



PEM High Availability Setup Guide

Release 8.3.0

EnterpriseDB

Dec 25, 2021

CONTENTS

1	High Availability Using Failover Manager	3
2	Initial Product Installation and Configuration	5
3	Set Up the Primary Node for Streaming Replication.	7
4	Set Up the Standby Nodes for Streaming Replication	9
5	Set Up EFM to Manage Failover on All Hosts	13
6	Current Limitations	19

This document provides detailed information to set up a high availability (HA) environment for PEM using EDB Postgres Failover Manager (EFM).

See [Platform Compatibility](#) for information about the platforms and versions supported by PEM.

This document uses *Postgres* to mean either the PostgreSQL or EDB Postgres Advanced Server database.

HIGH AVAILABILITY USING FAILOVER MANAGER

Postgres Enterprise Manager (PEM) assists database administrators, system architects, and performance analysts in administering, monitoring, and tuning Postgres database servers.

Failover Manager (EFM) is a high availability tool from EDB that enables a Postgres primary node to automatically failover to a standby node in the event of a software or hardware failure on the primary.

The examples in the following sections use these IP addresses:

- 172.16.161.200 - PEM Primary
- 172.16.161.201 - PEM Standby 1
- 172.16.161.202 - PEM Standby 2
- 172.16.161.203 - EFM Witness Node
- 172.16.161.245 - PEM VIP (used by agents **and** users to connect)

The following needs to use the VIP address:

- The PEM Agent binding of the monitored database servers
- Accessing the PEM Web Client
- Accessing the Webserver services

INITIAL PRODUCT INSTALLATION AND CONFIGURATION

1. Install the following on the primary and one or more standbys:
 - – [EDB Postgres Advanced Server 13](#) (backend database for PEM Server) - [PEM Server](#) - [EDB Failover Manager 4.1](#)
 - Refer to the installation instructions in the product documentation using the links above or refer to the instructions given on the [EDB repos website](#). You need to replace the `USERNAME:PASSWORD` with your username and password in the instructions to access the EDB repositories.
 - Ensure that the database server is configured to use `scram-sha-256` authentication method as the PEM Server configuration script does not work with `trust` authentication.
 - You need to install the `java-1.8.0-openjdk` package to install EFM.
2. Configure the PEM Server on the primary server with initial configuration of type 1 (i.e. Web Services and Database):
3. Add the following ports in the firewall on the primary and all the standby servers to allow the access:
 - – 8443 - for PEM Server (https) - 5444 - for EPAS 13 - 7800 - for EFM - 7908 - for EFM Admin
 - For example:

```
* $ sudo firewall-cmd --zone=public --add-port=5444/tcp --permanent
* success
* $ sudo firewall-cmd --zone=public --add-port=8443/tcp --permanent
* success
* $ sudo firewall-cmd --zone=public --add-port=7800/tcp --permanent
* success
* $ sudo firewall-cmd --zone=public --add-port=7809/tcp --permanent
* success
* $ sudo firewall-cmd --reload
* success
```


SET UP THE PRIMARY NODE FOR STREAMING REPLICATION.

1. Create the replication role using the following command.

```
*      $ /usr/edb/as13/bin/psql -h 172.16.161.200 -p 5444 -U enterprisedb edb -c
↳ "CREATE ROLE repl REPLICATION LOGIN PASSWORD 'password';
*      CREATE ROLE
```

- Give the password of your choice.

2. Configure the following in the `postgresql.conf` file:

```
*      wal_level = replica
*      max_wal_senders = 10
*      wal_keep_size = 500
*      max_replication_slots = 10
```

- For more detailed information on configuring parameters for streaming replication refer to the [PostgreSQL documentation](#).
- !!! Note

- The configuration parameters may be different **for** different versions of database_↵
↵server. You can email support at [techsupport@enterprisedb.
↵com](mailto:techsupport@enterprisedb.com) **for any help in** setting up these_↵
↵parameters.

