FUJITSU

FUJITSU Software
Enterprise Postgres 14 SP1

クラスタ運用ガイド
(データベース多重化編)

Linux

J2UL-2750-02ZJZ0(01)
2024年4月

まえがき

本書の目的

本書は、FUJITSU Enterprise Postgresのデータベース多重化機能を利用するために必要な事項を説明しています。

本書の読者

本書は、データベース多重化機能をセットアップおよび利用される方を対象としています。

なお、本書は、以下についての一般的な知識があることを前提に書かれています。

- PostgreSQL
- SQL
- Linux

本書の構成

本書は以下の構成になっています。

第1章 データベース多重化運用の概要

データベース多重化運用の概要について説明しています。

第2章 データベース多重化運用のセットアップ

データベース多重化運用のセットアップ方法について説明しています。

第3章 データベース多重化運用の操作

データベース多重化運用の操作について説明しています。

第4章 データベース多重化運用の異常時の対処

データベース多重化運用時に異常が発生した場合の対処について説明しています。

第5章 WebAdminを使用したMirroring Controllerの管理

WebAdminを使って、ストリーミングレプリケーションクラスタでMirroring Controllerをセットアップ、および管理する方法を説明しています。

付録A パラメータ

データベース多重化運用で必要となる各設定ファイルおよびパラメータについて説明しています。

付録B プライマリサーバとスタンバイサーバを同一サーバ内に構築する場合の補足事項

プライマリサーバとスタンバイサーバを同一サーバ内に構築する場合の補足事項について説明しています。

付録C ユーザーコマンド

ユーザーコマンドについて説明しています。

付録D ハートビート異常で即時に自動縮退を行うを選択した場合の注意事項

OSやサーバの生死監視において、ハートビート異常で無条件に自動縮退を行うを選択した場合の注意事項について説明しています。

付録E WebAdminで使用できない文字

WebAdminで使用できない文字を説明しています。

付録F 障害調査情報の採取

初期調査のための情報の採取方法について説明しています。

輸出管理規制

本ドキュメントを輸出または第三者へ提供する場合は、お客様が居住する国および米国輸出管理関連法規等の規制をご確認のうえ、必要な手続きをおとりください。

出版年月および版数

2024年　4月　第2.1版
2022年　9月　第2版
2022年　2月　初版

著作権

Copyright 2021-2024 FUJITSU LIMITED

変更履歴

変更内容	変更箇所	版数
Mirroring Controllerのディスク監視のタイムアウトに関するパラメータ(disk_check_timeout、disk_check_max_threads)を追加しました。	2.11.4.4 ディスクの異常監視のチューニング	第2.1版
	A.4.1 データベースサーバのサーバ定義ファイル	
管理ネットワークに異常が発生し、レプリケーションが切断された場合の動作を指定するパラメータ(enable_promote_on_os_and_admin_network_error)を追加しました。	A.4.1 データベースサーバのサーバ定義ファイル	

目　次

第1章 データベース多重化運用の概要 1
1.1 データベース多重化運用とは 1
1.1.1 データベース多重化運用による監視 4
1.1.2 スタンバイサーバでの参照系業務 6
1.1.2.1 プライマリサーバで行う主業務を優先する場合 6
1.1.2.2 同期スタンバイサーバで参照系の業務を行う場合 6
1.2 データベース多重化運用のシステム構成 6
1.2.1 Mirroring Controllerの資源 8
1.2.1.1 データベースサーバの資源 8
1.2.1.2 裁定サーバの資源 9
1.2.2 Mirroring Controllerのプロセス 9
1.2.2.1 データベースサーバのプロセス 9
1.2.2.2 裁定サーバのプロセス 9
1.2.3 管理用/ログ転送用ネットワークの冗長化 10
1.2.4 CPUアーキテクチャと製品に関する注意 10
1.3 ハートビート異常検出時の動作の決定 10
1.4 データベース多重化におけるセキュリティ 11
1.4.1 スタンバイサーバの認証 12
1.4.2 スタンバイサーバに転送するトランザクションログの暗号化 13

第2章 データベース多重化運用のセットアップ 14
2.1 インストール 15
2.2 セットアップの準備作業 16
2.2.1 データベースサーバの準備作業 16
2.2.1.1 バックアップ用のディスクの準備 16
2.3 裁定サーバのセットアップ 16
2.3.1 裁定サーバの設定 16
2.3.2 裁定サーバのユーザーコマンドの作成 18
2.3.3 Mirroring Controller裁定プロセスの起動 19
2.4 プライマリサーバのセットアップ 19
2.4.1 プライマリサーバのデータベース多重化運用のセットアップ 19
2.4.2 プライマリサーバのインスタンスの作成・設定・登録 23
2.4.3 プライマリサーバのMirroring Controllerの起動 27
2.5 スタンバイサーバのセットアップ 27
2.5.1 スタンバイサーバのデータベース多重化運用のセットアップ 28
2.5.2 スタンバイサーバのインスタンスの作成・設定・登録 29
2.5.3 スタンバイサーバのMirroring Controllerの起動 30
2.6 データベースサーバのユーザーコマンドの作成 31
2.7 ストリーミングレプリケーションの状態確認 32
2.8 接続状態の確認 33
2.8.1 データベースサーバでの接続状態の確認 33
2.8.2 裁定サーバでの接続状態の確認 34
2.9 アプリケーションの作成 34
2.9.1 アプリケーションの接続先サーバの設定 34
2.10 動作確認 35
2.11 チューニング 35
2.11.1 データベース多重化運用を安定させるためのチューニング 35
2.11.2 スタンバイサーバでの問い合わせを安定させるためのチューニング 35
2.11.3 スタンバイサーバでの問い合わせを安定させるためのチューニング（プライマリサーバで頻繁に更新する場合） 35
2.11.4 最適な異常監視による縮退運転を行うためのチューニング 36
2.11.4.1 OS/サーバの異常監視のチューニング 36
2.11.4.1.1 裁定サーバを利用して自動縮退を行う運用の異常監視のチューニング 36
2.11.4.1.2 縮退の判断を行うユーザーコマンドを呼び出して自動縮退を行う運用の異常監視のチューニング 41
2.11.4.1.3 メッセージを通知する運用の異常監視のチューニング 43
2.11.4.1.4 ハートビート異常で無条件に自動縮退を行う運用の異常監視のチューニング 43

