

# Volume Backups Openstack Blueprint

- Summary
- User Stories
- Design
  - Backup Create
  - Backup Restore
  - Backup Delete
  - volume\_backups table design
  - Volume and Backup State Transitions
    - Create Backup
    - Restore Backup
    - Delete Backup
  - Swift Object Naming
  - Metadata File
  - Compression
- Proposed API
  - Create backup
    - Operation
    - Request Parameters
    - Request JSON
    - Response JSON
    - Description
  - Delete backup
    - Operation
    - Request Parameters
    - Description
  - List backups
    - Operation
    - Request Parameters
    - Response JSON
    - Description
  - List details for all backups
    - Operation
    - Request Parameters
    - Response JSON
    - Description
  - Show backup details
    - Operation
    - Request Parameters
    - Response JSON
    - Description
  - Restore backup
    - Operation
    - Request Parameters
    - Request JSON
    - Response JSON
    - Description
- Future Features
  - Differential Backups
  - Scheduled Backups
  - Configurable Compression
  - Encrypted Backups

## Summary

This blueprint adds support for backing up user volumes to Swift. This backup service will allow the user to create, restore and delete backups as well as listing backups and showing the details of a specific backup.

The term backup as used in this blueprint refers to a copy of the original volume which is stored on Swift. This backup is independent of the original volume and may be used for archival and disaster recover purposes. This is distinct from a *snapshot* of the volume which may be generated using techniques such as copy-on-write and have dependencies on the original volume.

---

## User Stories

User wishes to backup existing and new volumes for archival or disaster recovery purposes.

---

# Design

## Backup Create

- The user requests a backup of an existing volume.
- A new entry is created in the volume\_backups table.
- The status of the volume and backup are modified in the volumes and volume\_backups tables respectively to indicate a create is in progress.
- The volume is broken into chunks.
- Each chunk is compressed - if compression is enabled - and written as an object to Swift.
- A metadata file containing details of the backup is also written to Swift

## Backup Restore

- The user requests a volume backup is restored.
- If the user does not specify a volume to restore to a new volume is created.
- The status of the volume and backup are modified in the volumes and volume\_backups tables respectively to indicate a restore is in progress.
- Each object in the backup is read from Swift, decompressed - if required - and written to the volume.

## Backup Delete

- The user requests a volume backup is deleted.
- The status of the backup is modified in the volume\_backups table to indicate a delete is in progress.
- Each object in the backup is deleted from Swift
- The backup entry in the volume\_backups table is marked deleted

## volume\_backups table design

A new table called *volume\_backups* needs to be created in the nova database.

created_at	datetime
updated_at	datetime
deleted_at	datetime
deleted	tinyint(1)
id	int(11)
volume_id	int(11)
user_id	varchar(255)
project_id	varchar(255)
backup_host	varchar(255)
availability_zone	varchar(255)
display_name	varchar(255)
display_description	varchar(255)
container	varchar(255)
status	varchar(255)
fail_reason	varchar(255)
swift_prefix	varchar(255)
size	int(11)
object_count	int(11)
backup_start	datetime
backup_end	datetime
backup_heartbeat	datetime

## Volume and Backup State Transitions

### Create Backup

#### Volume States

available ----- (create backup request received) -----> backing-up ----- (create backup completed successfully) -----> available  
backing-up ----- (error during backup create) -----> available

#### Backup States

creating ----- (create backup completed successfully) -----> available  
creating ----- (error during backup create) -----> error

### Restore Backup

#### Volume States

available ----- (restore backup request received) -----> restoring-backup ----- (restore backup completed successfully) -----> available  
restoring-backup ----- (error during backup restore) -----> error

#### Backup States

available ----- (restore backup request received) -----> restoring ----- (restore backup completed successfully) -----> available  
restoring ----- (error during backup restore but backup restore can be re-tried) -----> available  
restoring ----- (error during backup restore, backup restore cannot be re-tried) -----> error

### Delete Backup

#### Volume States

No volume state changes during backup deletion

#### Backup States

available ----- (delete backup request received) -----> deleting  
available ----- (delete backup request received) -----> deleting ---- (error during backup deletion) -----> error

error ----- (delete backup request received) -----> deleting  
error ----- (delete backup request received) -----> deleting ---- (error during backup deletion) -----> error

