Cloud Volumes ONTAP for Google Cloud

NetApp Automation

NetApp
May 08, 2024

Table of Contents

Cloud Volumes ONTAP for Google Cloud

Cloud Volumes ONTAP for Google Cloud - Burst to cloud

This article supports the NetApp Cloud Volumes ONTAP for Google Cloud Automation Solution, which is available to NetApp customers from the BlueXP Automation Catalog.

The Cloud Volumes ONTAP for Google Cloud Automation Solution automates the containerized deployment of Cloud Volumes ONTAP for Google Cloud, enabling you to deploy Cloud Volumes ONTAP for Google Cloud rapidly, without any manual intervention.

Before you begin

- You must download the Cloud Volumes ONTAP for Google Cloud - Burst to cloud automation solution through the BlueXP web UI. The solution is packaged as `cvo_gcp_flexcache.zip`.
- You must install a Linux VM on the same network as Cloud Volumes ONTAP.
- After you install the Linux VM, you must follow the steps in this solution to install the required dependencies.

Step 1: Install Docker and Docker Compose

Install Docker

The following steps use Ubuntu 20.04 Debian Linux distribution software as an example. The commands you run depend on the Linux distribution software that you are using. Refer to the specific Linux distribution software documentation for your configuration.

Steps

1. Install Docker by running the following commands:

```
sudo apt-get update
sudo apt-get install apt-transport-https ca-certificates curl gnupg-
agent software-properties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key
add -
sudo add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io
```

2. Verify the installation:

```
docker –version
```

3. Verify that a group named "docker" has been created on your Linux system. If necessary, create the group:

```
sudo groupadd docker
```

4. Add the user that needs to access Docker to the group:

```
sudo usermod -aG docker $(whoami)
```

5. Your changes are applied after you log out and log back in to the terminal. Alternatively, you can apply the changes immediately:

```
newgrp docker
```

Install Docker Compose

Steps

1. Install Docker Compose by running the following `sudo` commands:

```
sudo curl -L
"https://github.com/docker/compose/releases/download/1.29.2/docker-
compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose
```

2. Verify the installation:

```
docker-compose –version
```

Step 2: Prepare the Docker image

Steps

1. Copy the `cvo_gcp_flexcache.zip` folder to the Linux VM that you want to use to deploy Cloud Volumes ONTAP:

```
scp -i ~/private-key.pem -r cvo_gcp_flexcache.zip
gcpuser@IP_ADDRESS_OF_VM:LOCATION_TO_BE_COPIED
```

   - `private-key.pem` is your private key file for login without a password.
   - `gcpuser` is the VM username.
   - `IP_ADDRESS_OF_VM` is the VM IP address.
   - `LOCATION_TO_BE_COPIED` is the location where the folder will be copied.

2. Extract the `cvo_gcp_flexcache.zip` folder. You can extract the folder in the current directory or in a custom location.

   To extract the folder in the current directory, run:

   ```
   unzip cvo_gcp_flexcache.zip
   ```

   To extract the folder in a custom location, run:

   ```
   unzip cvo_gcp_flexcache.zip -d ~/<your_folder_name>
   ```

3. After you extract the content, run the following command to view the files:

   ```
    ls -la
   ```

   You should see a list of files, similar to the following example:

   ```
       total 32
       drwxr-xr-x   8 user  staff   256 Mar 23 12:26 .
       drwxr-xr-x   6 user  staff   192 Mar 22 08:04 ..
       -rw-r--r--   1 user  staff   324 Apr 12 21:37 .env
       -rw-r--r--   1 user  staff  1449 Mar 23 13:19 Dockerfile
       drwxr-xr-x  15 user  staff   480 Mar 23 13:19 cvo_gcp_source_code
       drwxr-xr-x   4 user  staff   128 Apr 27 13:43 cvo_gcp_variables
       -rw-r--r--   1 user  staff   996 Mar 24 04:06 docker-compose-
   deploy.yml
       -rw-r--r--   1 user  staff  1041 Mar 24 04:06 docker-compose-
   destroy.yml
   ```

4. Locate the `cvo_gcp_flexcache_ubuntu_image.tar` file. This contains the Docker image required to deploy Cloud Volumes ONTAP for Google Cloud.
5. Untar the file:

   ```
   docker load -i cvo_gcp_flexcache_ubuntu_image.tar
   ```

6. Wait a few minutes for the Docker image to load, and then validate that the Docker image loaded successfully:

   ```
   docker images
   ```

   You should see a Docker image named `cvo_gcp_flexcache_ubuntu_image` with the `latest` tag, as

shown in the following example:

```
REPOSITORY                                TAG         IMAGE ID       CREATED
SIZE
    cvo_gcp_flexcache_ubuntu_image      latest     18db15a4d59c    2 weeks
ago   1.14GB
```

> You can change the Docker image name if required. If you change the Docker image name, make sure to update the Docker image name in the `docker-compose-deploy` and `docker-compose-destroy` files.

Step 3: Update the JSON file

At this stage, you must update the `cxo-automation-gcp.json` file with a service account key to authenticate the Google Cloud provider.

1. Create a service account with permissions to deploy Cloud Volumes ONTAP and the BlueXP Connector. Learn more about creating service accounts.
2. Download the key file for the account and update the `cxo-automation-gcp.json` file with the key file information. The `cxo-automation-gcp.json` file is located in the `cvo_gcp_variables` folder.