3. Add the following entry in the host-based authentication (`/var/lib/edb/as13/data/pg_hba.conf`) file to allow the replication user to connect from all the standbys:

```
*      hostssl replication repl 172.16.161.201/24 scram-sha-256
```

- !!! Note

- You can change the cidr **range** of the IP address, **if** needed.

4. Modify the host-based authentication (`/var/lib/edb/as13/data/pg_hba.conf`) file for the `pem_user` role to connect to all databases using the `scram-sha-256` authentication method as below:

```
*      # Allow local PEM agents and admins to connect to PEM server
*      hostssl all      +pem_user  172.16.161.201/24  scram-sha-256
*      hostssl pem      +pem_user  127.0.0.1/32      scram-sha-256
*      hostssl pem      +pem_agent 127.0.0.1/32      cert
*      # Allow remote PEM agents and users to connect to PEM server
```

(continues on next page)

(continued from previous page)

*	hostssl pem	+pem_user	0.0.0.0/0	scram-sha-256
*	hostssl pem	+pem_agent	0.0.0.0/0	cert

5. Restart the EPAS 13 Server.

*	systemctl restart edb-as-13.service
---	-------------------------------------

SET UP THE STANDBY NODES FOR STREAMING REPLICATION

1. Stop the service for EPAS 13 on all the standby nodes:

```
*      $ systemctl stop edb-as-13.service
```

- !!! Note

- This example uses the ``pg_basebackup`` utility to create the replicas of the PEM backend database server on the standby servers. When using ``pg_basebackup``, you need to stop the existing database server and remove the existing data directories.

2. Remove the data directory of the database server on all the standby nodes:

```
*      $ sudo su - enterprisedb
*
*      $ rm -rf /var/lib/edb/as13/data/*
```

3. Create `.pgpass` file in the home directory of the `enterprisedb` user on all the standby nodes:

```
*      $ sudo su - enterprisedb
*
*      $ cat > ~/.pgpass << _EOF_
*      172.16.161.200:5444:replication:repl:CHANGE_ME
*      172.16.161.201:5444:replication:repl:CHANGE_ME
*      172.16.161.202:5444:replication:repl:CHANGE_ME
*      _EOF_
*
*      $ chmod 600 ~/.pgpass
```

4. Take the backup of the primary node on each of the standby nodes using `pg_basebackup`:

```
*      $ sudo su - enterprisedb /usr/edb/as13/bin/pg_basebackup -h 172.16.161.200 \
*      -D /var/lib/edb/as13/data -U repl -v -P -Fp -R -p 5444
```

- The backup command creates the `postgresql.auto.conf` and the `standby.signal` file on the standby nodes. The `postgresql.auto.conf` file content is as below:

```
*      sudo su - enterprisedb cat /var/lib/edb/as13/data/postgresql.auto.conf
*      # Do not edit this file manually
*      # It will be overwritten by the ALTER SYSTEM command.
*      primary_conninfo = 'user=repl passfile="/var/lib/edb/.pgpass" channel_
↳binding=prefer host=172.16.161.200 port=5444 sslmode=prefer sslcompression=0 ssl_min_
↳protocol_version=TLsv1.2 gssencmode=prefer krbsvrname=postgres target_session_
↳attrs=any'
```

(continues on next page)

(continued from previous page)

5. Edit the following parameter in `postgresql.conf` file on each of the standby nodes:

```
*      hot_standby = on
```

6. Start the EPAS 13 database server on each of the standby nodes:

```
*      $ systemctl enable edb-as-13
*
*      $ systemctl start edb-as-13
```

7. Copy the following files from the primary node to the standby nodes at the same location, overwriting any existing files, and set the following permissions on the files:

- `/etc/httpd/conf.d/edb-pem.conf` - `/etc/httpd/conf.d/edb-ssl-pem.conf` - `/root/.pem/agent1.crt` - `/root/.pem/agent1.key` - `/usr/edb/pem/agent/etc/agent.cfg` - `/usr/edb/pem/share/.install-config` - `/usr/edb/pem/web/pem.wsgi` - `/usr/edb/pem/web/config_setup.py`

