries to attempt.
backup_gcs_storage_class	NEARLINE	Storage class of the GCS bucket. This value is used if you specify a non-existent bucket in **backup_gcs_bucket**; in which case, the Google Cloud Storage backup driver specifies this as the GCS bucket storage class. For more information, see Storage Classes.

backup_gcs_retry_error_codes	429	List of GCS error codes.
backup_gcs_enable_progress_timer	True	Boolean to enable or disable the timer for sending periodic progress notifications to the Telemetry service (**ceilometer**) during volume backups. This is enabled by default (**True**)

WARNING

When creating new buckets, Google Cloud Storage charges according to the storage class that you choose (**backup_gcs_storage_class**). The default **NEARLINE** class is appropriate for backup services.

You cannot edit the location or class of a bucket after it is created. For more information, see Managing a bucket's storage class or location .

The following sample shows the typical contents of an environment file to configure GCS as a backup service:

/home/stack/templates/gcs-backup.yaml

```
parameter_defaults:
  ControllerExtraConfig: # 1
    cinder::config::cinder_config:
        DEFAULT/backup_driver
            value: cinder.backup.drivers.gcs.GoogleBackupDriver
        DEFAULT/backup_gcs_credential_file
            value: /etc/cinder/Cloud-Backup.json
        DEFAULT/backup_gcs_bucket
            value: mycinderbucket
        DEFAULT/backup_gcs_project_id
            value: cloud-backup-1370
        DEFAULT/backup_gcs_user_agent
            value: myuseragent
```

1. **ControllerExtraConfig** defines custom settings that you want to apply to all Controller nodes. Include the **cinder::config::cinder_config** to apply the settings to the Block Storage (**cinder**) service.

After you create the environment file, re-deploy the overcloud with the new configuration. For more information, see Chapter 4, *Re-deploying the overcloud*.

CHAPTER 4. RE-DEPLOYING THE OVERCLOUD

After you have created the environment file file in **/home/stack/templates/**, log in as the **stack** user. Then, run the **openstack overcloud deploy** command to deploy the new configuration:

```
$ openstack overcloud deploy --templates -e /home/stack/templates/gcs-backup.yaml
```

IMPORTANT

If you passed any extra environment files when you created the overcloud, pass them again here using the -e option to avoid making undesired changes to the overcloud.

For more information, see Deployment command options in the *Advanced Overcloud Customization* Guide.