

Upgrade Milvus from 2.3 to 2.4.4 without any data loss

- Access the first Standalone Cluster and create a pod using `vector-db` namespace.

```
1 apiVersion: v1
2 kind: Pod
3 metadata:
4   name: milvus-backup-pod
5   namespace: vector-db
6 spec:
7   containers:
8   - name: python-container
9     image: python:3.8
10    command: [ "sleep", "infinity" ]
```

- Exec into the pod using

```
1 kubectl exec -it milvus-backup-pod -n vector-db -- bash
```

- Install `milvus-cli` and `vim`

```
1 apt-get update
2 pip install milvus-cli
3 apt-get install vim
```

- Create a new file to connect to the Milvus service.

```
1 vim connect-milvus.py
```

- Paste this script in the new Python file.

```
1 from pymilvus import connections, utility
2
3 # Connect to Milvus
4 connections.connect(
5     alias="default",
6     host='prod-milvus-milvus.vector-db.svc.cluster.local',
7     port='19530'
8 )
9
10 # Check connection by listing collections
11 try:
12     collections = utility.list_collections()
13     print("Successfully connected to Milvus. Collections:")
14     for collection in collections:
15         print(f"- {collection}")
16 except Exception as e:
17     print(f"Failed to connect to Milvus: {e}")
```

- Run the Python script using

```
1 python3 connect-milvus.py
```

- Once you run the script, the pod gets connected to the Milvus service and it will list all the collections which are available in the Milvus.
- Now we need to clone the Milvus Backup tool repository to take the backup of Milvus data in Minio.

```
1 apt-get update
```

```

2 apt install git -y
3 apt-get install wget -y
4 wget https://go.dev/dl/go1.21.4.linux-amd64.tar.gz -O go.tar.gz
5 tar -xzf go.tar.gz -C /usr/local
6 echo export PATH=$HOME/go/bin:/usr/local/go/bin:$PATH >> ~/.profile
7 source ~/.profile
8 go version
9
10 git clone https://github.com/zilliztech/milvus-backup.git
11 cd milvus-backup
12 go get
13 go build

```

- Edit the backup configuration file

```

1 cd configs/
2 vim backup.yaml

```

```

1 # Configures the system log output.
2 log:
3   level: info # Only supports debug, info, warn, error, panic, or fatal. Default 'info'.
4   console: true # whether print log to console
5   file:
6     rootPath: "logs/backup.log"
7
8 http:
9   simpleResponse: true
10
11 # milvus proxy address, compatible to milvus.yaml
12 milvus:
13   address: prod-milvus-milvus.vector-db.svc.cluster.local
14   port: 19530
15   authorizationEnabled: false
16   # tls mode values [0, 1, 2]
17   # 0 is close, 1 is one-way authentication, 2 is two-way authentication.
18   tlsMode: 0
19   user: "username"
20   password: "password"
21
22 # Related configuration of minio, which is responsible for data persistence for Milvus.
23 minio:
24   # cloudProvider: "minio" # deprecated use storageType instead
25   storageType: "minio" # support storage type: local, minio, s3, aws, gcp, ali(aliyun), azure, tc(tencent)
26
27   address: prod-milvus-minio.vector-db.svc.cluster.local # Address of MinIO/S3
28   port: 9000 # Port of MinIO/S3
29   accessKeyID: minioadmin # accessKeyID of MinIO/S3
30   secretAccessKey: minioadmin # MinIO/S3 encryption string
31   useSSL: false # Access to MinIO/S3 with SSL
32   useIAM: false
33   iamEndpoint: ""
34
35   bucketName: "prod-milvus" # Milvus Bucket name in MinIO/S3, make it the same as your milvus instance
36   rootPath: "files" # Milvus storage root path in MinIO/S3, make it the same as your milvus instance
37
38   # only for azure
39   backupAccessKeyID: # accessKeyID of MinIO/S3
40   backupSecretAccessKey: # MinIO/S3 encryption string
41

```

```

42 backupBucketName: "prod-milvus-new" # Bucket name to store backup data. Backup data will store to
    backupBucketName/backupRootPath
43 backupRootPath: "backup" # Rootpath to store backup data. Backup data will store to
    backupBucketName/backupRootPath
44
45 backup:
46   maxSegmentGroupSize: 6G
47
48   parallelism:
49     # collection level parallelism to backup
50     backupCollection: 4
51     # thread pool to copy data. reduce it if blocks your storage's network bandwidth
52     copydata: 128
53     # Collection level parallelism to restore
54     restoreCollection: 2
55
56   # keep temporary files during restore, only use to debug
57   keepTempFiles: false
58
59   # Pause GC during backup through Milvus Http API.
60   gcPause:
61     enable: false
62     seconds: 7200
63   address: http://prod-milvus-milvus.vector-db.svc.cluster.local:9091