For example,

```
* $ mkdir -p /root/.pem
* $ chown root:root /root/.pem
* $ chmod 0755 /root/.pem
* $ mkdir -p /var/lib/pemhome/.pem
* $ chown pem:pem /var/lib/pemhome/.pem
* $ chmod 0700 /var/lib/pemhome/.pem
* $ mkdir -p /usr/edb/pem/logs
* $ chown root:root /usr/edb/pem/logs
* $ chmod 0755 /usr/edb/pem/logs
* $ for file in /etc/httpd/conf.d/edb-pem.conf \
*              /etc/httpd/conf.d/edb-ssl-pem.conf \
*              /root/.pem/agent1.crt \
*              /usr/edb/pem/agent/etc/agent.cfg \
*              /usr/edb/pem/share/.install-config \
*              /usr/edb/pem/web/pem.wsgi \
*              /usr/edb/pem/web/config_setup.py; do \
*     chown root:root ${file}; \
*     chmod 0644 ${file}; \
* done;
* $ chmod 0600 /root/.pem/agent1.key
* $ chown root:root /root/.pem/agent1.key
```

This ensures that webserver is configured on the standby and is disabled by default. It is enabled automatically at switchover by EFM.

!!! Note * You need to keep the certificates in sync on master and standbys manually whenever the certificates are updated.

8. Run the `configure-selinux.sh` script to configure the SELinux policy for PEM as following:

```
* $ /usr/edb/pem/bin/configure-selinux.sh
* getenforce found, now executing 'getenforce' command
* Configure the httpd to work with the SELinux
```

(continues on next page)

(continued from previous page)

```
* Allow the httpd to connect the database (httpd_can_network_connect_db = on)
* Allow the httpd to connect the network (httpd_can_network_connect = on)
* Allow the httpd to work with cgi (httpd_enable_cgi = on)
* Allow to read & write permission on the 'pem' user home directory
* SELinux policy is configured for PEM
* $ sudo chmod 640 /root/.pem/agent1.crt
```

!!! Note * At this point you should have a PEM Primary Server and two standbys that are ready to take over from the primary whenever needed.

SET UP EFM TO MANAGE FAILOVER ON ALL HOSTS

1. Prepare the Primary Node to support EFM:

- – Create a database user `efm` to connect to the database servers. - Grant the execute privileges on the functions related to WAL logs and the monitoring privileges to the user. - Add entries in `pg_hba.conf` to allow the `efm` database user to connect to the database server from all nodes on all the hosts. - Reload the configurations on all the database servers. For example,

```
*      $ cat > /tmp/efm-role.sql << _EOF_
*      -- Create a role for EFM
*      CREATE ROLE efm LOGIN PASSWORD 'password';
*      -- Give privilege to 'efm' user to connect to a database
*      GRANT CONNECT ON DATABASE edb TO efm;
*      -- Give privilege to 'efm' user to do backup operations
*      GRANT EXECUTE ON FUNCTION pg_current_wal_lsn() TO efm;
*      GRANT EXECUTE ON FUNCTION pg_last_wal_replay_lsn() TO efm;
*      GRANT EXECUTE ON FUNCTION pg_wal_replay_resume() TO efm;
*      GRANT EXECUTE ON FUNCTION pg_wal_replay_pause() TO efm;
*      GRANT EXECUTE ON FUNCTION pg_reload_conf() TO efm;
*      -- Grant monitoring privilege to the 'efm' user
*      GRANT pg_monitor TO efm;
*      _EOF_
*      $ /usr/edb/as13/bin/psql -h 172.16.161.200 -p 5444 -U enterprisedb edb -f /tmp/
↪efm-role.sql
*      CREATE ROLE
*      GRANT
*      GRANT
*      GRANT
*      GRANT
*      GRANT
*      GRANT
*      GRANT
*      GRANT ROLE
*
*      $ rm -f /tmp/efm-role.sql
*
*      $ cat > /var/lib/edb/as13/data/pg_hba.conf <<< _EOF_
*      hostssl      edb      efm      172.16.161.200/32      scram-sha-256
*      hostssl      edb      efm      172.16.161.201/32      scram-sha-256
*      hostssl      edb      efm      172.16.161.202/32      scram-sha-256
*      hostssl      edb      efm      172.16.161.203/32      scram-sha-256
*      _EOF_
*
```