### Swift Object Naming

Volume backup containers within Swift will have the following structure

```
<Container_Name>/<Object_Prefix>-<Object_ID>

<Container_Name>: Container name specified by the user. If no container name is specified a default
value will be used
<Object_Prefix>: volume_<volume_id>/<timestamp>/loc_<vendor_location_tag>_backup_<id>
<Object_ID>: Incrementing object count
```

### Metadata File

A metadata file will be written in JSON format to Swift as part of each volume backup.  
The metadata file will contain:

- A volume backup revision number
- The backup ID and the ID of the volume being backed-up
- The backup name and description if provided
- A created-on time-stamp
- The name of each object in the backup
- The compression algorithm used - if any - on each object in the backup
- The SHA1 digest of each object in the backup. The hash algorithm used may be user configurable.
- The uncompressed object size in bytes
- The object offset in bytes

Sample metadata file

```
{
  "revision": "1.0.0",
  "backup_id": "358",
```

```
"backup_name": "nightly001",
"backup_description": "production server nightly backup 12-Nov-2012",
"created_on": "20121112 13:50:21",
"volume_id": "16740"
"objects": [
  {
    "volume_16740/20121112135021/loc_az1_backup_358-00001": {
      "compression": "gzip",
      "sha1": "d314e48c959b978ece69549eb7b580859e3ec24a",
      "length": "52428800",
      "offset": "0"
    }
  },
  {
    "volume_16740/20121112135021/loc_az1_backup_358-00002": {
      "compression": "bzip2",
      "sha1": "a0965b7b3b048cc084355fe3409a14d8ea2c0d91",
      "length": "52428800",
      "offset": "52428800"
    }
  },
  {
    "volume_16740/20121112135021/loc_az1_backup_358-00003": {
      "compression": "bzip2",
      "sha1": "2c632d7595bad9506b3392e8b7f96a6331ff3744",
      "length": "52428800",
      "offset": "104857600"
    }
  },
  {
    "volume_16740/20121112135021/loc_az1_backup_358-00004": {
      "compression": "bzip2",
      "sha1": "80b13d7a3ba2da5cb0d844ecb64d7d26610d5c4f",
      "length": "52428800",
      "offset": "157286400"
    }
  },
  {
    "volume_16740/20121112135021/loc_az1_backup_358-00005": {
      "compression": "gzip",
      "sha1": "c16fa690e44eb78b6a4a5ea2b101b84fa2611e40",
      "length": "52428800",
      "offset": "209715200"
    }
  },
  {
    "volume_16740/20121112135021/loc_az1_backup_358-00006": {
      "compression": "gzip",
      "sha1": "dc1d793666e8b7971a6bb54f77c2f1c7e0fb6ad2",
      "length": "52428800",
      "offset": "262144000"
    }
  },
  {
    "volume_16740/20121112135021/loc_az1_backup_358-00007": {
      "compression": "none",
      "sha1": "31e663eb378ddleedc7494ad66f28e353d0e8732",
      "length": "52428800",
      "offset": "314572800"
    }
  },
  {
    "volume_16740/20121112135021/loc_az1_backup_358-00008": {
      "compression": "bzip2",
      "sha1": "4a05e8b620449cb6751175f5af6a9ac432ce16bd",
      "length": "52428800",
      "offset": "367001600"
    }
  },
  {
  }
```

```
"volume_16740/20121112135021/loc_az1_backup_358-00009": {
  "compression": "bzip2",
  "shal": "6b8007bed055402a7946cb222603dd90aaaf6820",
  "length": "52428800",
  "offset": "419430400"
}
},
{
  "volume_16740/20121112135021/loc_az1_backup_358-00010": {
    "compression": "gzip",
    "shal": "63b3ae2422ed777651414f0e2a0e5b3e62490f85",
    "length": "52428800",
    "offset": "471859200"
  }
},
{
  "volume_16740/20121112135021/loc_az1_backup_358-00011": {
    "compression": "bzip2",
    "shal": "66f2898fbbad48196080321a5ee35e99f25fba31",
    "length": "52428800",
    "offset": "524288000"
  }
},
{
  "volume_16740/20121112135021/loc_az1_backup_358-00012": {
    "compression": "bzip2",
    "shal": "66f2898fbbad48196080321a5ee35e99f25fb678",
    "length": "52428800",
    "offset": "576716800"
  }
},
{
  "volume_16740/20121112135021/loc_az1_backup_358-00013": {
    "compression": "bzip2",
    "shal": "66f2898fbbad48196080321a5ee31239f25fba31",
    "length": "52428800",
    "offset": "629145600"
  }
},
{
  "volume_16740/20121112135021/loc_az1_backup_358-00014": {
    "compression": "bzip2",
    "shal": "66f2898fbbad4819abc0321a5ee35e99f25fba31",
    "length": "52428800",
    "offset": "681574400"
  }
},
{
  "volume_16740/20121112135021/loc_az1_backup_358-00015": {
    "compression": "bzip2",
    "shal": "66f2898fbbad48196080321a5eedef99f25fba31",
    "length": "52428800",
    "offset": "734003200"
  }
}
```

```
    },  
  ],  
}
```

