Elastifile Migration to GCP Filestore

Recommended Procedure and Best Practices

Background

In August 2019, Elastifile was acquired by Google.
Google and Elastifile announced a planned End of Support life by September 30, 2023,
Resulting in extension until the equivalent Cloud Filestore managed service tiers support the critical Elastifile legacy product's features.

Elastifile users can use the product until the EOS date and receive urgent maintenance and support.
Elastifile encourages its customers to start evaluating Cloud Filestore and plan a migration to the managed service.

Objective

This document helps Elastifile customers to perform the migration to GCP Filestore managed-service as smoothly as possible, and provides a recommended procedure including all steps, best practices and tools for completing this task.

Performing the procedure by all steps is done by the customer only.

Overview

The high level plan of a successful migration includes the following steps:

1. Sizing and configuration-
   Architect the Filestore environment based on the Elastifile setups:
   a. Define the number of shares
   b. Define and configure the size of the Filestore instance
   c. Arrange the required GCP quota for deploying the Filestore instance
   d. Define and configure the snapshot/backup policies
   e. Migrate the network ACLs
2. Configure a migrator instance
3. Perform an initial sync of the data
4. Multiple incremental synchronizations for replicating deltas
5. Last synchronization
6. Remount all clients to work against the Filestore instance

The used open-source tool for synching the data is rclone.
Customers can choose any other tool if preferred.

Known Limitations

- Extended/Posix ACLs are not supported
- Soft links are synced as links, while hard links are synced as independent files and not links
- The migration process uses the Elastifile cluster performance and resources. Recommended to use at off-hours.

Detailed Procedure

Prerequisites

Build the Filestore environment based on the existing Elastifile clusters.

1. **Number of shares-**
   While Elastifile supports multiple shares, Filestore supports a single one.
   Alternative solutions:
   - Split the Elastifile system into multiple Filestore instances
   - Consolidate multiple data containers under single Filestore instance and use directory based mounting

2. **Size-**
   Filestore offers different capacity sizes per tier:
   - Enterprise: 1TB-10TB. Increment of 250GB.
   - HighScale: 10TB-100TB. Increment of 2.5TB.

   For additional information and Filestore tiers comparison, click here.

3. **Quota-**
   Make sure you have enough quota for deploying the Filestore instances in the required region:
   - Enterprise: Enterprise capacity (GB) per region
   - HighScale: High Scale SSD capacity (GB) per region

4. **Deploy** the Filestore instances based on the decisions above

5. **Snapshot and Backup schedulers-**
   Define the required scheduler. Coming soon.

6. **Migrate the network ACLs-**
   You can use Elastifile Exports ACLs Migration Tool.

Data Migration

1. Configure a migrator instance

a. Deploy a new GCE instance with the following properties:

   i. Same region and VPC as the Elastifile system and Filestore instance
   ii. Machine type: c2-standard-8
   iii. Boot device: 512GB persistent-SSD
   iv. Operating system: debian-11-bullseye
   v. The instance should have access to Elastifile and Filestore data shares

```
gcloud compute instances create migrator --project=<PROJECT_ID>
--zone=<GCP_ZONE> --machine-type=c2-standard-8
--network-interface=network-tier=PREMIUM,subnet=<VPC_SUBNETWORK>
--create-disk=auto-delete=yes,boot=yes,device-name=migrator,image=pr
ojects/debian-cloud/global/images/debian-11-bullseye-v20220920,mode=
rw,size=512,type=projects/<PROJECT_ID>/zones/<GCP_ZONE>/diskTypes/pd
-ssd
```

b. Install rclone and nfs dependencies on the instance

```
sudo apt install nfs-common rclone -y
```

2. Create a snapshot of the migrated Elastifile data container

a. SSH to the Elastifile EMS VM and load the elfs_admin source

```
sudo su -
source elfs_admin
```

b. List the data containers and locate the ID of the one to replicate

```
elfs-cli data_container list -t
```

c. Create the snapshot for the initial sync

```
elfs-cli snapshot create --data-container-id <DC_ID> --name
initial_sync
```

3. Mount the Elastifile share

a. SSH to the migrator instance

b. Run showmount command and locate the share you would like to backup

```
showmount -e <ELASTIFLE_IP_ADDRESS>
```

c. Create a mount point

```
mkdir /mnt/src
```

d. Mount the snapshot share using the below mount options
Note: For more details about the used mount options, refer here.

```
mount -o noatime,nodiratime,actimeo=120,nconnect=3
<ELASTIFLE_IP_ADDRESS>:/<SHARE_NAME>/root /mnt/src/
```

e. Verify you can access the content of the snapshot

```
ls -l /mnt/src/.snapshot/initial_sync
```