(continues on next page)

(continued from previous page)

```
*      $ /usr/edb/as13/bin/psql -h 172.16.161.200 -p 5444 -U enterprisedb edb -c \
→ "SELECT pg_reload_conf();"

```

2. Create the scripts on each node to start/stop the PEM Agent:

```
*      $ sudo cat > /usr/local/bin/start-pemagent.sh << _EOF_
*      #!/bin/sh
*      /bin/sudo /bin/systemctl enable pemagent
*      /bin/sudo /bin/systemctl start pemagent
*      _EOF_
*      $ sudo cat > /usr/local/bin/stop-pemagent.sh << _EOF_
*      #!/bin/sh
*      /bin/sudo /bin/systemctl stop pemagent
*      /bin/sudo /bin/systemctl disable pemagent
*      _EOF_
*      $ sudo chmod 770 /usr/local/bin/start-pemagent.sh
*      $ sudo chmod 770 /usr/local/bin/stop-pemagent.sh

```

3. Create a sudoers file (/etc/sudoers.d/efm-pem) on each node to allow the efm user to start/stop the pemagent:

```
*      $ sudo cat > /etc/sudoers.d/efm-pem << _EOF_
*      efm    ALL=(ALL)          NOPASSWD:    /bin/systemctl enable pemagent
*      efm    ALL=(ALL)          NOPASSWD:    /bin/systemctl disable pemagent
*      efm    ALL=(ALL)          NOPASSWD:    /bin/systemctl stop pemagent
*      efm    ALL=(ALL)          NOPASSWD:    /bin/systemctl start pemagent
*      _EOF_

```

4. Create an efm.nodes file on all nodes using the sample file (/etc/edb/efm-4.1/efm.nodes.in), and give read-write access to the efm OS user:

```
*      $ sudo cp /etc/edb/efm-4.1/efm.nodes.in /etc/edb/efm-4.1/efm.nodes
*      $ sudo chown efm:efm /etc/edb/efm-4.1/efm.nodes
*      $ sudo chmod 600 /etc/edb/efm-4.1/efm.nodes

```

5. Add the ip address and EFM PORT of the primary node in the /etc/edb/efm-4.1/efm.nodes file on the standby nodes:

```
*      $ sudo cat > /etc/edb/efm-4.1/efm.nodes <<< _EOF_
*      172.16.161.200:7800
*      _EOF_

```

6. Create the efm.properties file on all the nodes using the sample file (/etc/edb/efm-4.1/efm.properties.in), and grant read access to all the users:

```
*      $ sudo cp /etc/edb/efm-4.1/efm.properties.in /etc/edb/efm-4.1/efm.properties
*      $ sudo chown efm:efm /etc/edb/efm-4.1/efm.properties
*      $ sudo chmod a+r /etc/edb/efm-4.1/efm.properties

```

7. Encrypt the efm user's password using efm utility:

```
*      $ export EFMPASS=password
*      $ /usr/edb/efm-4.1/bin/efm encrypt efm --from-env
*      096666746b05b081d1a98e43d94c9dad