The metadata file will have the following name structure:

```
volume_<volume_id>/<timestamp>/loc_<vendor_location_tag>_backup_<id>_metadata
```

```
For example: volume_17040/20121112135021/az_az1_backup_358_metadata
```

## Compression

- Enabling of compression during backup creation is controlled by a flag in the nova.conf file. It is not user-specified in the API.
- The restore operation will use the *compression* field in the metadata file to determine if decompression is required when restoring a backup

---

## Proposed API

### Create backup

#### Operation

Verb	URI	Description
POST	v1.1/{tenant_id}/volume-backups	Creates a volume backup.

#### Request Parameters

Parameter	Description
tenant_id	The unique identifier of the tenant or account.
backup	A partial representation of a backup that is used to create a backup.

#### Request JSON

```
{  
  "backup": {  
    "display_name": "backup-001",  
    "display_description": "Nightly Backup 03-Sep-2012",  
    "volume_id": "9",  
    "container": "nightlybackups",  
  }  
}
```

Name	Description	Notes
display_name	User defined name for the backup.	Optional
display_description	User defined description for the backup.	Optional
volume_id	The identifier of the volume to be backed up.	Required
container	The identifier of the container to which the volume is to be backed up.	Optional

#### Response JSON

```

{
  "backup": {
    "id": "1",
    "display_name": "backup-001",
    "display_description": "Nightly Backup 03-Sep-2012",
    "volume_id": "9",
    "container": "nightlybackups",
    "status": "creating",
    "backup_start": "2012-09-03T22:00:00Z",
  }
}

```

Name	Description
id	The unique identifier of the backup.
display_name	User defined name for the backup.
display_description	User defined description for the backup.
volume_id	The identifier of the volume to be backed up.
container	The identifier of the container to which the volume is to be backed up.
status	The status of the backup operation.
backup_start	The date and time at which the backup operation started.

### Description

Backup a volume to Swift. This operation is asynchronous. You must list backups repeatedly to determine whether the backup was created. If the container is not specified a default container will be used. If a container does not exist the service will create it.

### Delete backup

#### Operation

Verb	URI	Description
DELETE	v1.1/{tenant_id}/volume-backups/{backup_id}	Deletes a specified volume backup.

#### Request Parameters

Parameter	Description
tenant_id	The unique identifier of the tenant or account.
backup_id	The unique identifier of the backup.

### Description

Deleting a previous volume backup from Swift. The operation does not require a request body and does not return a response body. This operation is asynchronous. You must list backups repeatedly to determine whether the backup was deleted.

### List backups

#### Operation

Verb	URI	Description
GET	v1.1/{tenant_id}/volume-backups	Lists backups.

#### Request Parameters

Parameter	Description
tenant_id	The unique identifier of the tenant or account.

### Response JSON

```
{
  "backups": [
    {
      "id": "1",
      "status": "available",
    },
    {
      "id": "2",
      "status": "creating",
    }
  ]
}
```

Name	Description
id	The unique identifier of the backup.
status	The status of the backup operation.

### Description

List backup id and status. The operation does not require a request body.

### List details for all backups

#### Operation

Verb	URI	Description
GET	v1.1/{tenant_id}/volume-backups/detail	Lists details for all backups.

### Request Parameters

Parameter	Description
tenant_id	The unique identifier of the tenant or account.

