



PEM High Availability Setup Guide

リリース 8.3.0

EnterpriseDB

2021 年 12 月 25 日

目次

第 1 章 High Availability Using Failover Manager	3
--	---

このドキュメントでは、EDB Postgres Failover Manager (EFM) を使用して PEM の高可用性 (HA) 環境をセットアップするための詳細情報を提供します。

PEM がサポートするプラットフォームとバージョンについては、[Platform Compatibility](#) を参照してください。

このドキュメントでは、PostgreSQL または EDB Postgres Advanced Server データベースを意味する *Postgres* を使用しています。

第1章 High Availability Using Failover Manager

Postgres Enterprise Manager (PEM) は、データベース管理者、システム設計者、およびパフォーマンスアナリストが Postgres データベースサーバーを管理、モニタリング、およびチューニングするのを支援します。

Failover Manager (EFM) は、EDB の高可用性ツールであり、プライマリでソフトウェアまたはハードウェアに障害が発生したイベント、Postgres プライマリノードがスタンバイノードに自動的にフェイルオーバーできるようにします。

次のセクションの例では、これらの IP アドレスを使用しています。

- 172.16.161.200-プライマリ PEM
- 172.16.161.201-PEM スタンバイ 1
- 172.16.161.202-PEM スタンバイ 2
- 172.16.161.203-EFM 監視ノード
- 172.16.161.245-PEM VIP (エージェントとユーザーが接続に使用)

以下は VIP アドレスを使用するニーズがあります。

- 監視対象データベースサーバーの PEM エージェントバインディング
- PEM Web クライアントへのアクセス
- Web サーバーサービスへのアクセス

製品の初期インストールと構成

1. プライマリおよび 1 つ以上のスタンバイに次をインストールします。

- - (PEM サーバーのバックエンドデータベース) -
- 上記のリンクを使用して製品文書のインストール手順を参照するか、[EDB repos website](#) に記載されている手順を参照してください。EDB リポジトリにアクセスするための手順で、`USERNAME:PASSWORD` をユーザー名とパスワードに置き換える必要があります。
- PEM サーバー構成スクリプトは `trust` 認証では動作しないため、データベースサーバが `scram-sha-256` 認証メソッドを使用するように構成されていることを確認してください。
- EFM をインストールするには、`java-1.8.0-openjdk` パッケージをインストールする必要があります。

2. タイプ 1 の初期構成（つまり、Web サービスとデータベース）を使用して、プライマリーサーバで PEM サーバーを構成します。

```
* *      $ 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
```

3. プライマリサーバーとすべてのスタンバイサーバのファイアウォールに次のポートを追加して、アクセスを許可します。

- -8443-PEM サーバー (https) の場合-5444-EPAS 13 の場合-7800-EFM の場合-7908-EFM 管理者の場合例：

```
*      $ 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
```

ストリーミングレプリケーションのプライマリノードをセットアップします。

1. 次のコマンドを使用して、レプリケーションロールを作成します。

```
*      $ /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
```

- 任意のパスワードを入力してください。

2. postgresql.conf ファイルで次を構成します。

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

ストリーミングレプリケーションのパラメーターの構成の詳細については、[PostgreSQL documentation](#) を参照してください。

- !!! Note

- 構成パラメーターは、データベースサーバのバージョンによって異なる場合があります。これらのパラメーターの設定に関するヘルプについては、techsupport@enterprisedb.com でサポートにメールを送信できます。

3. ホストベース認証 (/var/lib/edb/as13/data/pg_hba.conf) ファイルに次のエントリーを追加して、レプリケーションユーザがすべてのスタンバイから接続できるようにします。

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

- !!! Note
- 必要に応じて、IP アドレスの cidr レンジを変更できます。

4. `pem_user` ロールのホストベース認証 (`/var/lib/edb/as13/data/pg_hba.conf`) ファイルを変更して、以下のように `scram-sha-256` 認証メソッドを使用してすべてのデータベースに接続します。

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

5. EPAS 13 サーバーを再起動します。

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

ストリーミングレプリケーション用のスタンバイノードのセットアップ

1. すべてのスタンバイノードで EPAS 13 のサービスを停止します。

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

- !!! Note
- この例では、`pg_basebackup` ユーティリティを使用して、スタンバイサーバに PEM バックエンドデータベースサーバのレプリカを作成します。`pg_basebackup` を使用する場合、既存のデータベースサーバを停止し、既存のデータディレクトリを削除する必要があります。

2. すべてのスタンバイノードでデータベースサーバのデータディレクトリを削除します。

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

3. すべてのスタンバイノードの `enterprisedb` ユーザのホームディレクトリに `.pgpass` ファイルを作成します。

```
*      $ 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_
```

(次のページに続く)

(前のページからの続き)

```
*
*      $ chmod 600 ~/.pgpass
```

4. pg_basebackup を使用して、各スタンバイノードでプライマリノードのバックアップを作成します。

```
* *      $ 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
```