2.11.4.2 データベースプロセスの異常監視のチューニング....43
2.11.4.3 ストリーミングレプリケーションの異常監視のチューニング....45
2.11.4.4 ディスクの異常監視のチューニング....46
2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定....49
2.13 Mirroring Controller裁定プロセスの自動起動・停止の設定....51
2.14 バックアップ運用....52
2.14.1 データベース多重化運用情報のバックアップ....52
2.14.2 データベースのバックアップ運用....52

第3章 データベース多重化運用の操作....54
3.1 Mirroring Controller裁定プロセスの起動と停止....54
3.1.1 Mirroring Controller裁定プロセスの起動....54
3.1.2 Mirroring Controller裁定プロセスの停止....54
3.2 Mirroring Controllerの起動と停止....54
3.3 データベース多重化運用の状態確認....56
3.3.1 データベースサーバでの状態確認....56
3.3.2 裁定サーバでの状態確認....57
3.4 プライマリサーバの手動切り替え....58
3.5 スタンバイサーバの手動切り離し....58
3.6 ハートビート異常検出時の操作....59
3.7 Mirroring Controllerのメッセージ監視....59
3.8 サーバの保守....61
3.8.1 ローリングアップデート....61
3.8.2 保守停止....65
3.8.3 裁定サーバの保守....66
3.9 運用の変更....66
3.9.1 スタンバイサーバの停止を伴う変更....67
3.9.2 単体のサーバでの運用からデータベース多重化運用への変更....67
3.9.3 データベース多重化運用から単体のサーバでの運用への変更....68
3.9.4 裁定サーバを利用して自動縮退を行うデータベース多重化運用への変更....70
3.9.5 パラメータの変更....71
3.9.6 データベース多重化運用でのアンインストール....71

第4章 データベース多重化運用の異常時の対処....73
4.1 縮退運転になった場合の対処....73
4.1.1 切り替え発生後の縮退運転における運用操作....73
4.1.1.1 異常原因の特定とスタンバイサーバの再構築....75
4.1.1.1.1 Mirroring Controllerの停止....75
4.1.1.1.2 Mirroring Controller管理ディレクトリのリカバリ....75
4.1.1.1.3 異常原因の特定とリカバリ....75
4.1.1.2 スタンバイサーバの組込み....77
4.1.1.3 プライマリサーバの切り戻し....77
4.1.2 切り離し発生後の縮退運転における運用操作....78
4.1.2.1 異常原因の特定とスタンバイサーバの再構築....79
4.1.2.1.1 Mirroring Controllerの停止....79
4.1.2.1.2 Mirroring Controller管理ディレクトリのリカバリ....80
4.1.2.1.3 異常原因の特定とリカバリ....80
4.1.2.2 スタンバイサーバの組込み....80
4.1.3 縮退運転中の異常時の対処....80
4.2 自動切り替えに失敗した場合の対処....81
4.3 自動切り離しに失敗した場合の対処....81
4.4 すべてのデータベースサーバまたはインスタンスが停止した場合の対処....82
4.5 ユーザーの誤操作からのリカバリ....86

第5章 WebAdminを使用したMirroring Controllerの管理....88
5.1 Mirroring Controllerのセットアップ....89
5.2 Mirroring Controllerのセットアップを編集する....91
5.3 Mirroring Controllerの設定....92

5.4 Mirroring Controllerの停止 93
5.5 Mirroring Controllerの起動 94
5.6 フェイルオーバモードを無効にする 94
5.7 フェイルオーバモードを有効にする 94
5.8 Mirroring Controllerセットアップの削除 94
5.9 フェイルオーバ後のステータス更新 95
5.10 管理用ネットワーク兼ログ転送用ネットワークの異常時の対処 95
5.11 裁定サーバを利用して自動縮退を行う運用 96

付録A パラメータ 98
A.1 プライマリサーバで設定するパラメータ 98
A.2 スタンバイサーバで設定するパラメータ 100
A.3 ネットワーク定義ファイル 103
A.4 サーバ定義ファイル 106
A.4.1 データベースサーバのサーバ定義ファイル 106
A.4.2 裁定定義ファイル 114

付録B プライマリサーバとスタンバイサーバを同一サーバ内に構築する場合の補足事項 117
B.1 バックアップデータ格納先ディレクトリ 117
B.2 mc_ctlコマンドの実行方法 117

付録C ユーザーコマンド 119
C.1 フェンシングコマンド 119
C.2 裁定コマンド 120
C.3 状態遷移コマンド 121
C.3.1 切り替え後コマンド 121
C.3.2 切り離し前コマンド 122
C.3.3 組み込み後コマンド 122

付録D ハートビート異常で即時に自動縮退を行うを選択した場合の注意事項 124

付録E WebAdminで使用できない文字 127

付録F 障害調査情報の採取 128

索引 129

第1章 データベース多重化運用の概要

本章では、データベース多重化運用の概要について説明します。

ポイント

プロセスや管理ディレクトリなどの名称や説明に“Mirroring Controller”という単語を用いる場合があります。

1.1 データベース多重化運用とは

データベース多重化運用は、PostgreSQLのストリーミングレプリケーションに基づいた運用方式(ログシッピング方式)です。クラスタソフトウェアなどの他のソフトウェアを必要としません。

データベース多重化運用では、クラスタシステムを構成するすべてのサーバで複製されたデータベースを持ちます。更新を受け付けるサーバ(プライマリサーバ)から、データベースの更新トランザクションログをその他のサーバ(スタンバイサーバ)へと転送して、スタンバイサーバで更新トランザクションログを反映することで、これらの複製されたデータベースを維持します。クライアントドライバがプライマリサーバやスタンバイサーバを自動的に識別することで、アプリケーションは物理的なサーバを意識せずに透過的に接続できます。

また、データベースプロセス、ディスク、ネットワークなどのデータベース運用の継続に不可欠な要素の障害を検知する機能と簡易化されたスイッチオーバやスタンバイサーバの切り離し機能などを提供します。さらに、スタンバイサーバにおいて、参照系の業務を行うことができます。