### Response JSON



```

{
  "backups": [
    {
      "id": "1",
      "display_name": "backup-001",
      "display_description": "Nightly Backup 03-Sep-2012",
      "volume_id": "9",
      "container": "nightlybackups",
      "status": "available",
      "fail_reason": null,
      "backup_start": "2012-09-03T22:00:00Z",
      "size": 10,
      "object_count": 2
    },
    {
      "id": "2",
      "display_name": "backup-002",
      "display_description": "Nightly Backup 04-Sep-2012",
      "volume_id": "9",
      "container": "nightlybackups",
      "status": "creating",
      "fail_reason": null,
      "backup_start": "2012-09-04T22:00:00Z",
      "size": 10,
      "object_count": 2
    }
  ]
}

```

Name	Description
id	The unique identifier of the backup.
display_name	User defined name for the backup.
display_description	User defined description for the backup.
volume_id	The identifier of the volume to be backed up.
container	The identifier of the container to which the volume is to be backed up.
status	The status of the backup operation.
fail_reason	If status above is set to failed, this contains more detail about the failure, otherwise it is null.
backup_start	The date and time at which the backup operation started.
size	The size of the backup in gigabytes.
object_count	The number of objects in the backup.

## Description

List details for all backups. The operation does not require a request body.

## Show backup details

### Operation

Verb	URI	Description
GET	v1.1/{tenant_id}/volume-backups/{backup_id}	Show information for specified backup.

### Request Parameters

Parameter	Description
-----------	-------------

tenant_id	The unique identifier of the tenant or account.
backup_id	The unique identifier of the backup.

## Response JSON

```
{
  "backup": {
    "id": "1",
    "display_name": "backup-001",
    "display_description": "Nightly Backup 03-Sep-2012",
    "volume_id": "9",
    "container": "nightlybackups",
    "status": "available",
    "fail_reason": null,
    "backup_start": "2012-09-03T22:00:00Z",
    "size": 10,
    "object_count": 2
  }
}
```

Name	Description
id	The unique identifier of the backup.
display_name	User defined name for the backup.
display_description	User defined description for the backup.
volume_id	The identifier of the volume to be backed up.
container	The identifier of the container to which the volume is to be backed up.
status	The status of the backup operation.
fail_reason	If status above is set to failed, this contains more detail about the failure, otherwise it is null.
backup_start	The date and time at which the backup operation started.
size	The size of the backup in gigabytes.
object_count	The number of objects in the backup.

## Description

Show details for a specified backup. The operation does not require a request body.

## Restore backup

### Operation

Verb	URI	Description
POST	v1.1/{tenant_id}/volume-backups/{backup_id}/restore	Restore specified backup to a volume.

### Request Parameters

Parameter	Description
tenant_id	The unique identifier of the tenant or account.
backup_id	The unique identifier of the backup.
restore	A partial representation of a restore that is used to create a restore.

## Request JSON

```
{
  "restore": {
    "volume_id": "9",
  }
}
```

Name	Description	Notes
volume_id	The identifier of the volume which the backup will be restored to.	Optional

## Response JSON

```
{
  "restore": {
    "backup_id": "1",
    "volume_id": "9",
  }
}
```

Name	Description
backup_id	The unique identifier of the backup being restored.
volume_id	The identifier of the volume which the backup will be restored to.

## Description

Restore an existing backup to a volume. If the volume is not specified, this operation will create a new volume for the restore. This operation is asynchronous. To check the status of the restore operation, the user should show volume for the volume\_id returned in the response (status will be set to **restoring** while the the restore operation runs, **error\_restoring** if the restore operation fails and **available** if the restore operation succeeds).

---

## Future Features

Areas of future development include the following

### Differential Backups

Support for differential backups of volume data. The objects list stored in the metadata file could be used to facilitate this. This feature will most likely require database updates as well. The specifics of how this feature may work in various scenarios are yet to be determined.

### Scheduled Backups

Allow the user to schedule volume backups for times when the volume is not in-use

### Configurable Compression

- Allow for the type of compression algorithm to be chosen
- This could be facilitated by either allowing the user to specify the compression algorithm via the API or by the server admin setting a flag in nova.conf

### Encrypted Backups

- Encryption of volumes for backup on user side
- Encryption of volumes for backup on server side