バックアップコマンドは、スタンバイノードに postgresql.auto.conf および standby.signal ファイルを作成します。postgresql.auto.conf ファイルのコンテンツは次のとおりです。

```
*      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'
```

5. 各スタンバイノードで postgresql.conf ファイルの次のパラメータを編集します。

```
*      hot_standby = on
```

6. 各スタンバイノードで EPAS 13 データベースサーバを起動します。

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

7. 次のファイルをプライマリノードから同じ場所のスタンバイノードにコピーし、既存のファイルを上書きして、ファイルに次のアクセス許可を設定します。

- /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

例、

```
*      $ 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
```

(次のページに続く)

(前のページからの続き)

```
* $ 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
```

これにより、スタンバイで Web サーバーが構成され、デフォルトで無効になります。EFM によってスイッチオーバー時に自動的に有効になります。

!!! Note * 証明書が更新されるたびに、マスタとスタンバイで証明書の同期を手動で維持する必要があります。

8. `configure-selinux.sh` スクリプトを実行して、PEM の SELinux ポリシーを次のように構成します。

```
* $ /usr/edb/pem/bin/configure-selinux.sh
* getenforce found, now executing 'getenforce' command
* Configure the httpd to work with the SELinux
* 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 * この時点で、PEM プライマリサーバーと、必要に応じてプライマリから引き継ぐ準備ができている 2 つのスタンバイが必要です。

すべてのホストでフェイルオーバーを管理するための EFM のセットアップ

1. EFM をサポートするプライマリノードを準備します。

- データベースユーザに `efm` を作成して、データベースサーバーに接続します。-WAL ログに関連する機能の実行権限とモニタリング権限をユーザに付与します。`-pg_hba.conf` にエントリを追加して、`efm` データベースユーザがすべてのホスト上のすべてのノードからデータベースサーバーに接続できるようにします。-すべてのデータベースサーバーで構成を再読み込みします。例、

```
* $ 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
```

(次のページに続く)

(前のページからの続き)

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

2. 各ノードでスクリプトを作成して、PEM エージェントをスタート/停止します。

```
*      $ 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_
```

(次のページに続く)

(前のページからの続き)

```
* $ sudo chmod 770 /usr/local/bin/start-pemagent.sh
* $ sudo chmod 770 /usr/local/bin/stop-pemagent.sh
```

3. 各ノードで `sudoers` ファイル (`/etc/sudoers.d/efm-pem`) を作成して、`efm` ユーザが `pemagent` をスタート/停止できるようにします。

```
* $ 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. サンプルファイル (`/etc/edb/efm-4.1/efm.nodes.in`) を使用してすべてのノードで `efm.nodes` ファイルを作成し、`efm` OS ユーザに読み書きアクセスを許可します。

```
* $ 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. スタンバイノードの `/etc/edb/efm-4.1/efm.nodes` ファイルにプライマリノードの `ip address` と `EFM PORT` を追加します。

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

6. サンプルファイル (`/etc/edb/efm-4.1/efm.properties.in`) を使用してすべてのノードで `efm.properties` ファイルを作成し、すべてのユーザーに読み取りアクセスを許可します。

```
* $ 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. `efm` ユーティリティを使用して `efm` ユーザーのパスワードを暗号化します。

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

8. プロパティファイルで次のパラメータを編集します。

```
* db.user=efm
* db.password.encrypted=096666746b05b081d1a98e43d94c9dad
* db.port=5444
* db.database=edb
```

(次のページに続く)

(前のページからの続き)

```
*      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
```

(次のページに続く)

(前のページからの続き)

```
*      sudo.user.command=sudo -u %u
*      syslog.host=localhost
*      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. 監視ノードの `is.witness` 設定パラメータの値を `true` に設定します。

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

10. プライマリノードで EFM サービスを有効にしてスタートします。

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

11. スタンバイがプライマリノードで開始されたクラスターに結合できるようにします。

```
*      /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. スタンバイノードと EFM 監視ノードで EFM サービスを有効にしてスタートします。

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

13. 任意のノードから EFM クラスターステータスを確認します。

```
*      $ 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
```

(次のページに続く)

(前のページからの続き)

```
*      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
*      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.
```

このステータスは、EFM が正常にセットアップされ、PEM サーバーのフェイルオーバーを管理していることを確認します。

フェイルオーバーの場合、スタンバイのいずれかがプライマリノードとして昇格されます。PEM エージェントは、新しいプライマリノードに自動的に接続します。このプロシージャを使用して、障害が発生したプライマリノードを新しいスタンバイに交換できます。

#現在の制限

現在の制限には以下が含まれます：-ユーザーの Web コンソールセッションはスイッチ中に失われます-Preferences ダイアログから行われたユーザごとの設定は、ファイルシステムのローカル構成ファイルに保存されるため失われます-バックグラウンドプロセス、経由で開始 Backup、Restore、および Maintenance ダイアログ、およびそれらのログはシステム間で共有されず、スイッチオーバー中に失われます。