EDB Postgres
HA環境構築のベスト・プラクティス

EnterpriseDB
高鶴 勝治

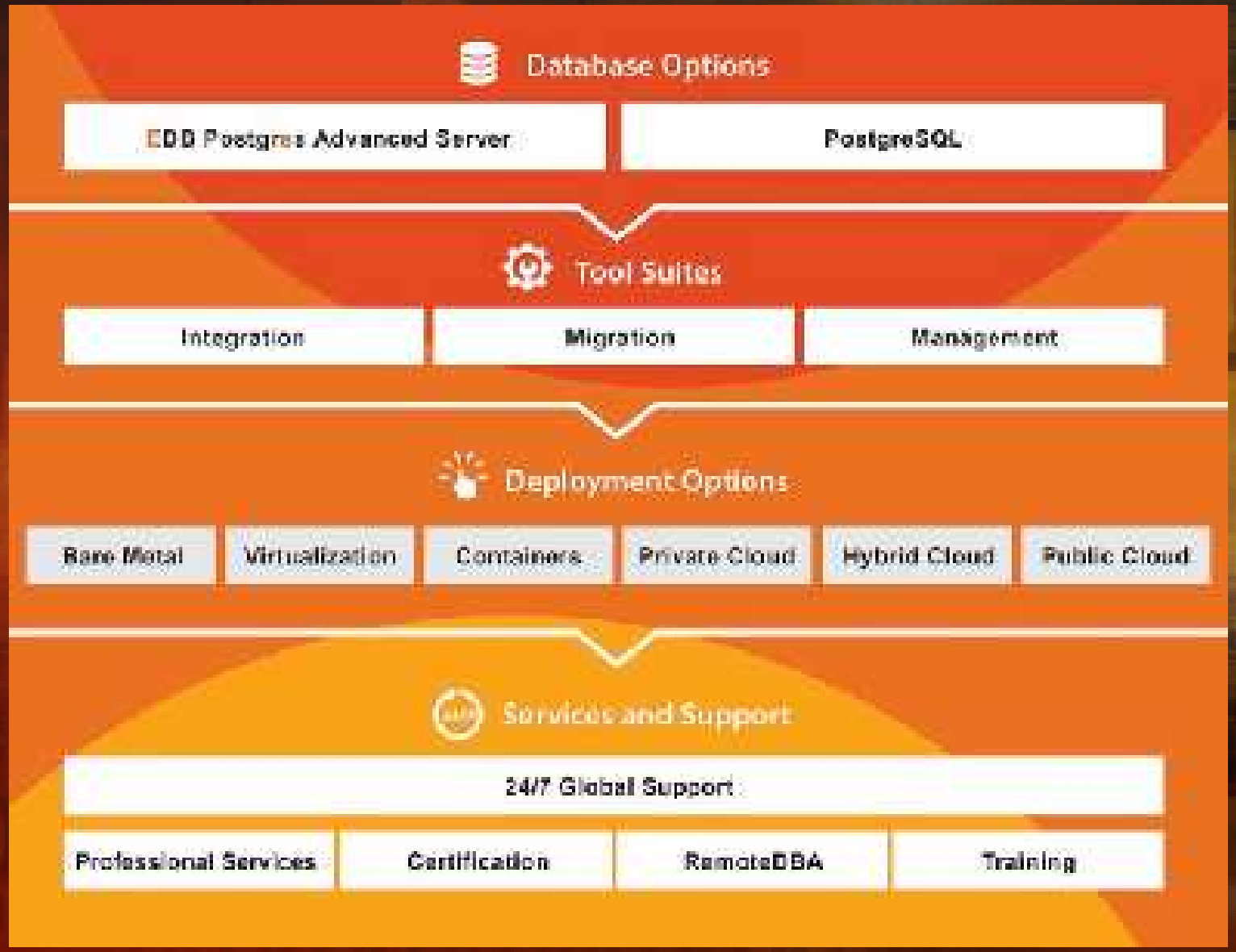
Agenda

1. EDB Postgres Platform
2. EDB Postgres におけるHA構成
3. Pgpool- II と EDB Failover Manager(EFM)
4. データベースサーバの構成
5. EDB Postgres Failover Manager (EFM) 設定

EDB Postgres Platform

データベースを核とした製品とサービス

EDB POSTGRES PLATFORM



EDB POSTGRES

EDB Postgres Platform： ツール・スイート

インテグレーション・スイート

ほぼリアルタイムで、さまざまなデータベース管理システムとのデータを交換

- EDB Postgres Data Adapters (FDW) - Hadoop,MySQL,Mongo,Spark との連携
- EDB Postgres Replication Server - 異種DBMS間レプリケーション(SMR), MMR
- EDB Postgres XA Connector - XA準拠、TM配下のRMとして稼働　(*) コントロール・リリース

