Contents

Introduction

The Essex Swift Administrator's Guide provides instructions for managing an OpenStack Swift environment. The guide is meant to compliment the existing OpenStack Swift Administrator?s Guide by including details of the Cisco OpenStack reference architecture.

Dependencies

The Admin Guide is based on the following software versions:

- **Ubuntu:** 12.04
- **Puppet:** 2.7.11
- **swift-proxy:** 1.4.8+stable+17-0~precise1
- **swift-account:** 1.4.8+stable+17-0~precise1
- **swift-container:** 1.4.8+stable+17-0~precise1
- **swift-object:** 1.4.8+stable+17-0~precise1

Remove a Failed Hard Drive

This process is used for removing a failed hard drive from a Swift Storage Node.

1. Comment-out or remove definitions for the bad hard drive from the container, object, account rings of the swift-nodes manifest. Here is an example of commenting-out device sdb from a Swift Storage Node named swift01:

```
#@@ring_object_device { "${swift01_local_net_ip}:6000/sdb":
#  zone        => $swift_zone,
#  weight      => 1,
#}

#@@ring_account_device { "${swift01_local_net_ip}:6002/sdb":
#  zone        => $swift_zone,
#  weight      => 1,
#}

#@@ring_container_device { "${swift01_local_net_ip}:6001/sdb":
#  zone        => $swift_zone,
#  weight      => 1,
#}
```

2. From the Proxy Node, remove the bad drive from the 3 swift rings:

```
swift-ring-builder /etc/swift/account.builder remove <ip_address>/<device_name>
```

```
swift-ring-builder /etc/swift/container.builder remove <ip_address>/<device_name>

swift-ring-builder /etc/swift/object.builder remove <ip_address>/<device_name>
```

3. From the Proxy Node, rebalance the Swift rings:

```
swift-ring-builder /etc/swift/account.builder rebalance

swift-ring-builder /etc/swift/container.builder rebalance

swift-ring-builder /etc/swift/object.builder rebalance
```

4. Verify the drive has been removed from the 3 rings:

```
swift-ring-builder /etc/swift/account.builder

swift-ring-builder /etc/swift/container.builder

swift-ring-builder /etc/swift/object.builder
```

5. Repeat steps 3-5 if you have additional proxy nodes.

Add a New Hard Drive

This process is used for adding a new hard drive to a Swift Storage Node.

1. Uncomment or add definitions for the new hard drive of the container, object, account rings of the swift-nodes manifest. Here is an example of adding device sdb to a Swift Storage Node named swift01:

```
@@ring_object_device { "${swift01_local_net_ip}:6000/sdb":
  zone        => $swift_zone,
  weight      => 1,
}

@@ring_account_device { "${swift01_local_net_ip}:6002/sdb":
  zone        => $swift_zone,
  weight      => 1,
}

@@ring_container_device { "${swift01_local_net_ip}:6001/sdb":
  zone        => $swift_zone,
  weight      => 1,
}
```

2. Run puppet on the Storage Node where the new drive has been added.

```
puppet agent -t -d
```

3. **Note:** Depending on how long it has been since the rings have been rebalanced, you may receive an error similar to the one below. You can disregard the error.

```
Scheduling refresh of Swift::Ringbuilder::Rebalance[account]info:
Swift::Ringbuilder::Rebalance[object]: Scheduling refresh ofExec[rebalance_object]
err: /Stage[main]/Swift::Ringbuilder/Swift::Ringbuilder::Rebalance[object]/Exec[rebalance_o
Failed to callrefresh: swift-ring-builder /etc/swift/object.builder rebalance returned 1
instead of one of [0] at /usr/share/puppet/modules/swift/manifest/ringbuilder/rebalance.pp:
Swift::Ringbuilder::Rebalance[account]: Scheduling refresh of Exec[rebalance_account]err: /
/Swift::Ringbuilder/Swift::Ringbuilder::Rebalance[account]/Exec[rebalance_account]:
Failed to call refresh: swift-ring-builder /etc/swift/account.builder rebalance
returned 1 instead of one of [0] at /usr/share/puppet/modules/swift/manifests/ringbuilder/r
info:Swift::Ringbuilder::Rebalance[container]: Scheduling refresh of Exec[rebalance_contain
/Stage[main]/Swift::Ringbuilder/Swift::Ringbuilder::Rebalance[container]/Exec[rebalance_con
Failed to call refresh: swift-ring-builder /etc/swift/container.builder
```

```
rebalance returned 1 instead of one of [0] at /usr/share/puppet/modules/swift/manifests/rin
notice: Finished catalog run in 9.10 seconds
```