複製方法には同期モードを用います。

注意

FUJITSU Enterprise Postgresでは、WebAdminまたはMirroring Controllerを使用する場合、1つのプライマリサーバと1つのスタンバイサーバで構成されたクラスタシステムをサポートしています。

- クラスタシステムに非同期スタンバイサーバを追加で接続することも可能ですが、これらのスタンバイサーバはMirroring Controllerの監視対象ではありません。
- クラスタシステムに同期スタンバイサーバを追加で接続することはできません。

参照

本マニュアルでは、ストリーミングレプリケーション機能については説明しません。

ストリーミングレプリケーション機能の詳細は、“PostgreSQL Documentation”の“High Availability, Load Balancing, and Replication”を参照してください。

図1.1 プライマリサーバからスタンバイサーバへのフェイルオーバ

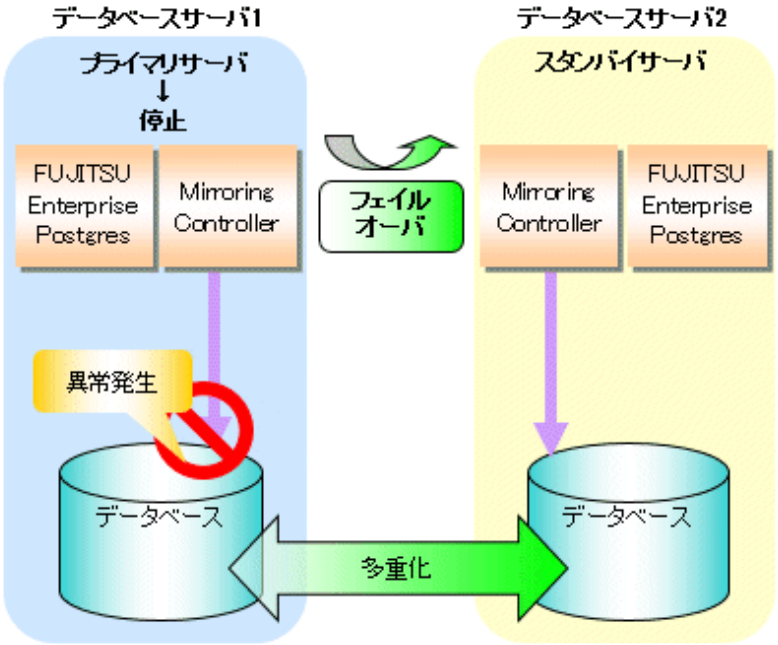
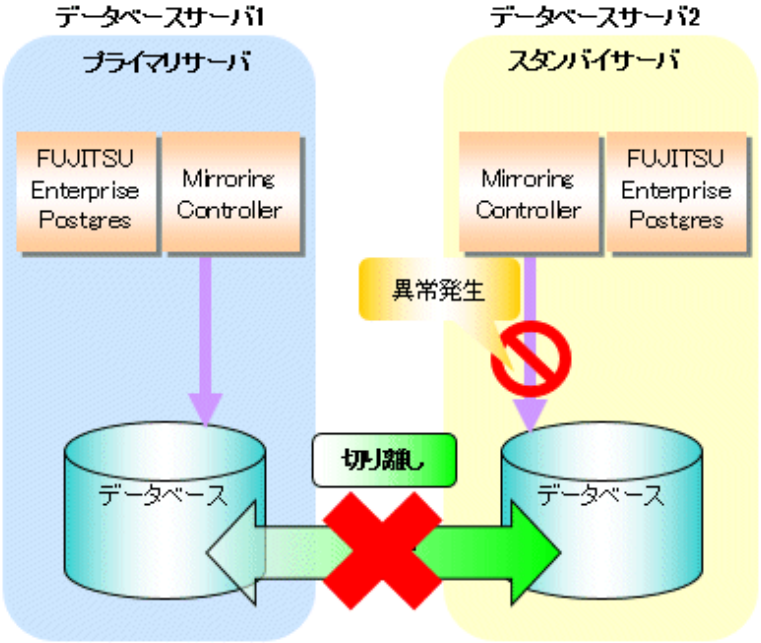


図1.2 スタンバイサーバの切り離し



裁定サーバによるデータベースの縮退

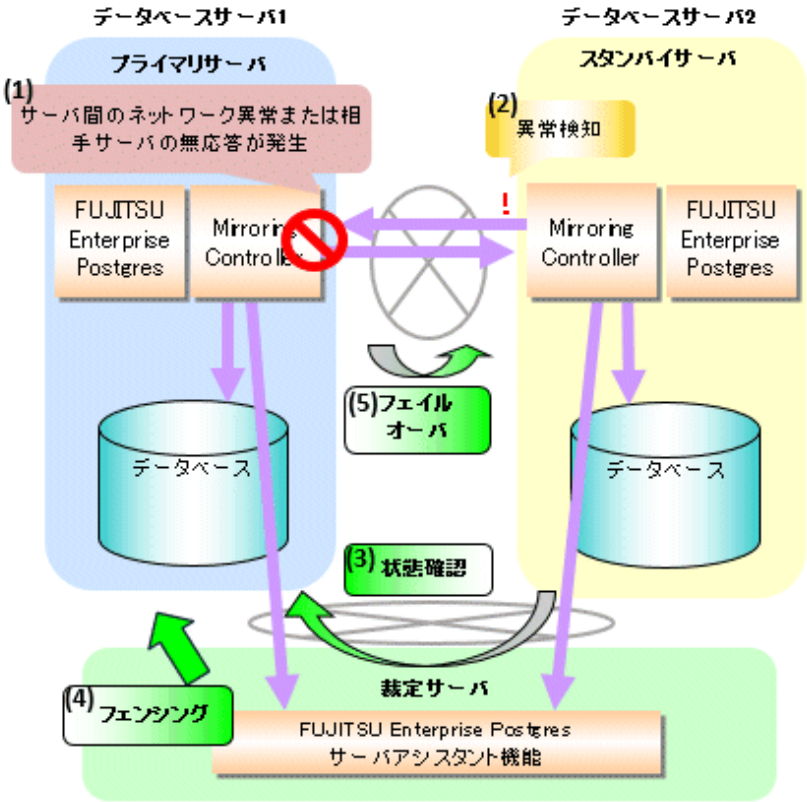
FUJITSU Enterprise Postgresでは、データベース多重化運用でデータベースサーバ間のネットワーク異常やサーバが不安定な状態になるなど、データベースサーバが相互の状態を正確に把握できない場合に、第三者としてデータベースサーバの状態を客観的に判断し、必要に応じて異常なデータベースの隔離を行う、サーバアシスタント機能を提供しています。この機能により、第三者としてデータベースサーバの状態を判断できることから、障害発生時に両サーバが同時にプライマリサーバとなる状態(スプリットブレイン)を防ぐことができます。サーバアシスタント機能をインストールしたサーバ(裁定サーバ)を利用することで、裁定サーバによるデータベースの縮退を行うことができます。