マイグレーション・スイート

高価なレガシー・データベースからデータをシステマチックに移行

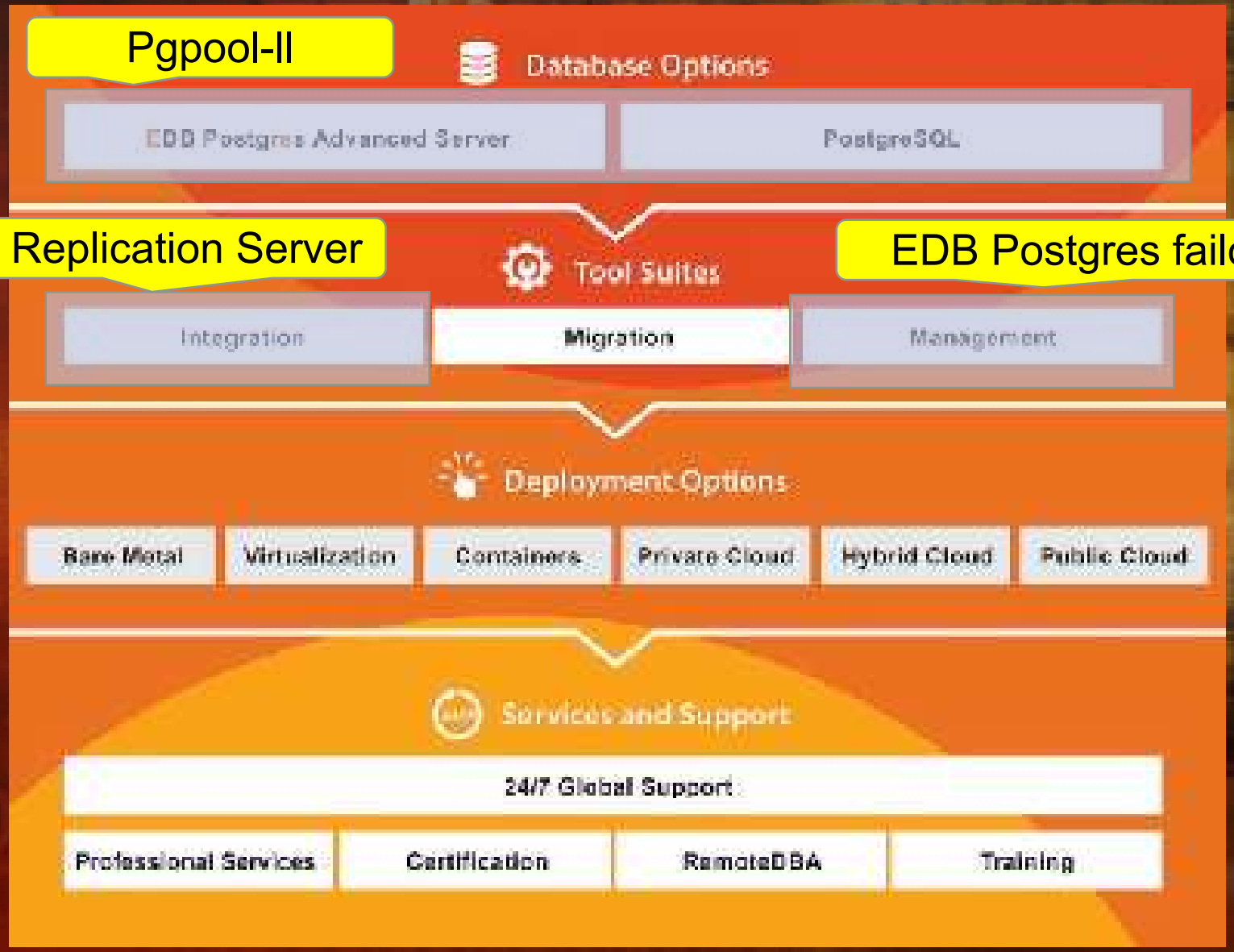
- EDB Postgres Migration Assessment　　*サービスとしてご提供
- EDB Postgres Migration Tool Kit - 異種DBMSからのシステマチックな移行

マネージメント・スイート

管理、監視、チューニング、高可用性やバックアップ、ディザスター・リカバリなど、ミッションクリティカルなシステムに必要な管理ツール

- EDB Postgres Enterprise Manager - 複数のEPAS/PostgreSQLの一元的な運用管理
- EDB Postgres Failover Manager - EPAS/PostgreSQLのHA
- EDB Postgres Backup and Recovery – EPAS/PostgreSQLのポリシー・ベースのバックアップ、ブロック・レベル・インクリメンタル・バックアップ

EDB POSTGRES PLATFORM



EDB POSTGRES

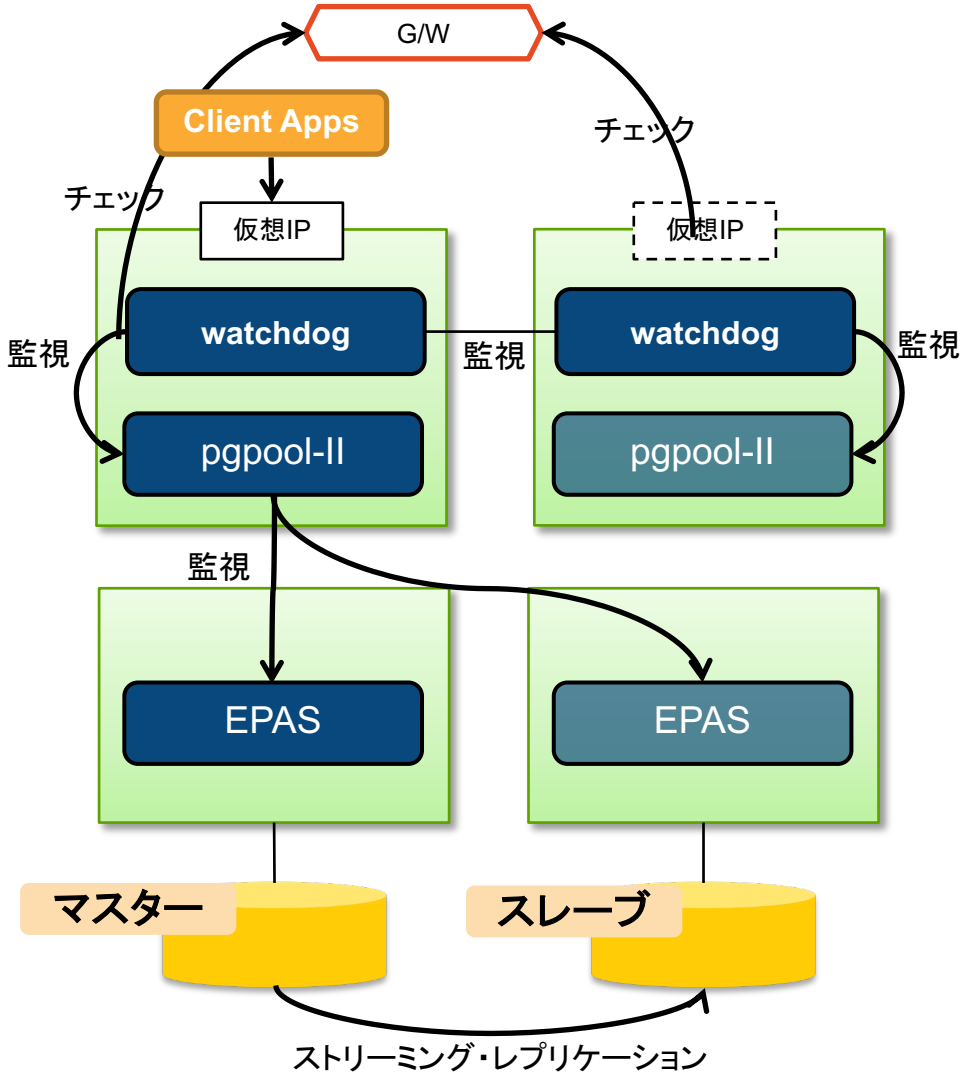
EDB Postgresで実現可能なHAクラスタ構成

EDB Postgres におけるHA構成パターン

- EPASでは、下記構成が可能
  - ① Pgpool-II による Master – Slave 構成
  - ② EFM による Master – Slave 構成
  - ③ Pgpool-II & EFM による Master – Slave 構成
  - ④ EDB Replication Server による Active –Active 構成
  - ⑤ クラスタ・ソフトによる Active – Passive 構成
- EDBは、pgpool-IIもサポート。但し、サポートされる機能は以下の通り。
  - ① ロード・バランシング
  - ② コネクション・プーリング
  - ③ Pgpool-II自体の可用性のための使用、WatchdogプロセスとMaster-Slaveモード