4. From the Proxy Node, rebalance the Swift rings:

```
swift-ring-builder /etc/swift/account.builder rebalance

swift-ring-builder /etc/swift/container.builder rebalance

swift-ring-builder /etc/swift/object.builder rebalance
```

5. Verify the drive has been added to the 3 Swift rings:

```
swift-ring-builder /etc/swift/account.builder

swift-ring-builder /etc/swift/container.builder

swift-ring-builder /etc/swift/object.builder
```

6. Repeat steps 3-4 if you have additional Proxy Nodes.

Remove a Failed Storage Node

This process is used for removing a failed Storage Node from the Swift Ring.

1. Remove the node definition for the failed Storage Node from swift-node manifest. In this example I place three X's in the node name so that it does not match the Puppet Master:

```
node /XXXswift01/ inherits swift_base {
```

2. Remove storeconfigs entries for Storage Node in Puppet Master database. The example below removes a Storage Node with the IP address of 192.168.220.71:

```
mysql -u<admin_user> -p<admin_password> puppet

mysql> delete from resources where title like "192.168.220.71:%";
```

3. Remove the bad Storage Node from the Swift Proxy. The example below removes a storage node with an IP address of 192.168.220.71 and ring builder files located at /etc/swift/:

```
swift-ring-builder /etc/swift/account.builder remove 192.168.220.71

swift-ring-builder /etc/swift/container.builder remove 192.168.220.71

swift-ring-builder /etc/swift/object.builder remove 192.168.220.71
```

4. From the Proxy Node, rebalance the Swift rings:

```
swift-ring-builder /etc/swift/account.builder rebalance

swift-ring-builder /etc/swift/container.builder rebalance

swift-ring-builder /etc/swift/object.builder rebalance
```

5. Verify the Node has been removed from the 3 Swift rings:

```
swift-ring-builder /etc/swift/account.builder

swift-ring-builder /etc/swift/container.builder

swift-ring-builder /etc/swift/object.builder
```

Add a New Storage Node

This process is used for adding a new Storage Node to the Swift Ring.

1. Add a node definition to the swift-nodes manifest. The swift-nodes manifest provides an example of node definitions.
2. Remove the swift::ringsync storeconfig resource from the Puppet Master database:

   ```
   mysql -u<admin_user> -p<admin_password> puppet

   mysql> delete from resources where restype ='Swift::Ringsync';
   ```
3. Verify the Swift::Ringsync restype doe not appear the the database:

   ```
   mysql> select * from resources;
   ```
4. Run puppet if the Node is already managed by the Puppet Master (Build Node).

   ```
   puppet agent -t -d
   ```
5. If the Storage Node is new to the environment, add a node definition to the cobbler-node manifest.

   Apply the configuration to the Puppet Master:

   ```
   puppet apply -v /etc/puppet/manifests/cobbler-node.pp
   ```

   Verify the node has been added to Cobbler

   ```
   cobbler system list
   ```

   Start the provisioning process

   ```
   cobbler system poweron --name=<hostname or FQDN>
   ```
6. From the Proxy Node, rebalance the Swift rings:

   ```
   swift-ring-builder /etc/swift/account.builder rebalance

   swift-ring-builder /etc/swift/container.builder rebalance

   swift-ring-builder /etc/swift/object.builder rebalance
   ```
7. Verify the Node has been removed from the 3 Swift rings:

   ```
   swift-ring-builder /etc/swift/account.builder

   swift-ring-builder /etc/swift/container.builder

   swift-ring-builder /etc/swift/object.builder
   ```
8. Repeat steps 6-7 if you have additional Proxy Nodes.

Feedback

Provide documentation bugs or feedback to openstack-support@cisco.com