```

- Take the backup of Milvus Collection in Minio using

```

1 cd ..
2 ./milvus-backup create -n prod_milvus_new

```

- Backup of Milvus collection in Minio is successfull.
- Now we will verify in Minio whether the collections are present or not.
- Download all the required packages (minio-client) to connect to Minio through pod.

```

1 apt-get update
2 apt install wget -y
3 wget https://dl.min.io/client/mc/release/linux-amd64/mc
4 chmod +x mc
5 mv mc /usr/local/bin/

```

- Connect to minio client using

```
1 mc alias set myminio http://prod-milvus-minio.vector-db.svc.cluster.local:9000 minioadmin minioadmin
```

- List the minio buckets

```
1 mc ls myminio
```

- If minio bucket called `prod-milvus-new` is present, then the Data has been successfully backed up in Minio.
- Now we will copy the collections from Minio to S3 buckets.
- In AWS s3 service in `us-east-2` region create an empty bucket with name `prod-milvus-new`
- Now connect to the AWS S3 service inside pod using

```
1 mc alias set s3 https://s3.us-east-2.amazonaws.com AWS_Access_Key_ID AWS_Access_Secret_Key
```

- Copy the Minio Bucket (`prod-milvus-new`) data to AWS S3 (`prod-milvus-new`) bucket.

```
1 mc mirror myminio/prod-milvus-new s3/prod-milvus-new
```

- Verify in AWS S3 service UI whether the data has been copied or not in the bucket.

- Delete the `vector-db` namespace and all its resources.

```
1 kubectl delete ns vector-db
```

- Clone the Katonic-Installer using

```
1 git clone https://github.com/katonic-dev/Katonic-Installer.git
```

- Go into the milvus directory

```
1 cd Katonic-Installer/platform-deployment/yamls/milvus
```

- Create a new `vector-db` namespace using

```
1 kubectl create ns vector-db
```

- In `milvus-standalone.yaml` file edit the image to `milvusdb/milvus:v2.4.4` and size to `128Gi`.
- Apply this file using

```
1 kubectl create -n vector-db -f milvus-standalone.yaml
```

- Check if all the pods are in running state using

```
1 kubectl get pods -n vector-db
```

- Describe the Milvus Pod and check the version of Milvus whether it is correctly set to v2.4.4

- Create a pod using `vector-db` namespace.

```
1 apiVersion: v1
2 kind: Pod
3 metadata:
4   name: milvus-backup-restore-pod
5   namespace: vector-db
6 spec:
7   containers:
8   - name: ubuntu-container
9     image: ubuntu:20.04
10    command: [ "sleep", "infinity" ]
11    securityContext:
12      runAsUser: 0
13      runAsGroup: 0
```

- Exec into the pod using

```
1 kubectl exec -it milvus-backup-restore-pod -n vector-db -- bash
```

- Install `milvus-cli` and `vim`

```
1 apt-get update
2 apt-get install pip -y
3 pip install milvus-cli
4 apt-get install vim
```

- Create a new file to connect to the Milvus service.

```
1 vim connect-milvus.py
```

- Paste this script in the new Python file.

```

1 from pymilvus import connections, utility
2
3 # Connect to Milvus
4 connections.connect(
5     alias="default",
6     host='prod-milvus-milvus.vector-db.svc.cluster.local',
7     port='19530'
8 )
9
10 # Check connection by listing collections
11 try:
12     collections = utility.list_collections()
13     print("Successfully connected to Milvus. Collections:")
14     for collection in collections:
15         print(f"- {collection}")
16 except Exception as e:
17     print(f"Failed to connect to Milvus: {e}")

```

- Run the Python script using

```
1 python3 connect-milvus.py
```

- Once you run the script, the pod gets connected to the Milvus service and it will list all the collections which are available in the Milvus.
- Download all the required packages (minio-client) to connect to Minio through pod.

```

1 apt-get update
2 apt install wget -y
3 wget https://dl.min.io/client/mc/release/linux-amd64/mc
4 chmod +x mc
5 mv mc /usr/local/bin/