   Example

   ```
   {
     "type": "service_account",
     "project_id": "",
     "private_key_id": "",
     "private_key": "",
     "client_email": "",
     "client_id": "",
     "auth_uri": "https://accounts.google.com/o/oauth2/auth",
     "token_uri": "https://oauth2.googleapis.com/token",
     "auth_provider_x509_cert_url":
   "https://www.googleapis.com/oauth2/v1/certs",
     "client_x509_cert_url": "",
     "universe_domain": "googleapis.com"
   }
   ```

   The file format must be exactly as shown above.

Step 4: Subscribe to BlueXP

You can subscribe to NetApp BlueXP in the Google Cloud Marketplace.

Steps

1. Navigate to the Google Cloud console and select **Subscribe to NetApp BlueXP**.

2. Configure the BlueXP portal to import the SaaS subscription to BlueXP.

   You can configure this directly from the Google Cloud Platform. You will be redirected to the BlueXP portal to confirm the configuration.

3. Confirm the configuration in the BlueXP portal by selecting **Save**.

For more information, see Manage Google Cloud credentials and subscriptions for BlueXP.

Step 5: Enable required Google Cloud APIs

You must enable the following Google Cloud APIs in your project to deploy Cloud Volumes ONTAP and the Connector.

- Cloud Deployment Manager V2 API
- Cloud Logging API
- Cloud Resource Manager API
- Compute Engine API
- Identity and Access Management (IAM) API

Learn more about enabling APIs

Step 6: Create an external volume

You should create an external volume to keep the Terraform state files and other important files persistent. You must make sure that the files are available for Terraform to run the workflow and deployments.

Steps

1. Create an external volume outside of Docker Compose:

   ```
   docker volume create <volume_name>
   ```

   Example:

   ```
   docker volume create cvo_gcp_volume_dst
   ```

2. Use one of the following options:

   a. Add an external volume path to the `.env` environment file.

      You must follow the exact format shown below.

      Format:

      `PERSISTENT_VOL=path/to/external/volume:/cvo_gcp`

      Example:
      `PERSISTENT_VOL=cvo_gcp_volume_dst:/cvo_gcp`

b. Add NFS shares as an external volume.

   Make sure that the Docker container can communicate with the NFS shares and that the correct permissions, such as read/write, are configured.

   i. Add the NFS shares path as the path to the external volume in the Docker Compose file, as shown below:
   Format:

   ```
   PERSISTENT_VOL=path/to/nfs/volume:/cvo_gcp
   ```

   Example:
   `PERSISTENT_VOL=nfs/mnt/document:/cvo_gcp`

3. Navigate to the `cvo_gcp_variables` folder.

   You should see the following files in the folder:

   - `terraform.tfvars`
   - `variables.tf`

4. Change the values inside the `terraform.tfvars` file according to your requirements.

   You must read the specific supporting documentation when modifying any of the variable values in the `terraform.tfvars` file. The values can vary depending on region, availability zones, and other factors supported by Cloud Volumes ONTAP for Google Cloud. This includes licenses, disk size, and VM size for single nodes and high availability (HA) pairs.

   All supporting variables for the Connector and Cloud Volumes ONTAP Terraform modules are already defined in the `variables.tf` file. You must refer to the variable names in the `variables.tf` file before adding to the `terraform.tfvars` file.

5. Depending on your requirements, you can enable or disable FlexCache and FlexClone by setting the following options to `true` or `false`.

   The following examples enable FlexCache and FlexClone:

   - `is_flexcache_required = true`
   - `is_flexclone_required = true`

Step 7: Deploy Cloud Volumes ONTAP for Google Cloud

Use the following steps to deploy Cloud Volumes ONTAP for Google Cloud.

Steps

1. From the root folder, run the following command to trigger deployment:

   ```
   docker-compose -f docker-compose-deploy.yml up -d
   ```

   Two containers are triggered, the first container deploys Cloud Volumes ONTAP and the second container sends telemetry data to AutoSupport.

The second container waits until the first container completes all of the steps successfully.

2. Monitor progress of the deployment process using the log files:

```
docker-compose -f docker-compose-deploy.yml logs -f
```

This command provides output in real time and captures the data in the following log files:
`deployment.log`

`telemetry_asup.log`

You can change the name of these log files by editing the `.env` file using the following environment variables:

`DEPLOYMENT_LOGS`

`TELEMETRY_ASUP_LOGS`

The following examples show how to change the log file names:

`DEPLOYMENT_LOGS=<your_deployment_log_filename>.log`

`TELEMETRY_ASUP_LOGS=<your_telemetry_asup_log_filename>.log`

After you finish

You can use the following steps to remove the temporary environment and clean up items that were created during the deployment process.

Steps

1. If you deployed FlexCache, set the following option in the `terraform.tfvars` file, this cleans up FlexCache volumes and removes the temporary environment that was created earlier.

   `flexcache_operation = "destroy"`

   The possible options are `deploy` and `destroy`.

2. If you deployed FlexClone, set the following option in the `terraform.tfvars` file, this cleans up FlexClone volumes and removes the temporary environment that was created earlier.

   `flexclone_operation = "destroy"`

   The possible options are `deploy` and `destroy`.