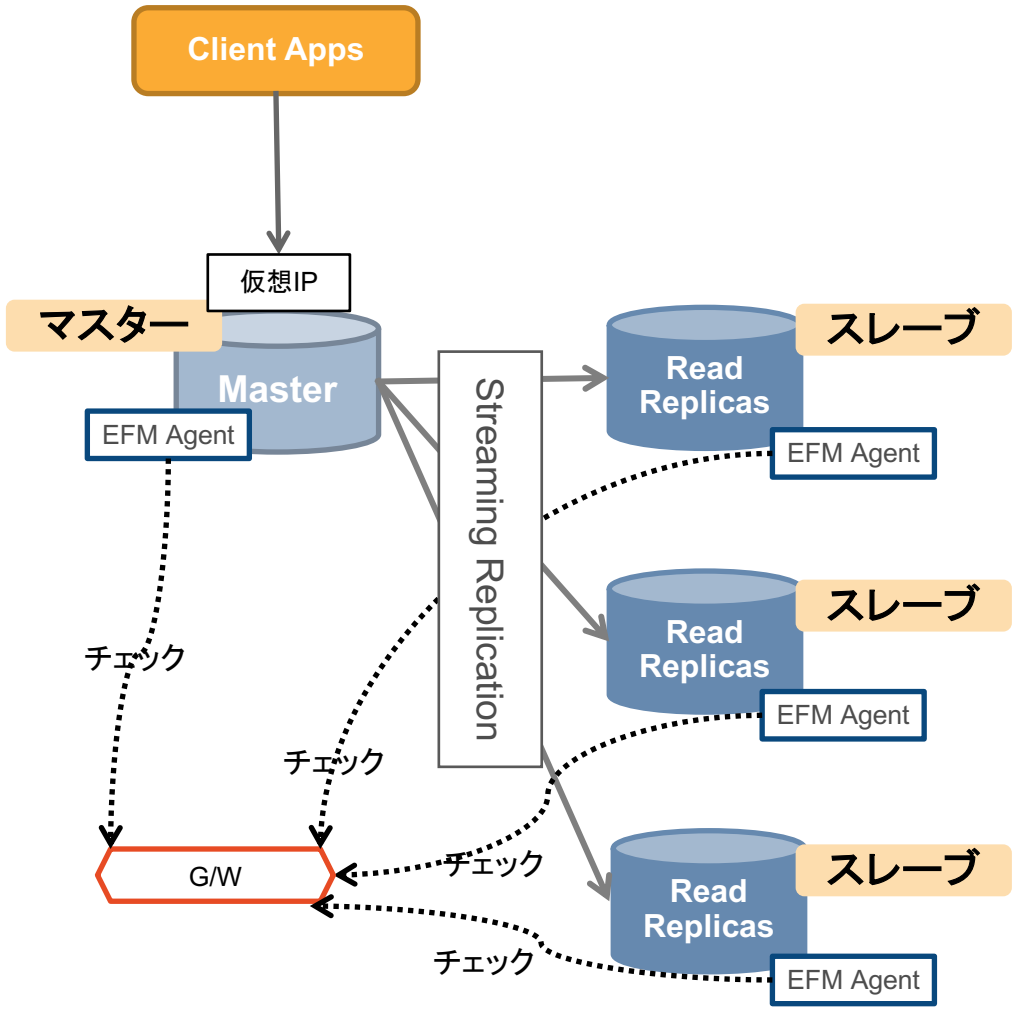
  ＊pgpool-IIの「レプリケーション機能」と「パラレル・クエリ機能」はサポート対象外

EPAS HA構成パターン①
/ Pgpool-II による Master – Slave 構成



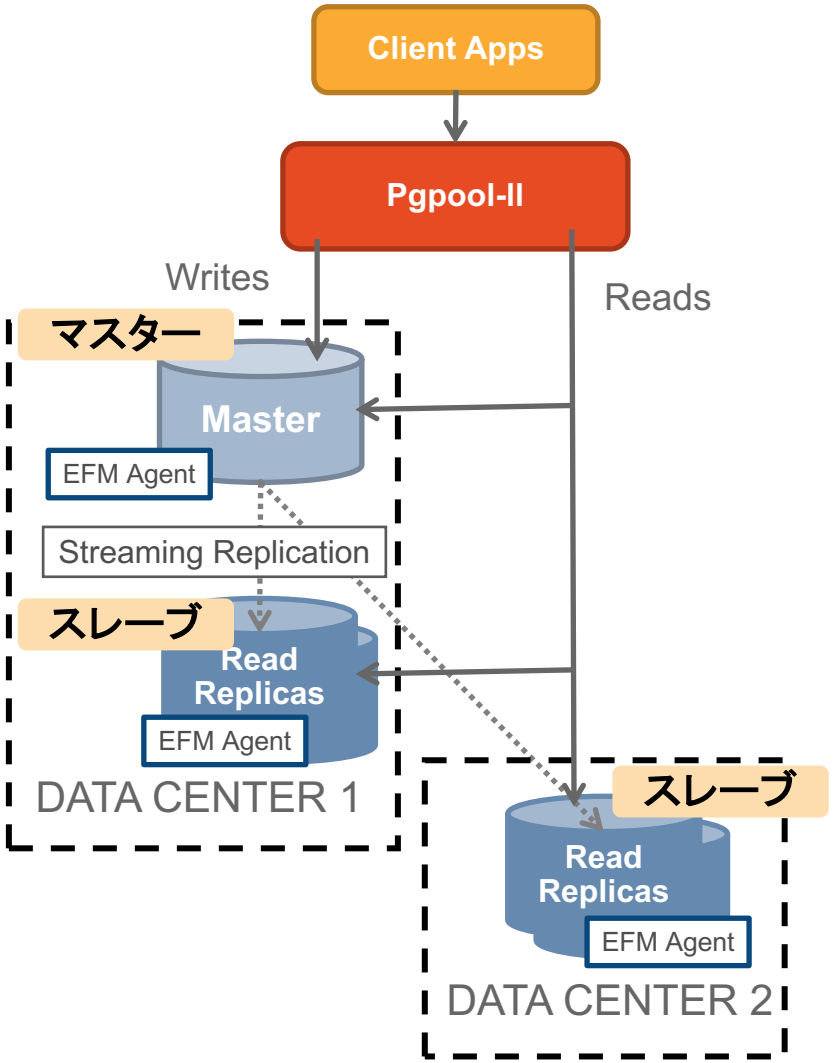
- pgppool－Ⅱは、Proxyサーバとして稼働。クライアント・アプリケーションは、pgppool－Ⅱがマスタ DBに付与する仮想IP(VIP)に対し接続。
- watchdogによる pgppool－Ⅱの死活監視
  - クラスタ・ソフトウェア(pgpoolHA等)が不要
  - プロセス存在のチェックではなく、サービス提供の可否をチェック（Default: “select 1”）
- watchdog 同士間の相互監視と情報共有
  - pgpool－Ⅱの稼働状況
  - EPAS/PostgreSQLの稼働状況
- マスタDB⇒スレーブDBは、ストリーミング・レプリケーションを使用。
- マスタDBの障害時は、スレーブDBへ自動切り替え(含VIPの切替え)
- コネクション・プーリングとロードバランシングが可能

