{...PRE-SETUP...}

Info: This is done before you run anything to setup the swift nodes (admin, proxy, storage … etc)

1. **Create the "swiftops" user on all systems:**
   - w3m -dump https://raw.github.com/btorch/swift-setup/master/contrib/setup_local_swiftops.sh | bash
   - wget https://raw.github.com/btorch/swift-setup/master/contrib/setup_remote_swiftops.exp && chmod 755 setup_remote_swiftops.exp
   - ./setup_remote_swiftops.exp IP_ADDR ROOT_PASSWORD [RSA .PUB FILE] ***(REPEAT AS MANY TIMES NEEDED)***
   - Now test access to the boxes from the **Admin box**

2. **Hosts file & DSH all group)**
   - Create a /etc/hosts file on the admin box for all swift nodes using the Management Net IPs
   - Install and create a **DSH** "all" group with every swift node (excluding admin box)

3. **Udev Rules Creation** *(SKIP THIS STEP IF DOING A VIRTUAL MACHINE ENVIRONMENT)*

   - The udev rule creation below are only needed to be performed on ONE storage node
   - The script will generate files for each run under /tmp/10_swift.rules.EPOCHTIME
   - Once you are done generating the proper files concatenate them to form a single .rules file
   - wget https://raw.github.com/btorch/swift-setuptools/master/contrib/udev_drive_rules.sh && chmod 755 udev_drive_rules.sh
   - sudo ./udev_drive_rules.sh -h ***(understand what you will be doing first)***
     - Example run: sudo ./udev_drive_rules.sh -c 0 -d sda -n 7
   - create a **10_swift.rules** from the */tmp/10_swift.rules.EPOCHTIME file(s) generated*
   - copy the file to /etc/udev/rules.d/ and run ***"udevadm trigger"*** to create symlinks *(verify them)*
   - If all good then copy the file to the admin box onto a safe location that you will remember

4. **Distribute the udev rules file**
   - Copy the 1***0_swift.rules*** *created on step 3 over all* ***storage nodes*** *and* ***reboot them***

5. **Drive setup** ***(On each storage node):***

   Suggestion: Use DSH and do things in parallel whenever possible
   - sudo apt-get install xfsprogs -Vy
   - wget https://raw.github.com/btorch/swift-setup/master/contrib/setup_drives.sh && chmod 755 setup_drives.sh
   - Seek Help -> sudo ./setup_drives.sh -h

6. **Hard drive burn-in** *(SKIP THIS STEP IF DOING A VIRTUAL MACHINE ENVIRONMENT)*
   - *Run bonnie++ on the drives to find out what breaks (may take a couple days)*
   - *Run stress tests as well*
   - *Once this is finished one should re-create the File System again (step 5)*

{...SWIFT SETUP...}

Info: One can deploy from any system where swift-setup is installed, as long as access to the swift Management Network is available and so is the swiftops private ssh key.

1. On any system

- git clone git://github.com/btorch/swift-setup.git
- READ : https://github.com/btorch/swift-setup/blob/master/README.md

2. Install the tools

- cd swift-setup && sudo python setup.py install

3. Create a swift-setup.conf

- cd /etc/swift-setup && sudo cp swift-setup.conf-sample swift-setup.conf
- Modify the values of the config files to suit your environment needs

4. Populate the /etc/swift-setup/hosts files

- Add hosts to the existing files or
- Copy over DSH groups you already have setup on the admin box **(pre-setup)**
- Make sure to have Hostname to IP mapping on /etc/hosts if using hostnames

5. Initialize the template

- Copy/Modify any files that you desire to the templates before initializing it
  - Example:
    - copy the udev rules create on pre-setup to storage/etc/udev/rules.d/
- sudo swift-setup init

6. Deploying the swift systems

- swift-setup deploy -H admin1.swift -t admin
- swift-setup deploy -g proxy -t proxy
- swift-setup deploy -g storage -t storage
- …..