```

8. Edit the following parameters in the properties file:

```
* db.user=efm
* db.password.encrypted=096666746b05b081d1a98e43d94c9dad
* db.port=5444
* db.database=edb
* db.service.owner=enterprisedb
* db.service.name=edb-as-13
* db.bin=/usr/edb/as13/bin
* db.data.dir=/var/lib/edb/as13/data
* jdbc.sslmode=require
* user.email=username@example.com
* from.email=node1@efm-pem
* notification.level=INFO
* notification.text.prefix=[PEM/EFM]
* bind.address=172.16.161.200:7800
* admin.port=7809
* is.witness=false
* local.period=10
* local.timeout=60
* local.timeout.final=10
* remote.timeout=10
* node.timeout=50
* encrypt.agent.messages=true
* stop.isolated.primary=true
* stop.failed.primary=true
* primary.shutdown.as.failure=false
* update.physical.slots.period=0
* ping.server.ip=8.8.8.8
* ping.server.command=/bin/ping -q -c3 -w5
* auto.allow.hosts=false
* stable.nodes.file=false
* db.reuse.connection.count=0
* auto.failover=true
* auto.reconfigure=true
* promotable=true
* use.replay.tiebreaker=true
* standby.restart.delay=0
* reconfigure.num.sync=false
* reconfigure.sync.primary=false
* minimum.standbys=0
* recovery.check.period=1
* restart.connection.timeout=60
* auto.resume.period=0
* virtual.ip=172.16.161.245
* virtual.ip.interface=ens33
* virtual.ip.prefix=24
* virtual.ip.single=true
* check.vip.before.promotion=true
* pgpool.enable=false
* sudo.command=sudo
* sudo.user.command=sudo -u %u
* syslog.host=localhost
```

(continues on next page)

(continued from previous page)

```
*      syslog.port=514
*      syslog.protocol=UDP
*      syslog.facility=LOCAL1
*      file.log.enabled=true
*      syslog.enabled=false
*      jgroups.loglevel=INFO
*      efm.loglevel=INFO
*      jvm.options=-Xmx128m
*      script.remote.post.promotion=/usr/local/bin/stop-pemagent.sh
*      script.post.promotion=/usr/local/bin/start-pemagent.sh
```

9. Set the value of the `is.witness` configuration parameter on the witness node to `true`:

```
*      is.witness=true
```

10. Enable and start the EFM service on the primary node:

```
* *      $ systemctl enable edb-efm-4.1
* *      $ systemctl start edb-efm-4.1
```

11. Allow the standbys to join the cluster started on the primary node:

```
*      /usr/edb/efm-4.1/bin/efm allow-node efm 172.16.161.201
*      /usr/edb/efm-4.1/bin/efm allow-node efm 172.16.161.202
*      /usr/edb/efm-4.1/bin/efm allow-node efm 172.16.161.203
```

12. Enable and start the EFM service on the standby nodes and the EFM witness node:

```
* *      $ systemctl enable edb-efm-4.1
* *      $ systemctl start edb-efm-4.1
```

13. Check the EFM cluster status from any node:

```
*      $ sudo /usr/edb/efm-4.1/bin/efm cluster-status efm
*      Cluster Status: efm
*      Agent Type  Address          DB      VIP
*      -----
*      Primary    172.16.161.200  UP      172.16.161.245*
*      Standby    172.16.161.201  UP      172.16.161.245
*      Standby    172.16.161.202  UP      172.16.161.245
*      Witness    172.16.161.203  N/A     172.16.161.245
*      Allowed node host list:
*      172.16.161.200 172.16.161.201 172.16.161.202 172.16.161.203
*      Membership coordinator: 172.16.161.200
*      Standby priority host list:
*      172.16.161.201 172.16.161.202
*      Promote Status:
*      DB Type      Address          WAL Received LSN  WAL Replayed LSN  Info
*      -----
*      Primary    172.16.161.200  0/F7A3808         0/F7A3808
*      Standby    172.16.161.201  0/F7A3808         0/F7A3808
*      Standby    172.16.161.202  0/F7A3808         0/F7A3808
*      Standby database(s) in sync with primary. It is safe to promote.
```

This status confirms that EFM is set up successfully and managing the failover for the PEM Server.

In case of failover, any of the standbys get promoted as the primary node. PEM Agents automatically connect to the new primary node. You can replace the failed primary node with a new standby using this procedure.

CURRENT LIMITATIONS

The current limitations include:- Web console sessions for the users are lost during the switch over.- Per user settings done from the **Preferences** dialog are lost as they're stored in local configuration files on the file system.- Background processes, started via **Backup**, **Restore**, and **Maintenance** dialogs, and their logs are not shared between the systems and are lost during switch over.