EPAS HA構成パターン②
/ EFM による Master – Slave 構成



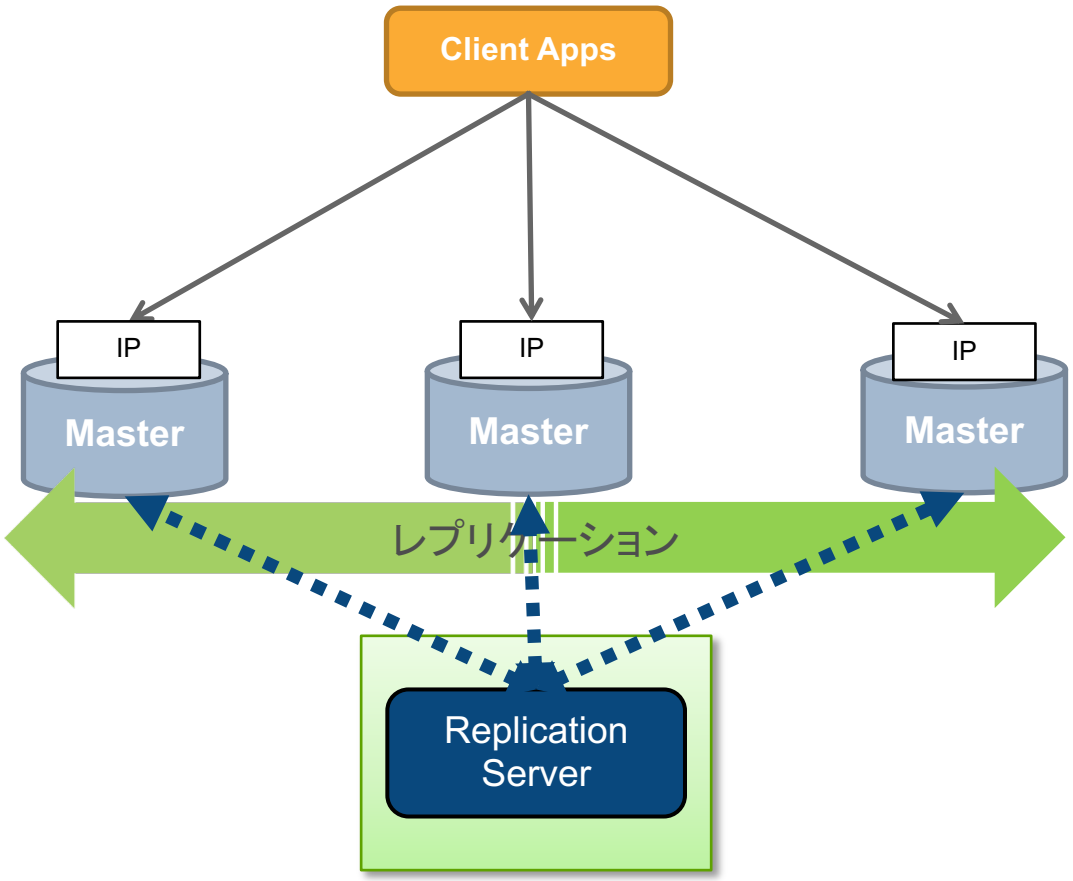
- エージェント型で、クライアント・アプリケーションは、EFMがマスタDBに付与する仮想IP(VIP)に対してアクセス(DBに対する直接アクセス)
- DBサーバ上でAgentが稼働。
  マスタ・サーバ: Master Agent
  スレーブ・サーバ: Slave Agent
- 2台のDB構成(Master 1台,Slave 1台)の場合は、Witness Agentにより、スプリット・ブレインを回避
- マスタDB⇒スレーブDBは、ストリーミング・レプリケーションを使用。
- マスタDBの障害時は、スレーブDBへ自動切り替え(含VIPの切替え)

