# EFM Demo

## Preparation

Log onto efm1 – We will call this efm1a

Log onto efm1 – We will call this efm 1b

Log onto efm2

Make sure that primary is 189!!

## Demo Script

**efm 1a** - watch sudo /usr/edb/efm-4.7/bin/efm cluster-status efmdemo

**efm 1a** - Point out primary is 189

**efm 1b** - hostname -I

**efm 1b** - Point out this is 189 the Primary

**efm 1b** - sudo -i -u enterprisedb psql postgresql://redhat-efm1,redhat-efm2/edb?target\_session\_attrs=read-write

**efm 1b** - SELECT inet\_server\_addr();

**efm 2** - sudo /usr/edb/efm-4.7/bin/efm promote efmdemo -switchover

Discuss what happens during switchover:

* First makes sure that the cluster is in a healthy state for promote – Show status
* Then ensures that the standby exists and is also healthy in an UP status.
* As part of the switch over EFM will fence off the primary node from any updates so that no new transactions are processed during the switchover to maintain data integrity.
* EFM ensures that the standby to be promoted is fully synchronized with the current primary, prior to promoting.
* Promotes standby to primary.
  + Reconfigures the standby node to be the primary.
  + Demotes the primary and reconfigures the primary node to be a standby node.
  + Starts up the replication from the new primary node.
* Updates the cluster configuration on who the new primary is.
* EFM performs checks to ensure that the new primary is operating correctly by running replication checks.
* Then releases the fencing and allows for new transactions
* EFM logs the switchover
* Notifications sent to DBAs if configured.
* **Other Considerations:**
  + When you switch over you must make sure you stop write traffic to the primary node being switched, or ensure that the client gracefully. There is an option to also run pre and post scripts that can run before and after the promote if needed.
  + Ensure you have enough disk-space for your WAL archive when performing the switch so all data is captured. 5000 TPS per section
  + When doing this for an upgrade, rolling may only be used with minor versions as all nodes need to be on the same major version.

**efm1b** - SELECT inet\_server\_addr(); - Twice

**efm1b** – State it is now pointing to new Primary.

**efm2** – Discuss now we are going to simulate an auto-failover scenario with the Postgres DB going down

**efm2** - sudo systemctl stop postgres.service

**Discussion Points:**

* EFM Agents are now within 15 sec sending a heartbeat message between the nodes using JGroups.
* Primary agent informs other EFM agents on the stand by and witness about failure.
* A **consensus** is reached between the stand by and witness node to verify failure.
* EFM waits for the configurable time out to ensure primary is down and it’s not a transient issue.
* EFM then facilitates the communication to notify the other nodes about this even so that the cluster can take action to failover.
* EFM determines which standby it will failover to based on priority, then other factors such as replication lag, readiness
* Then performs the actions previously discussed.
* Discuss how now there’s only one healthy node in this case. May want 2 stand-by nodes because now you’ll need to recover the standby which will require another shutdown period.
* EFM gives you the 4-9s – 52 minutes and 36 seconds a year

**efm1b** - SELECT inet\_server\_addr(); - Twice

**efm1b** – State is now pointing back to 189.

**efm2** - As ec2-user cd /home/ec2-user/scripts

**efm2** - ./efm\_rebuild\_node.sh

**Discussion Points:**

* Script then performs a restore of the data directory and WAL logs to the standby node and restarts cluster.