裁定サーバによるデータベースの縮退では、データベースサーバが相互の状態を確認できない場合に、裁定サーバにもう一方のデータベースサーバの状態を問い合わせます。裁定サーバは問い合わせを受けた時点で、状態が不安定であると判断した場合は、対象の

データベースサーバをクラスタシステムから隔離します(フェンシング)。裁定サーバはデータベースサーバからの問い合わせに即時に応答できるように、データベースサーバに対して定期的にハートビートを行っています。フェンシングは、Mirroring Controllerを使用している環境に応じて、処理をカスタマイズすることが可能です。

また、データベースサーバは、運用中にいつでも裁定サーバに確認依頼ができるように、裁定サーバに対して常にハートビートを行っています。これらの流れを以下の図に示します。

図1.3 裁定サーバによるデータベースの縮退



注意

裁定サーバは、データベースサーバとは異なる物理サーバに配置してください。裁定サーバを利用する場合のシステム構成については、“1.2 データベース多重化運用のシステム構成”を参照してください。

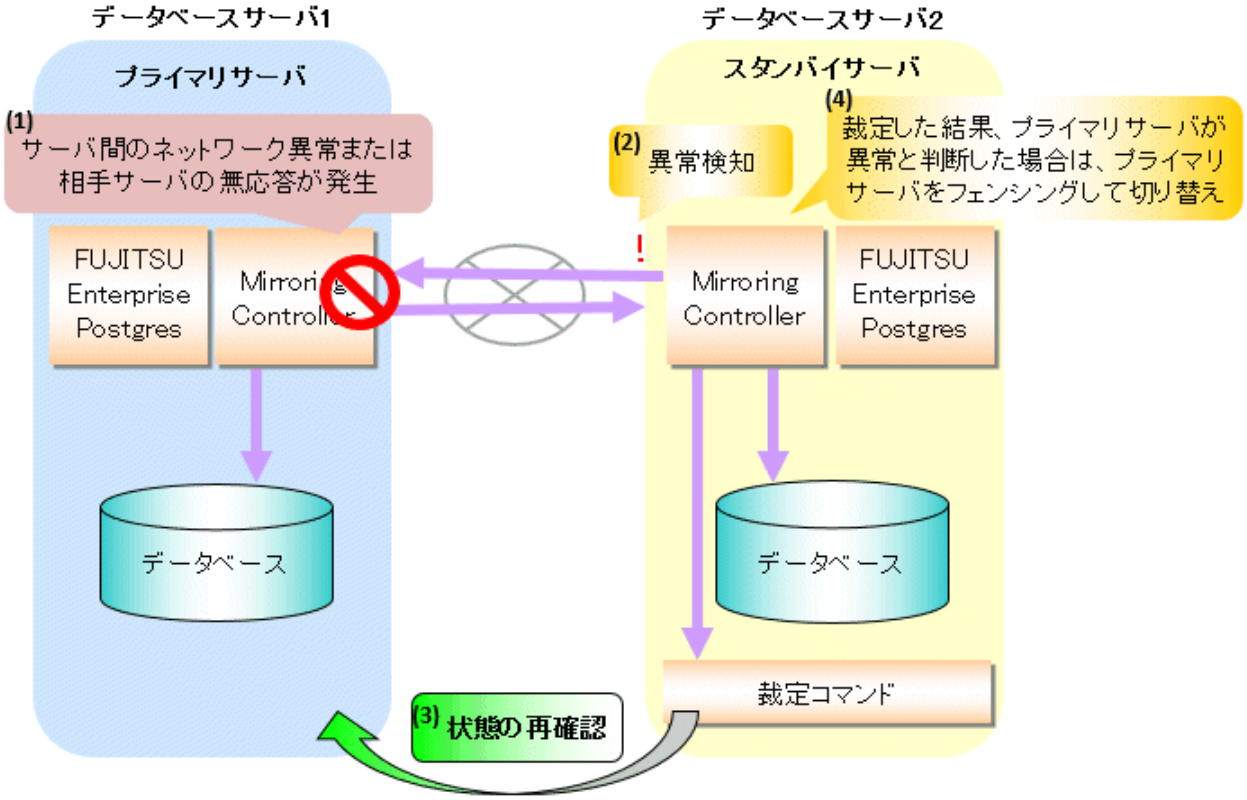
参照

フェンシングコマンドの詳細については、“2.3.2 裁定サーバのユーザーコマンドの作成”または“付録C ユーザーコマンド”を参照してください。

裁定コマンドを利用したデータベースの縮退

裁定コマンドは、裁定サーバの代わりに裁定処理を行うユーザーコマンドです。裁定サーバを配置できない場合は、裁定コマンドを使用することで、データベースサーバの裁定処理を行うことが可能です。これらの流れを以下の図に示します。

図1.4 裁定コマンドを利用したデータベースの縮退



参照

ユーザーコマンドの詳細については、“2.6 データベースサーバのユーザーコマンドの作成”または“付録C ユーザーコマンド”を参照してください。

1.1.1 データベース多重化運用による監視

データベース多重化運用では、以下の監視を行います。

- OSやサーバのダウンおよび無応答

  各サーバのMirroring Controller間でハートビートを行うことで、相互にOSやサーバの異常を検知します。

  データベース多重化運用を行う環境や運用方法に最適な動作を以下から選択が可能です。

  - 裁定サーバを利用して自動縮退(切り替え/切り離し)を行う

    デフォルトの動作です。

    裁定サーバがデータベースサーバの状態を客観的に判断し、状態が不安定なデータベースサーバのクラスタシステムからの隔離と縮退を行います。

    “裁定サーバによるデータベースの縮退”を参照してください。

  - 縮退の判断を行うユーザーコマンドを呼び出して自動縮退を行う

    物理的に裁定サーバを配置できない場合に、利用者が代わりに裁定処理を行うことが可能な場合に選択します。

    Mirroring Controllerがユーザーコマンドへ縮退有無の判断を問い合わせます。本コマンドでは、データベースサーバの状態を判断し、Mirroring Controllerに縮退を行うか否かを通知します。