EPAS HA構成パターン③
/ Pgpool-II & EFM による Master – Slave 構成



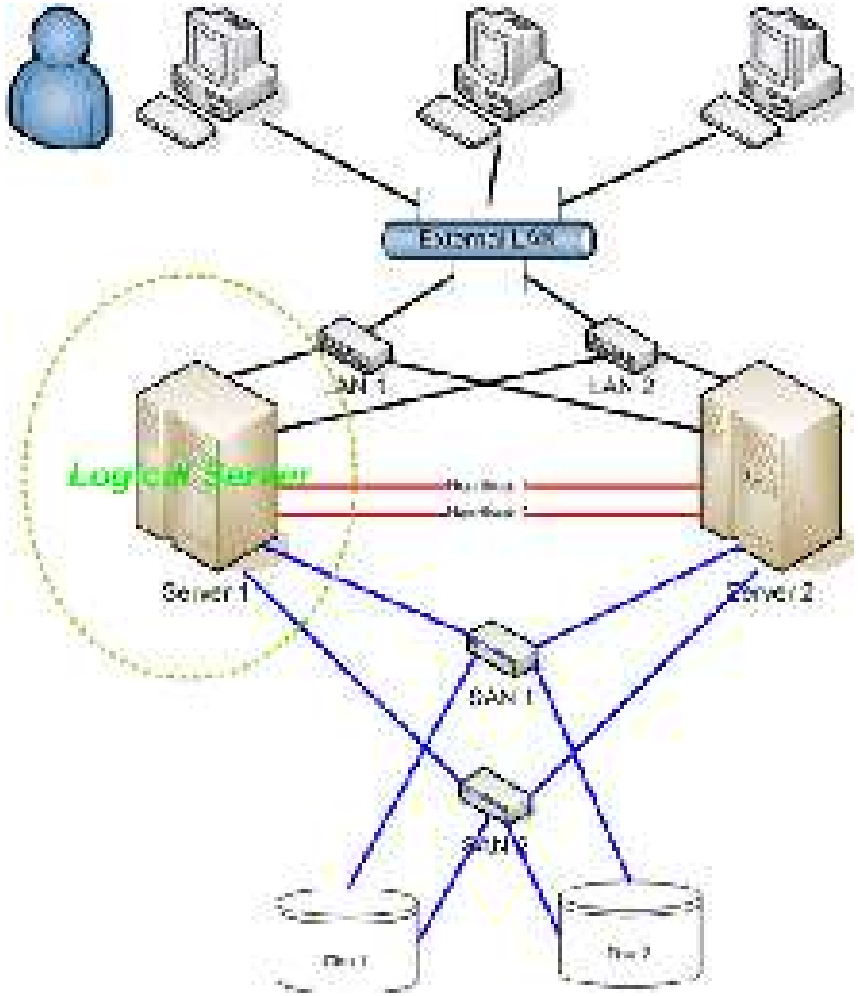
- Pgpool– II と EFM を組み合わせた構成
- Pgpool– II は、コネクション・プーリングとロード・バランサー機能を提供
- EFMは、DBサーバのHA機能を提供
- スケール・アウトと大量センションを処理する必要がある時に有効

EPAS HA構成パターン④
/ Replication Server による Active –Active 構成



- EDB Postgres Replication サーバによる、テーブル・レベルのマルチ・マスター・レプリケーション。
- クライアント・アプリケーションは、各MasterサーバのIPアドレスにアクセス。

EPAS HA構成パターン⑤
/ クラスタ・ソフトによる Active – Passive 構成



- データを共有ディスクで一元管理
- 実績あるクラスタリングソフトにより、障害時に確実なフェイル・オーバー
  - Service Guard
  - Red Hat Cluster
  - Veritas Cluster
  - LinuxHA
  - Cluster Pro
  - LifeKeeper

HAクラスタ構成の比較

- サービス継続能力は、**Master-Slave型の方が高い**

パターン	形態	使用製品	マスターノード障害時		マスタDBのディスク障害	コネクション・プーリング	スケール・アウト
			切替時間	データ・ロス可能性			
①	Master-Slave	• Pgpool-II	検知時間 +Promote時間	非同期レプリケーション あり 同期レプリケーション なし	DB障害検知後、Slaveに切替	○	○ (*3)
②		• EDB Postgres Failover Manager					X (*4)
③		• Pgpool-II (*1) • EDB Postgres Failover Manager				○	○ (*3)
④	Active-Active	• EDB Postgres Replication Server	検知時間	あり	DB障害検知後、他サーバで継続	○	○ (*5)
⑤	Active-Passive	• クラスター・ソフトウェア (*2)	検知時間+FSマウント時間+EDB Postgres 起動時間	なし	メディア・リカバリ +EDB Postgres 起動+フォワード・リカバリ	○	X

(*1) pgpool-II は、コネクション・プーリング機能とロードバランシング機能を提供。

(*2) RHCSや、Cluster Pro Life Keeper 等を使用。

(*3) Slave利用による参照系のスケールアウトを実現。

(*4) EFMを単体で使用する場合、複数スレーブDB構成をサポートしているので、CPUオフロード的な使用は可能。

(*5) アプリケーション・パーティショニング可能な場合は、更新系のスケール・アウトが可能。但し、Conflictの対応が必要。

Pgpool-II と EDB Postgre Failover (EFM)

EDB POSTGRES

Pgpool-II

- Proxyサーバであり、VIPを提供。
- クライアント・アプリケーションは、pgpool-II経由でEPAS/PostgreSQLに接続
- データベースの外からデータベースのヘルス・チェック/障害検知と自動フェイル・オーバ
- フェイル・オーバ後のオンライン・リカバリ機能を提供

⇒ **データベースのスプリット・ブレイン回避の対応は別途必要**

EDB Postgres Failover Manager

- エージェント型 + Witnessにより、データベースのSplit-Brainの防止
- VIPを提供
- データベースのヘルス・チェック &カスタム・モニタリング機能
- マスタ障害時自動フェイル・オーバとクラスタ再構成
- ストリーミング・レプリケーション同期状態のチェック
- 複数Slaveサーバの優先度を動的に指定可能
- DBサーバ間ネットワーク(ストリーミング・レプリケーション用)のヘルスチェック
- スイッチ・オーバ/スイッチ・バック機能
- Fencingスクリプトによる外部リソースの制御
- クラスタ構成変更時のメールユーザ処理実行
- クラスタ構成変更時のメール送信機能
- マスターDB障害後のサーバ再起動による、Split-Brainの防止

製品のカバー範囲

	Pgpool-II	EFM	備　　考
VIP機能提供	○	○	
DBヘルスチェック&自動フェイル・オーバ	○	○	EFM: recovery.conf 自動生成によるSplit-Brainの防止
スイッチ・オーバー/スイッチ・バック	○	○	EFM: 同期状態のチェックとrecovery.conf 自動生成による容易で確実なスイッチ・オーバ
複数スレーブDB対応	○	○	
ロード・バランシング機能	○	X	Pgpool-II の強み
コネクション・プーリング機能	○	X	
データベースのSplit-Brainの防止	X	○	EFM: Agent型+Witness
カスタム・モニタリング機能	X	○	EFM: 追加のヘルスチェックを実装可能
複数Slaveサーバの優先度の動的な指定	X	○	
DBサーバ間ネットワークのヘルチェック	X	○	
Fencingスクリプトによる外部リソース制御	X	○	EFMの強み
クラスタ構成変化時のユーザ処理実行	X	○	
クラスタ構成変化時のメール送信	X	○	
オンライン・リカバリ機能	○	X	EFM: BARTによるオンライン・リカバリを推奨