```

- Connect to minio client using

```
1 mc alias set myminio http://prod-milvus-minio.vector-db.svc.cluster.local:9000 minioadmin minioadmin
```

- Create an empty minio bucket using

```
1 mc mb myminio/prod-milvus-new
```

- Connect to AWS S3 service from pod using

```
1 mc alias set s3 https://s3.us-east-2.amazonaws.com AWS_Access_Key_ID AWS_Access_Secret_Key
```

- Copy the S3 bucket (prod-milvus-new) data to the newly created minio bucket (prod-milvus-new) in standalone cluster using

```
1 mc mirror s3/prod-milvus-new myminio/prod-milvus-new
```

- Now we need to clone the Milvus Backup tool repository to restore the Milvus Data in same milvus standalone cluster but with upgraded milvus (v2.4.4)

```

1 apt-get update
2 apt install git -y
3 apt-get install wget -y
4 wget https://go.dev/dl/go1.21.4.linux-amd64.tar.gz -O go.tar.gz
5 tar -xzf go.tar.gz -C /usr/local
6 echo export PATH=$HOME/go/bin:/usr/local/go/bin:$PATH >> ~/.profile
7 source ~/.profile
8 go version

```

```
9
10 git clone https://github.com/zilliztech/milvus-backup.git
11 cd milvus-backup
12 go get
13 go build
```

- Edit the backup configuration file

```
1 cd configs/
2 vim backup.yaml
```

```
1 # Configures the system log output.
2 log:
3   level: info # Only supports debug, info, warn, error, panic, or fatal. Default 'info'.
4   console: true # whether print log to console
5   file:
6     rootPath: "logs/backup.log"
7
8 http:
9   simpleResponse: true
10
11 # milvus proxy address, compatible to milvus.yaml
12 milvus:
13   address: prod-milvus-milvus.vector-db.svc.cluster.local
14   port: 19530
15   authorizationEnabled: false
16   # tls mode values [0, 1, 2]
17   # 0 is close, 1 is one-way authentication, 2 is two-way authentication.
18   tlsMode: 0
19   user: "username"
20   password: "password"
21
22 # Related configuration of minio, which is responsible for data persistence for Milvus.
23 minio:
24   # cloudProvider: "minio" # deprecated use storageType instead
25   storageType: "minio" # support storage type: local, minio, s3, aws, gcp, ali(aliyun), azure, tc(tencent)
26
27   address: prod-milvus-minio.vector-db.svc.cluster.local # Address of MinIO/S3
28   port: 9000 # Port of MinIO/S3
29   accessKeyID: minioadmin # accessKeyID of MinIO/S3
30   secretAccessKey: minioadmin # MinIO/S3 encryption string
31   useSSL: false # Access to MinIO/S3 with SSL
32   useIAM: false
33   iamEndpoint: ""
34
35   bucketName: "prod-milvus" # Milvus Bucket name in MinIO/S3, make it the same as your milvus instance
36   rootPath: "files" # Milvus storage root path in MinIO/S3, make it the same as your milvus instance
37
38   # only for azure
39   backupAccessKeyID: # accessKeyID of MinIO/S3
40   backupSecretAccessKey: # MinIO/S3 encryption string
41
42   backupBucketName: "prod-milvus-new" # Bucket name to store backup data. Backup data will store to
43   backupBucketName/backupRootPath
44   backupRootPath: "backup" # Rootpath to store backup data. Backup data will store to
45   backupBucketName/backupRootPath
46
47 backup:
48   maxSegmentGroupSize: 6G
```

```
47
48 parallelism:
49     # collection level parallelism to backup
50     backupCollection: 4
51     # thread pool to copy data. reduce it if blocks your storage's network bandwidth
52     copydata: 128
53     # Collection level parallelism to restore
54     restoreCollection: 2
55
56 # keep temporary files during restore, only use to debug
57 keepTempFiles: false
58
59 # Pause GC during backup through Milvus Http API.
60 gcPause:
61     enable: false
62     seconds: 7200
63     address: http://prod-milvus-milvus.vector-db.svc.cluster.local:9091
```

- Now restore the Milvus Data using

```
1 cd ..
2 ./milvus-backup restore -n prod_milvus_new
```

- Restoration of data is successful in the Cluster with the upgraded milvus without any data loss.
- To verify whether the same collections exist or not in the cluster after restoring data in the new and updated Milvus, run the python script

```
1 python3 connect-milvus.py
```

- We can also check from the UI side whether Milvus has been upgraded or not without any data loss.