    “裁定コマンドを利用したデータベースの縮退”を参照してください。

  - メッセージを通知する

    データベースサーバの2台構成でシステムを構築する場合に選択します。

Mirroring Controllerが異常検出時にメッセージをシステムログに出力します。そのため、ハートビート異常によりスプリットブレインを引き起こすことはありませんが、プライマリサーバでOSやサーバがダウンまたは無応答が発生した場合に自動切り替えが行われません。

－ ハートビート異常で無条件に自動縮退を行う

Mirroring Controllerがハートビート異常で無条件に自動縮退を行うため、推奨できません。

- データベースプロセスのダウンおよび無応答

  Mirroring Controllerが、定期的にデータベースのプロセスにアクセスして状態を確認します。アクセスの可否やタイムアウトの監視によって、プロセスの異常を検知します。

- ディスクの障害

  Mirroring Controllerが、定期的に以下のデータ格納先ディスクにファイルを作成し、I/Oエラーの有無によってディスクの異常を検知します。

  － データ格納先のディスク

  － トランザクションログ格納先のディスク

  － テーブル空間格納先のディスク

  検知可能な障害とは、ディスクヘッダや装置の電源不良などのシステム全体におよぶ物理的な障害です。

- ストリーミングレプリケーションの異常

  Mirroring Controllerが、定期的にPostgreSQLのシステムビューにアクセスすることで、ストリーミングレプリケーション(ログ転送用ネットワークおよびWAL送信/受信プロセス)の異常を検知します。

- Mirroring Controllerプロセスのダウンおよび無応答

  Mirroring Controllerでの監視処理を継続するために、Mirroring Controllerプロセスのダウンや無応答も合わせて監視しています。

  Mirroring Controller監視プロセスが、定期的にMirroring Controllerプロセスに問い合わせを行うことで、Mirroring Controllerプロセスのダウンおよび無応答を検知します。異常が検知されるとMirroring Controller監視プロセスにより自動でMirroring Controllerの再起動が行われます。

ポイント

- OSまたはサーバの生死監視においてハートビート異常を検出した場合の動作として、メッセージを出力する選択を行った場合には、ハートビート異常により自動縮退は行われません。
  しかし、プライマリサーバでWAL送信プロセスの異常を検知した場合には、スタンバイサーバを切り離すため、結果的にスタンバイサーバでOSやサーバのダウンまたは無応答が発生した場合でも自動切り離しが行われる場合があります。

- データベースプロセスの無応答やテーブル空間格納先のディスク障害を検知した場合にプライマリサーバの切り替えを行うか否かをパラメータで選択することができます。詳細は、“付録A パラメータ”を参照してください。

- スタンバイサーバを切り離した場合には、Mirroring Controllerが自動的にプライマリサーバのpostgresql.confファイルのsynchronous_standby_namesパラメータをコメントアウトします。これによって、プライマリサーバにおけるアプリケーション処理が停止することを防止します。

注意

プライマリサーバが切り替わって縮退運転になった場合には、もとのプライマリサーバは自動ではスタンバイサーバとして組込まれません。異常原因を取り除いてから組込みを実施してください。詳細は、“4.1 縮退運転になった場合の対処”を参照してください。

1.1.2 スタンバイサーバでの参照系業務

1.1.2.1 プライマリサーバで行う主業務を優先する場合

スタンバイサーバで参照系の業務を行っている場合、プライマリサーバの切り替えが発生すると、負荷や競合の観点で主業務に影響が出ることがあります。これは、スタンバイサーバから切り替わった新プライマリサーバにおいて、旧プライマリサーバで行っていた主業務と旧スタンバイサーバで継続している参照系の業務の両方を処理するためです。

そのため、主業務への影響を抑えるために参照系の業務を縮退させたい場合には、以下のユーザーコマンドを利用することで、旧スタンバイサーバで行っていた参照業務のコネクションを切断するなどの選択が可能となります。

- 切り替え後コマンド

注意

プライマリサーバの切り替え後も参照系の業務を継続して行う場合は、サーバリソースなどの見積りや性能への影響を十分考慮してください。

1.1.2.2 同期スタンバイサーバで参照系の業務を行う場合

Mirroring Controllerでは、ログ転送用ネットワークの障害などのプライマリサーバでの業務継続を阻害する異常の場合には、スタンバイサーバをクラスタシステムから自動で切り離す場合があります。

そのため、接続先が同期スタンバイサーバであることが前提の参照系の業務を動作させる場合、切り離されることによって想定されない過去データを参照しないために、以下のユーザーコマンドや機能を利用することで、業務を一時的に停止させるなどの選択が可能となります。

- 切り離し前コマンド
- 切り離し時のスタンバイサーバのインスタンス強制停止（サーバ定義ファイルのパラメータに指定）

また、スタンバイサーバがクラスタシステムに組み込まれた場合には、以下のユーザーコマンドを利用することで、参照系の業務を開始/再開させることが可能となります。

- 組み込み後コマンド

参照

- 各ユーザーコマンドの詳細については、“2.6 データベースサーバのユーザーコマンドの作成”または“付録C ユーザーコマンド”を参照してください。
- データベースサーバのサーバ定義ファイルについては、“A.4.1 データベースサーバのサーバ定義ファイル”を参照してください。