パターン選択の考え方

パターン②／パターン③で考える

ダイナミックなスケール・アウト(Read)が必要

アプリケーション外での、コネクション・プーリングが必要
(大量なセッションを処理する必要がある)

→ pgpool-II は“**必須**”
⇒ パターン① or パターン③

データベースのフェイル・オーバーを、pgpool-II or EFMどちらで行うか

pgpool-II を用いたHA構築のノウハウがある → pgpool-II を使用[パターン①]

それ以外
- DB直接接続 → EFMを 使用[パターン②]
- スケール・アウト → Pgpool-II + EFM 組み合わせ [パターン③]
- コネクション・プーリング → Pgpool-II + EFM 組み合わせ [パターン③]

3rd party クラスタ・ソフトの必要性

- Pgpool-IIおよびEDB Failover Manager は、select文の実行成否で、データベースの健全性をチェック。
  (*) EFMでは、カスタム・モニタリングで、健全性を別途定義可能
- NIC障害やDisk障害の検知までは行わないため、データベースの健全性チェックがエラーとなるまでに、タイムラグが発生するケースがある。
- EFM agent の稼働チェックを行うことで、可用性を更に高めることができる。
  (*) EFM機能によるDBA通知により対応は可能。

3rd Party クラスタ・ソフトを併用することで、障害検知の精度を上げ、サービス停止時間を縮小させるが可能。

データベース・サーバの構成

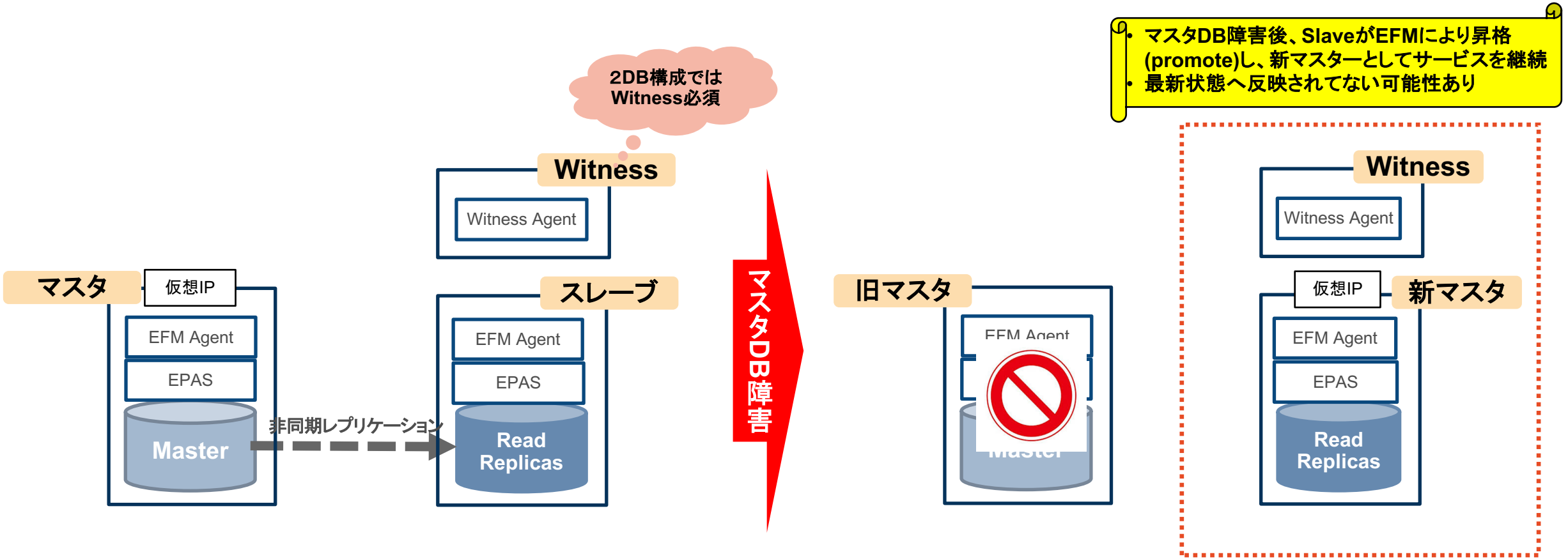
EDB POSTGRES

ストリーミング・レプリケーション構成の選択

- 1ノードの障害のみ想定する場合
  - データ・ロスの可能性を許容可能： **非同期レプリケーション**
    - 2DB(マスタ+非同期スレーブ1台)+Witness [パターン①]
    - Witness Agentは、アプリケーション・サーバ、もしくは、バックアップ・サーバ上で稼働
  - データ・ロス・ゼロの実現： **同期レプリケーション**
    - 3DB(マスタ+同期スレーブ1台+非同期スレーブ1台) [パターン②]
    - Witness Agentは不要
- 2ノードの障害まで想定する場合 [パターン③ / パターン④]
  - DBサーバ数を1台に増やす
  - 非同期の場合、2台目障害時のスプリット・ブレインを防ぐため、Witness Agent を組み込む