7. Verify all systems are up

{...SWIFT RING DEPLOY...}

Info: There are instructions for folsom and grizzly+ releases
https://github.com/pandemicsyn/swift-ring-master
https://github.com/pandemicsyn/swiftscout

1. Start up all the swift services on all the storage nodes if not up yet

- swift-init all start (ignore the ring complaints)
- Make sure the services are up and listening on 600[0-2] ports

2. Create the rings *(For folsom, cd /srv/ring --- For grizzly, cd /etc/swift)*

- Choose the proper <part> <replicas> <min_hour> for the cluster in question
- **Finding a proper part_power:** python -c 'import math; print math.ceil(math.log(MAX_NUMBER_OF_DRIVES_EVER * 100 / 3, 2))'
- **Run:** for i in account container object ; do **sudo** swift-ring-builder $i.builder create <part> <reps> <mhour> ; done

FOLSOM: (*scripts located at the admin box under /srv/ring/scripts)*

3. Start adding nodes to the rings

- *Example*: /srv/ring/scripts/ring_add.sh -i 172.16.0.7 -r object -z 1 -w 100 -c 0 -s 1 -e 6 (-h for help/info)

4. Once all devices have been added to the all rings

- **sudo** /srv/ring/scripts/rebalance_ring.sh
- **sudo** /srv/ring/scripts/updatemd5.sh
- dsh -Mc -g GROUP "sudo sh -c '/usr/local/bin/retrievering.sh {RINGSERVER_IP}' "
- Restart all swift services everywhere

GRIZZLY: *(You will be using swiftscout and swift-ring-master utilities)*

4. Install the python utilities

- The swiftscout is installed only on the admin box
- The swift-ring-master is installed on the admin box and all nodes that require the swift rings

5. Start adding nodes to the rings using drivescout

- wget https://raw.github.com/btorch/swift-setup/master/contrib/drivescout_wrapper.sh && chmod 755 drivescout_wrapper.sh
- Read usage ./drivescout_wrapper.sh
- Verify the rings and then rebalance: sudo swift-ring-builder BUILDER_FILE rebalance

6. Start setting up *swift-ring-master* on the Admin box

- sudo cp /usr/share/swift-ring-master/ring-master.conf-sample /etc/swift/ring-master.conf
- Make any changes to the configs that you may see fit
- sudo cp /usr/share/swift-ring-master/swift-ring-master-init /etc/init.d/
- sudo cp /usr/share/swift-ring-master/swift-ring-master-wsgi-init /etc/init.d/
- sudo chown -R swift.swift /etc/swift
- sudo mkdir /var/log/ring-master && sudo chown swift.swift /var/log/ring-master
- sudo /etc/init.d/swift-ring-master-init start && sudo /etc/init.d/swift-ring-master-wsgi-init start
  - ***NOTE***: services are not started up at boot time unless you enable it with update-rc.d

7. Start setting up *swift-ring-master* on all systems that require the ring

- ***Assuming pkg is already installed on all systems***
- dsh -Mc -g GROUP 'sudo cp /usr/share/swift-ring-master/ring-minion.conf-sample /etc/swift/ring-minion.conf'
- dsh -Mc -g GROUP 'sudo sed -i "s;^#ring_master =.*;ring_master = http://RINGSERVER_IP:8090/;" /etc/swift/ring-minion.conf '
- dsh -Mc -g GROUP 'sudo mkdir /var/log/ring-master && sudo chown swift.swift /var/log/ring-master'
- dsh -Mc -g GROUP 'sudo swift-ring-minion-server -f -o ; sudo chown -R swift.swift /etc/swift'

8. Restart all swift services everywhere

- dsh -Mc -g GROUP 'sudo swift-init all restart'

{...POST-SETUP...}

1. **Test installation with swift or swiftly or curl**
2. **Enable the cron jobs on the git repo and push them out**
   - *proxy crons:*
     - memcache-restart
     - swift_ring_check
   - *storage crons:*
     - swift-device-audit
     - swift-recon-cron
     - swift_ring_check
     - xfs-corruption-check

{...TESTS...}

- **Authing:**
  curl -s -d '{"auth":{"passwordCredentials":{"username": "swiftops", "password": "swift"}, "tenantName": "swiftops"}}' -H 'Content-type: application/json' -XPOST http://127.0.0.1:5000/v2.0/tokens | python -mjson.tool |less

- **touch ~/.swiftrc and add below for "swift" cli tool**
  export OS_TENANT_NAME=swiftops
  export OS_USERNAME=swiftops
  export OS_PASSWORD=PASSSWORD_HERE
  export OS_REGION_NAME=RegionOne
  export OS_AUTH_URL=http://IP_ADDR:5000/v2.0
  export AUTH_VERSION=2.0

- **touch ~/.swiftlyrc and add below for "swiftly" cli tool (Not installed by default)**
  export SWIFTLY_AUTH_TENANT=swiftops
  export SWIFTLY_AUTH_USER=swiftops
  export SWIFTLY_AUTH_KEY=PASSSWORD_HERE
  export SWIFTLY_REGION=RegionOne
  export SWIFTLY_AUTH_URL=http://IP_ADDR:5000/v2.0
  export SWIFTLY_SNET=True
  export SWIFTLY_CACHE_AUTH=true