ポイント

上記のユーザーコマンドや機能の処理結果に関係なくMirroring Controllerは処理を継続します。

1.2 データベース多重化運用のシステム構成

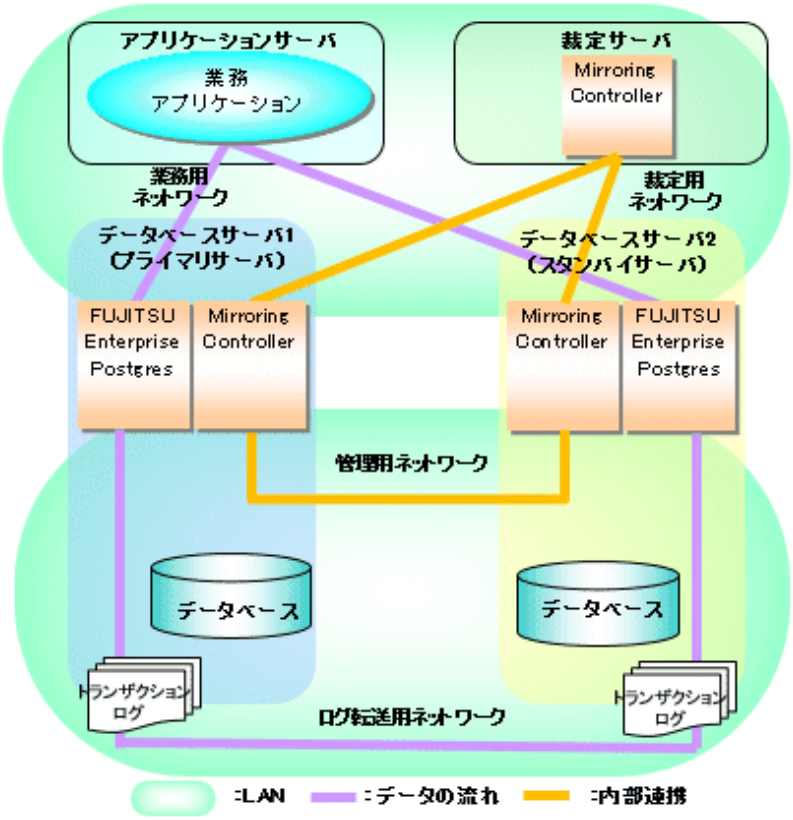
データベース多重化システムを構成する製品、機能、およびネットワークについて説明します。

データベース多重化システムは、以下のネットワークを使用します。

ネットワークの種類	目的
業務用ネットワーク	データベースにアクセスするアプリケーションとデータベースサーバ間のネットワークです。
裁定用ネットワーク	裁定サーバがプライマリサーバとスタンバイサーバの状態確認やデータベースサーバのMirroring Controllerと通信するためのネットワークです。また、業務用ネットワークが外部から切り離されている場合には、裁定用ネットワークと兼用す

ネットワークの種類	目的
	ることができます。ネットワークのセキュリティについては“1.4 データベース多重化におけるセキュリティ”を参照してください。
管理用ネットワーク	プライマリサーバおよびスタンバイサーバのMirroring Controllerによる相互監視や、他サーバのMirroring Controllerを制御するために使用するネットワークです。
ログ転送用ネットワーク	データベースを多重化するために、データベースのトランザクションログの転送などに使用するネットワークです。

図1.5 データベース多重化運用のシステム構成



裁定サーバは、第三者としてデータベースサーバの状態確認とフェンシングを行うことを目的に設置します。そのため、裁定サーバの効果を得るためには、以下の構成を考慮してください。

- 裁定サーバは、データベースサーバと異なるサーバに設置してください。
- 裁定用ネットワークは、管理用ネットワークやログ転送用ネットワークでの回線障害や負荷の影響を受けないネットワークを利用してください。管理用ネットワークやログ転送用ネットワークの異常を正しく判断するために必要です。

ポイント

- 裁定サーバは、アプリケーションサーバと兼用することが可能です。ただし、サーバの負荷に考慮してください。
- 裁定サーバは、他のクラスタシステムなどと連携し、冗長化することを推奨します。
- 裁定サーバは、プライマリサーバおよびスタンバイサーバと同一バージョンのFUJITSU Enterprise Postgresと組み合わせて使用してください。
- 裁定サーバはデータベースサーバと異なるプラットフォームに構築することも可能です。

注意

データベースサーバのハートビート監視はOSのpingコマンドを利用しているため、管理用ネットワークおよび裁定用ネットワークにおいてICMPが利用可能なようにネットワークの設定を行ってください。

1.2.1 Mirroring Controllerの資源

Mirroring Controllerのデータベースサーバと裁定サーバの資源について説明します。

1.2.1.1 データベースサーバの資源

Mirroring Controllerの資源は、Mirroring Controller管理ディレクトリのみです。このディレクトリには、Mirroring Controllerの動作を定義するファイルや動作時に一時的に作成するファイルを格納します。

注意

- Mirroring Controller管理ディレクトリは、FUJITSU Enterprise Postgresが管理するディレクトリと同じディレクトリ配下に作成しないでください。

  同じディレクトリ配下に作成すると、誤って削除されたり、FUJITSU Enterprise Postgresのリカバリとともに古いファイルに戻されたりするなどの予期しないトラブルのもとになります。

  FUJITSU Enterprise Postgresが管理するディレクトリの詳細は、“導入ガイド（サーバ編）”の“資源配置用のディレクトリの準備”を参照してください。