ストリーミング・レプリケーション構成パターン①

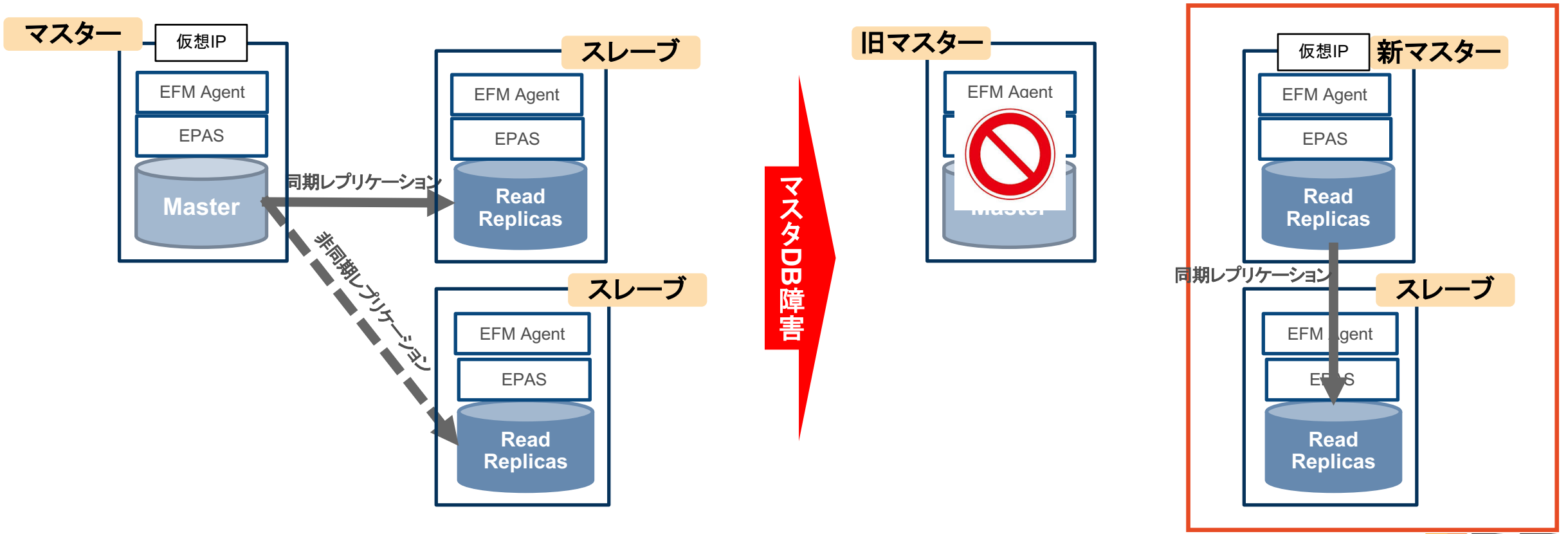
- 2DB(マスタ+非同期スレーブ1台)+Witness



ストリーミング・レプリケーション構成パターン②

- 3DB(マスタ+同期スレーブ1台+非同期スレーブ1台)

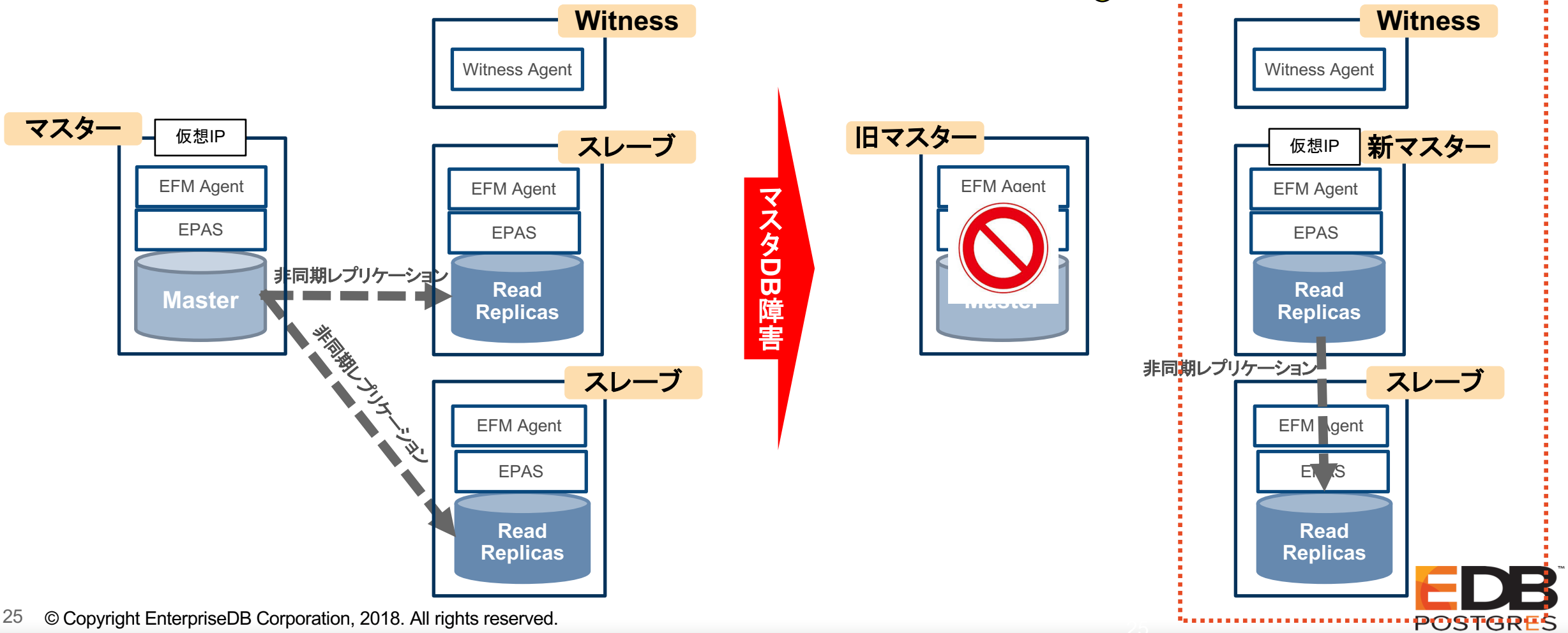
マスタDB障害後、SlaveがEFMにより昇格(promote)し、データ・ロスなしで、新マスターとしてサービスを継続。



ストリーミング・レプリケーション構成パターン③

- 3DB(マスタ+非同期スレーブ2台)+Witness(オプション)

