Cluster

Cluster configuration, theory, and troubleshooting.

Pacemaker

Pacemaker provides a framework to manage the availability of resources. It's essentially the core component on a cluster that allows you to manage resource locations etc.

Installing

apt-get install pacemaker pcs resource-agents

yum install pacemaker pcs resource-agents

Configuration

Add all hosts related to the cluster to /etc/hosts (including the host you're on, also remove any reference to localhost/ 127.0.0.1)
Set password for hacluster user:

```
passwd hacluster
```

authorise the nodes to be included in the cluster:

```
pcs host auth node01.srv.world node02.srv.world
```

start services for cluster:

```
pcs cluster start --all
```

enable services:

```
pcs cluster enable --all
```

Verify installation is working and nodes are connected:

```
pcs cluster status


pcs status corosync
```

Deleting a cluster installation

```
pcs cluster stop --all


pcs cluster destroy --all
```

Explanation

PCS (Pacemaker) Cluster

Services:

PCS (Pacemaker)

Pacemaker **provides a framework to manage the availability of resources**. Resources are services on a host that needs to be kept highly available.

ie pacemaker controls which services are running where.

Corosync

DRBD

DRBD is the service used for synchronisation of data (usually web and database files) on a cluster or HA solution. Not to be confused with Unision (which is typically used for synchronisation of config files due to not updating files as quickly as DRBD).

Installing

yum install drbd-utils* drbd-dkms

apt-get install drbd-utils* drbd-dkms

may need to add repos for this.

The above installs both the DRBD service files, as well as the required kernel module.

Check that the kernel module is loaded with:

```
modprobe drbd
```

Configuration of disks for DRBD

https://clusterlabs.org/pacemaker/doc/2.1/Clusters_from_Scratch/epub/shared-storage.html

DRBD will need its own block device on each node.

In this example, I've added a 10GB disk to each node,

```
root@b4sed-02:/etc# lsblk
NAME            MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda               8:0    0 19.1G  0 disk
├─sda1            8:1    0 18.8G  0 part /
├─sda14           8:14   0    1M  0 part
└─sda15           8:15   0  256M  0 part /boot/efi
sdb               8:16   0   10G  0 disk
└─VG_data-DRBD2 253:0    0   10G  0 lvm
sr0              11:0    1 1024M  0 rom


root@b4sed-02:/home# pvcreate /dev/sdb
  Physical volume "/dev/sdb" successfully created.
```

```
root@b4sed-02:/home# vgcreate VG_data /dev/sdb
  Volume group "VG_data" successfully created


root@b4sed-02:/home# vgdisplay | grep -e Name -e Free
  VG Name               VG_data
  Free  PE / Size       2559 / <10.00 GiB


root@b4sed-02:/home# lvcreate --name DRBD2 -l2559 VG_data
  Logical volume "DRBD2" created.
```

Configuration of DRBD

typically configured via /etc/drbd.conf

example configuration from my setup:

```
resource wwwdata {
 protocol C;
 meta-disk internal;
 device /dev/drbd1;
 syncer {
  verify-alg sha1;
 }
 net {
  allow-two-primaries;
 }
 on b4sed-01 {
  disk   /dev/VG_data/DRBD1;
  address  10.0.0.2:7789;
 }
 on b4sed-02 {
  disk   /dev/VG_data/DRBD2;
  address  10.0.0.3:7789;
 }
}
```

Once configuration file is in place, DRBD can be deployed via the below commands:

```
drbdadm create-md resourcename #wwwdata in my example


modprobe drbd


drbdadm up resourcename
```

as a side note, in my example I was getting errors relating to the kernel module, a kernel update and reboot resolved this.

Check status

```
drbdadm status
```

or

```
cat /proc/drbd
```

At this point you'll likely see data inconsistency:

```
root@b4sed-01:/var/log# drbdadm status
wwwdata role:Secondary
  disk:Inconsistent
  b4sed-02 connection:Connecting
```

This is because the data might differ on each node, to specific which node has the correct data we need to set the primary node, using the below command:

```
drbdadm primary --force wwwdata
```

Once done, run the deployment commands again but on the 2nd node:

```
drbdadm create-md resourcename #wwwdata in my example


modprobe drbd


drbdadm up resourcename
```

Once done, give some time for the connection to be made, check the status again:

```
root@b4sed-01:~# drbdadm status
wwwdata role:Primary
  disk:UpToDate
```

```
  b4sed-02 role:Secondary
    peer-disk:UpToDate
```

Configure the DRBD disk

mkfs.ext4 /dev/drbd1

nearly there now...

Add DRBD to the cluster

```
pcs cluster cib drbd_cfg #this queues up changed to be deployed to the cluster in one go
```

add the constraints and such:

```
pcs -f fs_cfg resource create WebFS Filesystem \
    device="/dev/drbd1" directory="/var/www/html" fstype="ext4"

pcs -f fs_cfg constraint colocation add \
    WebFS with WebData-clone INFINITY with-rsc-role=Master

pcs -f fs_cfg constraint order \
    promote WebData-clone then start WebFS

pcs -f fs_cfg constraint colocation add http_server with WebFS INFINITY

pcs -f fs_cfg constraint order WebFS then http_server
```

if all looks good, you can push these changes with the below command:

```
pcs cluster cib-push fs_cfg --config
```

time now for testing - place 1 node in standby and ensure the failover is sucesful.

Unison

Clusters use unison, usually for synchronisation of configuration files.