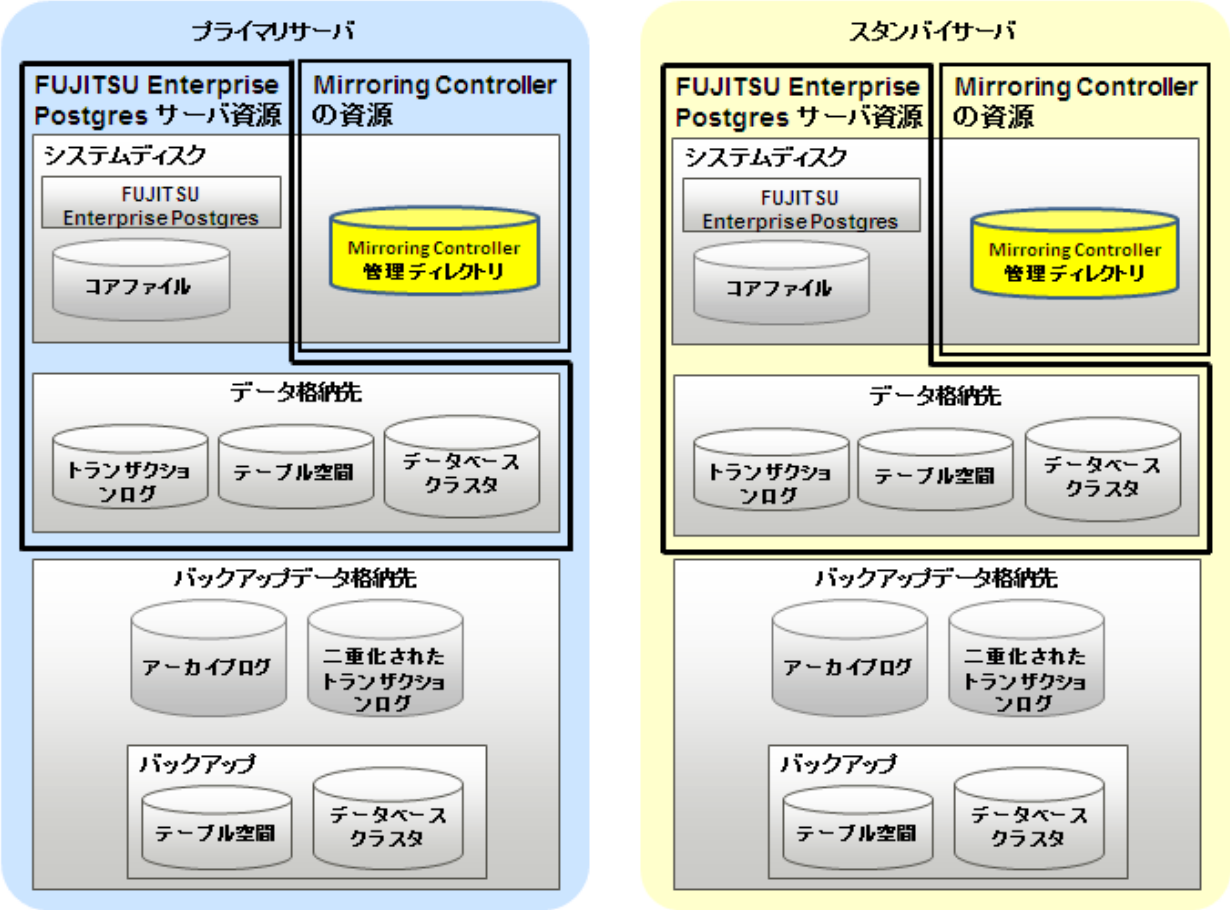
- Mirroring Controllerの資源は、“運用ガイド”の“データベースのバックアップ”にあるバックアップ方法では、バックアップされません。したがって、FUJITSU Enterprise Postgresのサーバ資源に加えて、利用者自身でバックアップを取得してください。なお、バックアップはMirroring Controllerを停止してから取得してください。

- 自動切り替え/切り離し機能を有効化した場合、Mirroring Controllerの監視対象のインスタンスに対してsynchronous_standby_namesパラメータの編集を行わないでください。それにより、編集後にMirroring Controllerの切り替えが行われるとデータが損失したり、編集直後にSQLアクセスが停止する場合があります。

- 仮想マシンやクラウド上で構築する場合は、仮想マシンが異なる物理サーバ上に配置されるようにしてください。仮想マシンの配置方法については、ご使用の仮想マシンのソフトウェアやクラウドベンダのドキュメントを参照してください。

バックアップの対象になるのは、プライマリサーバです。バックアップ対象のプライマリサーバは、切り替えや切り戻しなどの操作により、どのサーバで動作するのかは不明です。また、取得したバックアップデータをどのサーバで使用して復旧するのかも不明です。したがって、各サーバがプライマリサーバになったときにバックアップを取得してください。

図1.6 Mirroring Controllerの資源をバックアップする場合の構成



1.2.1.2 裁定サーバの資源

裁定サーバの資源は、Mirroring Controller裁定プロセスの管理ディレクトリのみです。このディレクトリには、Mirroring Controller裁定プロセスの動作を定義するファイルや、動作時に一時的に作成するファイルを格納します。

1.2.2 Mirroring Controllerのプロセス

Mirroring Controllerのデータベースサーバと裁定サーバのプロセスについて説明します。

1.2.2.1 データベースサーバのプロセス

データベースサーバのプロセスは、Mirroring ControllerプロセスとMirroring Controller監視プロセスから構成されます。

プロセスの種類	目的
Mirroring Controllerプロセス	データベースサーバ間でOS/サーバ、プロセスの生死監視、およびディスクの異常監視を行います。また、裁定サーバへの裁定要求を行います。
Mirroring Controller監視プロセス	Mirroring Controllerプロセスの生死監視を行います。Mirroring Controllerプロセスで無応答やダウンが発生した場合は、監視プロセスが自動で再起動を行います。

1.2.2.2 裁定サーバのプロセス

裁定サーバのプロセスは、Mirroring Controller裁定プロセスのみです。

プロセスの種類	目的
Mirroring Controller裁定プロセス	プライマリサーバおよびスタンバイサーバが検知した異常について再確認を行います。また、プライマリサーバおよびスタンバイサーバに異常があると判断した場合は、フェンシングを行います。

1.2.3 管理用/ログ転送用ネットワークの冗長化

管理用ネットワークは、Mirroring Controllerが各サーバの状態を確認しあうための重要なネットワークです。

また、ログ転送用ネットワークは、データの鮮度を維持するための重要なネットワークです。

したがって、これらにはOSやネットワークドライバベンダが提供するネットワークの冗長化機能のチャンネルボンディング機能を使用して、障害に強いネットワーク構成にしてください。

1.2.4 CPUアーキテクチャと製品に関する注意

データベース多重化システムのログ転送先に、PostgreSQLのストリーミングレプリケーションのみを設定したサーバを指定することはできません。

1.3 ハートビート異常検出時の動作の決定

データベース多重化運用を行う環境や運用方法に合わせて、OS/サーバの生死監視によるハートビート異常検出時の動作を決定します。

以下の4つから選択が可能であり、Mirroring Controllerのパラメータで設定します。

- 裁定サーバによる自動縮退(切り替え/切り離し)
- 縮退の判断を行うユーザーコマンドを呼び出して自動縮退
- メッセージ通知
- 無条件に自動縮退(切り替え/切り離し)