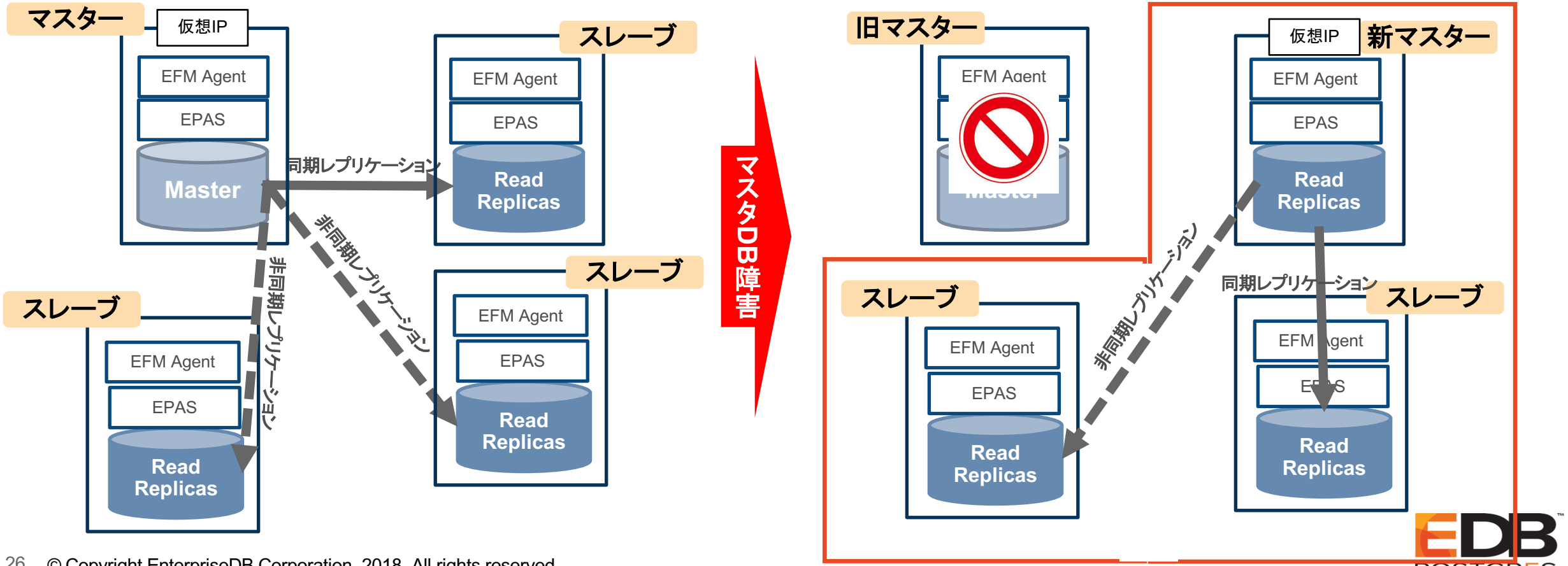
- マスタDB障害後、SlaveがEFMにより昇格(promote)し、新マスターとしてサービスを継続。但し、最新状態へ反映されてない可能性あり
- Witnessがあれば、新マスターの障害時もスプリット・ブレインなしで、フェイル・オーバ可能



ストリーミング・レプリケーション構成パターン④

- 4DB(マスタ+同期スレーブ1台+非同期スレーブ2台)

- マスタDB障害後、SlaveがEFMにより昇格(promote)し、データ・ロスなしで、新マスターとしてサービスを継続。
- 2台目の非同期スレーブを用意することで、新マスターの障害時もスプリット・ブレインなしで、フェイル・オーバ可能



EDB Postgres Failover Manager (EFM) 設定

EFM環境の構築手順

#	実施内容	efm01	efm02	efm03	備　　考
1	OSインストール & NIC 設定	✔	✔	✔	• Java と　SMTP 含む • NIC Bonding
2	EDB Postgres Advance Server インストール	✔	✔	✔	
3	ストーミング・レプリケーションの構成	✔ (Master)	✔ (Slave)	✔ (Slave)	• レプリケーション・ユーザの作成 • postgresql.conf / pg_hba.conf 設定 • pg_basebackup で複製 • recover.conf 作成
4	EDB Failover Managerインストール	✔	✔	✔	• yumコマンドによるインストール
5	EFM 設定　& 起動	✔	✔	✔	• efm.properties 設定 • efm.nodes 設定 • プロセスの起動/停止は、systemctl or service

efm.properties@efm01

```
db.user=enterprisedb
db.password.encrypted=*****************************************
db.port=5444
db.database=edb
db.service.owner=enterprisedb
db.service.name=edb-as-10
db.bin=/opt/edb/as10/bin
db.recovery.conf.dir=/opt/edb/as10/data
jdbc.sslmode=disable
user.email=katsuji.takatsuru@enterprisedb.com
virtualIp=192.168.0.160
virtualIp.interface=ens32
virtualIp.prefix=24
admin.port=7802
pingServerIp=192.168.0.1
pingServerCommand=/bin/ping -q -c3 -w5
is.witness=false
script.notification=
local.period=10
local.timeout=60
local.timeout.final=10
remote.timeout=10
node.timeout=50
bind.address=192.168.0.161:7801
stop.isolated.master=true
auto.allow.hosts=true
db.reuse.connection.count=0
auto.failover=true
auto.reconfigure=true
promotable=true
minimum.standbys=0
recovery.check.period=2
auto.resume.period=10
script.fence=
script.post.promotion=
script.resumed=
script.db.failure=
script.master.isolated=
script.remote.pre.promotion=
script.remote.post.promotion=
script.custom.monitor=
custom.monitor.interval=
custom.monitor.timeout=
custom.monitor.safe.mode=
sudo.command=sudo
sudo.user.command=sudo -u %u
log.dir=/var/log/efm-3.0
jgroups.loglevel=FINER
efm.loglevel=FINER
jvm.options=-Xmx32m
```

① ② ③

efm.nodes

efm.nodes@edbefm01

```
# List of node address:port combinations separated by whitespace.
192.168.0.162:7801 192.168.0.163:7801
```

efm.nodes@edbefm02

```
# List of node address:port combinations separated by whitespace.
192.168.0.161:7801 192.168.0.163:7801
```

efm.nodes@edbefm03

```
# List of node address:port combinations separated by whitespace.
192.168.0.161:7801 192.168.0.162:7801
```

efmコマンドによるクラスタ環境確認

[実行コマンド]

[root@edbefm01 ~]# /usr/edb/efm-3.0/bin/efm cluster-status efm

[実行結果]

```
Cluster Status: efm
VIP: 192.168.100.160

      Agent Type  Address              Agent  DB       Info
      --------------------------------------------------------------
      Master      192.168.0.161        UP     UP
      Standby     192.168.0.162        UP     UP
      Standby     192.168.0.163        UP     UP

Allowed node host list:
      192.168.0.161 192.168.0.162 192.168.0.163

Membership coordinator: 192.168.0.161

Standby priority host list:
      192.168.0.162 192.168.0.163

Promote Status:

      DB Type     Address              XLog Loc         Info
      --------------------------------------------------------------
      Master      192.168.0.161        0/B0000D0
      Standby     192.168.0.162        0/B0000D0
      Standby     192.168.0.163        0/B0000D0

      Standby database(s) in sync with master. It is safe to promote.
```

まとめ

- EDB Postgres Failover Manager を用いた、Master-Slave 構成

- 非同期レプリケーションであれば2台以上、同期レプリケーションであれば3台以上