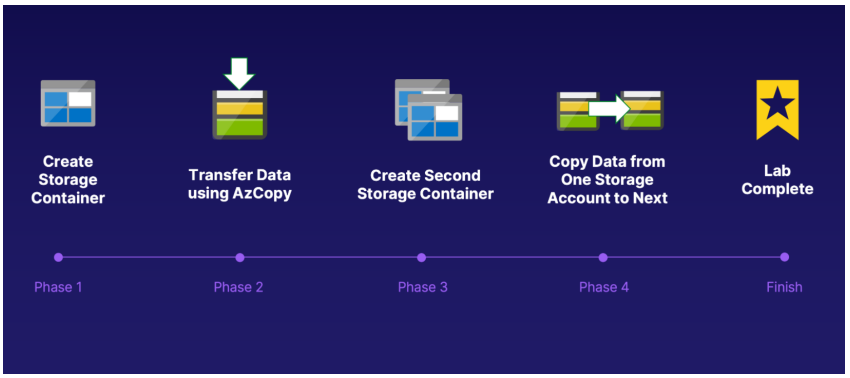
Use AzCopy to Transfer Data from One Storage Account to Another

Introduction

In the scenario for this hands-on lab, you are a storage administrator. The Data Science department needs you to migrate the data from one of their storage accounts to a new storage account with hierarchical namespaces.
They are currently using Azure HD Insight, but need the big data analytics capabilities that come with Gen 2 Data Lake Storage. Use the AzCopy tool within Azure Cloud Shell to perform the migration.



Solution

Log in to the Azure Portal using the credentials provided on the lab instructions page.

- In the Azure Portal, click the Cloud Shell icon (>_) in the upper right.
- Select PowerShell.
- Click Show advanced settings.
- For Storage account, select Create new and give it a globally unique name (e.g., "cloudshell" with a series of numbers at the end).
- For File share, select Create new and give it a name of "fileshare".
- Click Create storage.

Create a Storage Container

1. In Azure Cloud Shell, switch to bash:
   ```
   PS /home/cloud> bash
   ```

2. Change to the cloud drive directory:
   cd clouddrive

3. Run a for loop to generate data:
   cloud@azure:~$ for i in {000..100}
   > do
   >   echo hello > "File${i}.txt"
   > done

4. List the files to verify that the loop ran successfully:
   ls

5. List the storage accounts:
   az storage account list --query '[].{name:name, resourceGroup:resourceGroup}' -o json

6. List the storage account key for the storage account starting in stor1:
   az storage account keys list -g "<RESOURCE_GROUP_ID>" -n "<STORAGE_ACCOUNT_NAME>"

7. Set environment variables for the storage account name and key:
AZURE_STORAGE_ACCOUNT1="<STOR1_ACCOUNT_NAME>"
AZURE_STORAGE1_KEY="<KEY1_VALUE>"

8. List the storage account key for the storage account starting in stor2:
   az storage account keys list -g "<RESOURCE_GROUP_ID>" -n "<STORAGE_ACCOUNT_NAME>"

9. Set environment variables for the storage account name and key:
   AZURE_STORAGE_ACCOUNT2="<STOR2_ACCOUNT_NAME>"
   AZURE_STORAGE2_KEY="<KEY2_VALUE>"

10. Create the storage container:
    az storage container create -n "container1" --account-name $AZURE_STORAGE_ACCOUNT1 --account-key $AZURE_STORAGE1_KEY

Use AZCOPY to Transfer Data to Container

1. Generate a SAS token:

```
az storage container generate-sas --account-name
$AZURE_STORAGE_ACCOUNT1 \
--name "container1" --permissions acdlrw --expiry <DATE> --auth-mode login \
--as-user
```

Note: The date must be within 7 days of the date that the SAS token is generated.

2. Copy the token.

3. List the storage account endpoints:

```
az storage account list --query [].primaryEndpoints.blob -o json
```

4. Copy the files to the storage endpoint beginning with stor1 using AzCopy:

```
azcopy copy './' '<STOR1_ENDPOINT>/container1?<SAS_TOKEN>' --recursive=true
```

5. In the Azure Portal, verify that the files were added to the storage account successfully.

Create a Second Storage Container

1. Create the second storage container:

```
az storage container create -n "container2" --account-name
$AZURE_STORAGE_ACCOUNT2 --account-key $AZURE_STORAGE2_KEY
```

2. Verify that the container was created successfully:

```
az storage container list --account-name $AZURE_STORAGE_ACCOUNT2 \
--account-key $AZURE_STORAGE2_KEY
```

Copy Data between Storage Accounts using

1. Generate a SAS token for container2:

```
az storage container generate-sas --account-name
$AZURE_STORAGE_ACCOUNT2 \
--name "container2" --permissions acdlrw --expiry <DATE> --auth-mode login \
--as-user
```

Note: The date must be within 7 days of the date that the SAS token is generated.

2. Copy the token.

3. List the storage account endpoints:

```
az storage account list --query [].primaryEndpoints.blob -o json
```

4. Use AzCopy to transfer the files from container1 to container2:

```
azcopy copy '<STOR1_ENDPOINT>/container1/?<CONTAINER1_SAS_TOKEN>'
'<STOR2_ENDPOINT>/container2?<CONTAINER2_SAS_TOKEN>' --recursive=true
```

In the Azure Portal, verify that the files were transferred from container1 to container2.