4. Mount the Filestore share

a. SSH to the migrator instance

b. Run showmount command and locate the share you would like to migrate to

```
showmount -e <FILESTORE_IP_ADDRESS>
```

c. Create a mount point

```
mkdir /mnt/dst
```

d. Mount the Filestore share using the below mount options

```
mount -o noatime,nodiratime,actimeo=120,nconnect=3
<FILESTORE_IP_ADDRESS>:/<SHARE_NAME> /mnt/dst/
```

5. Perform Initial Sync

a. From the migrator instance, sync the data to the Filestore target using the created snapshot
Note: For more details about the used rclone flags, refer here.

```
rclone sync /mnt/src/.snapshot/initial_sync/ /mnt/dst --checkers 256
--transfers 256 -v -P -l --create-empty-src-dirs --log-file=/tmp/sync.log
```

6. Perform Incremental syncs

a. Create a newer snapshot on the Elastifile system

```
elfs-cli snapshot create --data-container-id <DC_ID> --name
second_sync
```

b. From the migrator instance, verify you can access the content of the snapshot

```
ls -l /mnt/src/.snapshot/second_sync
```

c. From the migrator instance, sync the changes

```
rclone sync /mnt/src/.snapshot/second_sync/ /mnt/dst --checkers 256
--transfers 256 -v -P -l --create-empty-src-dirs
--log-file=/tmp/sync.log
```

d. Repeat the process with new snapshots name until the sync time is low and fits the agreed downtime window

7. Perform Final Sync (Downtime Start)

   a. Move the share into a read only mode so no new writes can be written

   b. Create the final snapshot on the Elastifile system

```
elfs-cli snapshot create --data-container-id <DC_ID> --name final_sync
```

   c. From the migrator instance, verify you can access the content of the snapshot

```
ls -l /mnt/src/.snapshot/final_sync
```

   d. From the migrator instance, sync the changes

```
rclone sync /mnt/src/.snapshot/final_sync/ /mnt/dst --checkers 256
--transfers 256 -v -P -l --create-empty-src-dirs
--log-file=/tmp/sync.log
```

8. (Optional) Test the New Filestore destination

   Take a single client or application server to mount and work against the Filestore share to make sure the data is valid and the processes can run successfully.

9. Start using the Filestore instance (Downtime Finish)

   Remount all users to work against the Filestore share

Appendixes

Filestore Tiers Comparison

Filestore Tier	Basic	High Scale	Enterprise
Capacity	1 TB - 64 TB	10 TB - 100 TB	1-10TB
Scale	Scale-up	Scale-out	Scale-out
Grow and shrink	Grow only	Grow and **Shrink**	Grow and **Shrink**
Throughput (read)	SSD 1.2 GB/s	≈260 MiB/s/TB Max 26 GiB/sec	≈120 MiB/s/TB Max 1.2 GiB/sec
Max IOPS	SSD 60,000	920K	120K
Protocols	NFSv3, NLM (locking)	NFSv3, NLM (locking)	NFSv3, NLM (locking)
Availability	Zonal	Zonal	Regional
SLA	99.9%	99.9%	99.99%
Data protection	Backups	—	Snapshots
IP-range Access Controls	RO/RW, Root-Squash	RO/RW, Root-Squash	RO/RW, Root-Squash

Mount Options

The following mount options are used:

- noatime - Increase performance by not updating file access time on source and destination file systems.
- nodiratime - Increase performance by not updating directory access time on source and destination file systems.
- actimeo=120 - attribute caching for increased performance.
- nconnect=<INT> - using multiple tcp connections for better read/write performance.

Rclone Flags

The following rclone gflags are used:

`--checkers`	Number of checkers to run in parallel (default 8)
`--transfers`	Number of file transfers to run in parallel (default 4)
`-M, --metadata`	If set, preserve metadata when copying objects
`-v, --verbose`	Print lots more stuff (repeat for more)
`-P, --progress`	Show progress during transfer
`-l, --links`	Translate symlinks to/from regular files with a '.rclonelink' extension
`--create-empty-src-dirs`	Create empty source dirs on destination after sync
`--log-file`	Log everything to this file

Note: For a full list of the flags, run `rclone help flags` on the migrator instance