OS/サーバの生死監視が検知する異常発生時のプライマリサーバでの業務継続の可否は以下のとおりです。

OS/サーバの生死監視が検知する異常発生時の業務継続の可否

動作	異常事象				
	サーバ/OSのダウンまたは無応答		管理用ネットワークの異常	ログ転送用ネットワークの異常	管理用ネットワーク兼ログ転送用ネットワークの異常
	プライマリサーバ	スタンバイサーバ			
裁定サーバによる自動縮退	○(切り替え)	○(切り離し)	○	○(切り離し)	○(切り離し)
ユーザーコマンドを呼び出して自動縮退	○(切り替え)	○(切り離し)	○	○(切り離し)	○(切り離し)
メッセージ通知	×(メッセージ通知のみ)	×(メッセージ通知のみ)	○	○(切り離し)	○(切り離し)
無条件に自動縮退	○(切り替え)	○(切り離し)	×(スプリットブレインの発生)	○(切り離し)	×(スプリットブレインの発生)

○:業務継続可

×:業務継続不可

1.4 データベース多重化におけるセキュリティ

データベースサーバは、クラスタシステムを構成するすべてのサーバで複製されたデータベースを持ちます。プライマリサーバとスタンバイサーバのあいだでデータベースの更新トランザクションログを転送して、スタンバイサーバでその更新トランザクションログを反映することで、これらの複製されたデータベースを維持しています。

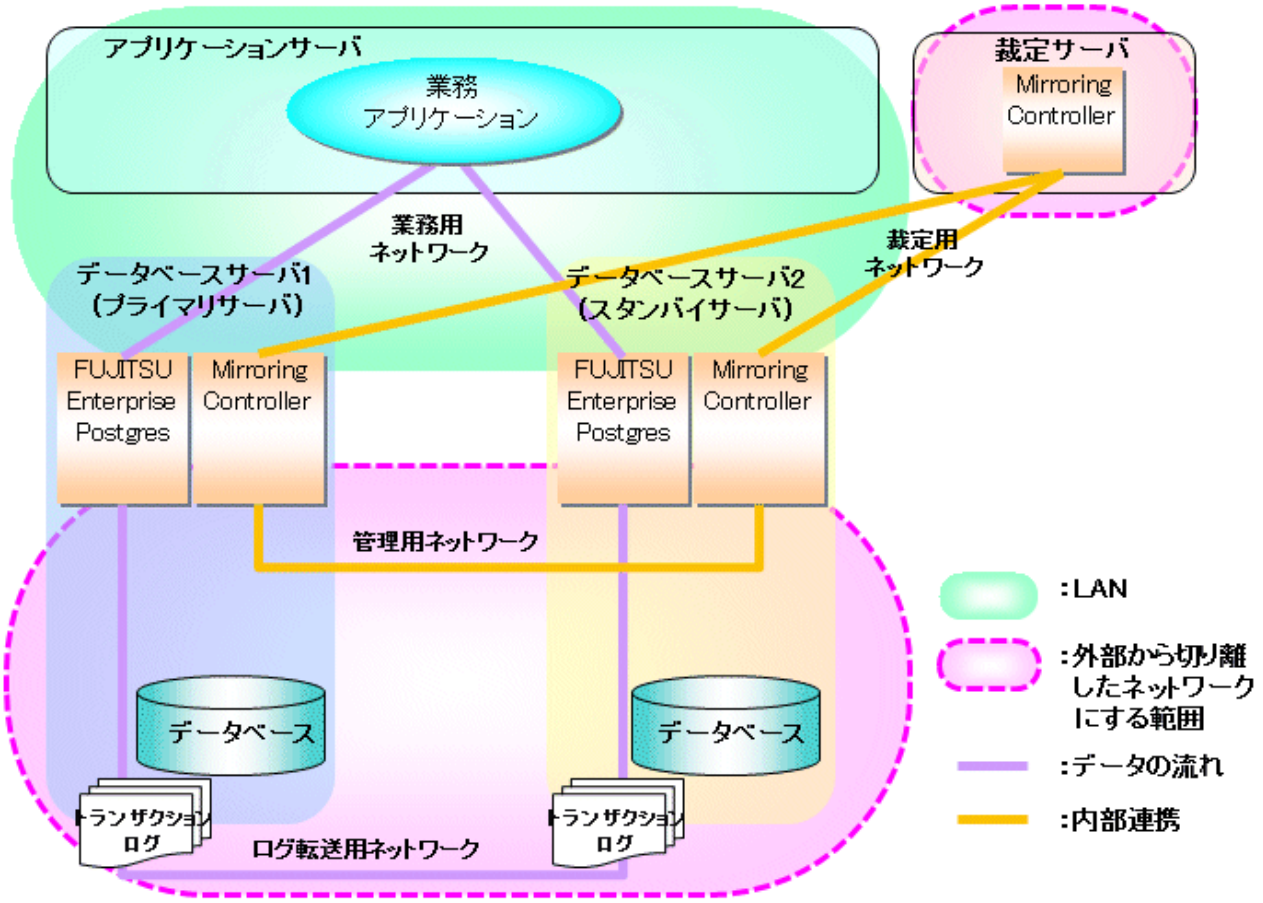
データベースへの不正アクセスからの防御やトランザクションログの転送時におけるデータの機密保護が必要な環境でデータベース多重化運用を行う場合には、以下の点に注意してセキュリティを考慮してください。

- レプリケーション接続時にはtrust認証を使用しないでください
- 管理用ネットワークとログ転送用ネットワークは、図1.7 セキュリティのように外部から切り離したネットワークとして構築してください。

また、Mirroring Controllerがデータベースサーバから裁定サーバへ接続する回線は、以下の点に注意し、セキュリティを考慮してください。

- 図1.7 セキュリティのように、裁定サーバを外部から切り離したネットワークとして構築してください。

図1.7 セキュリティ



しかし、例えば、ネットワーク遅延の影響が少ない近接地の事業所にサーバを配置したい場合など、上記の構成に当てはまらない場合も考えられます。

この場合には、以下の機能を組み合わせてセキュリティを確保してください。

- スタンバイサーバの認証
- スタンバイサーバに転送するトランザクションログの暗号化

これらを組み合わせた場合のセキュリティを以下に示します。

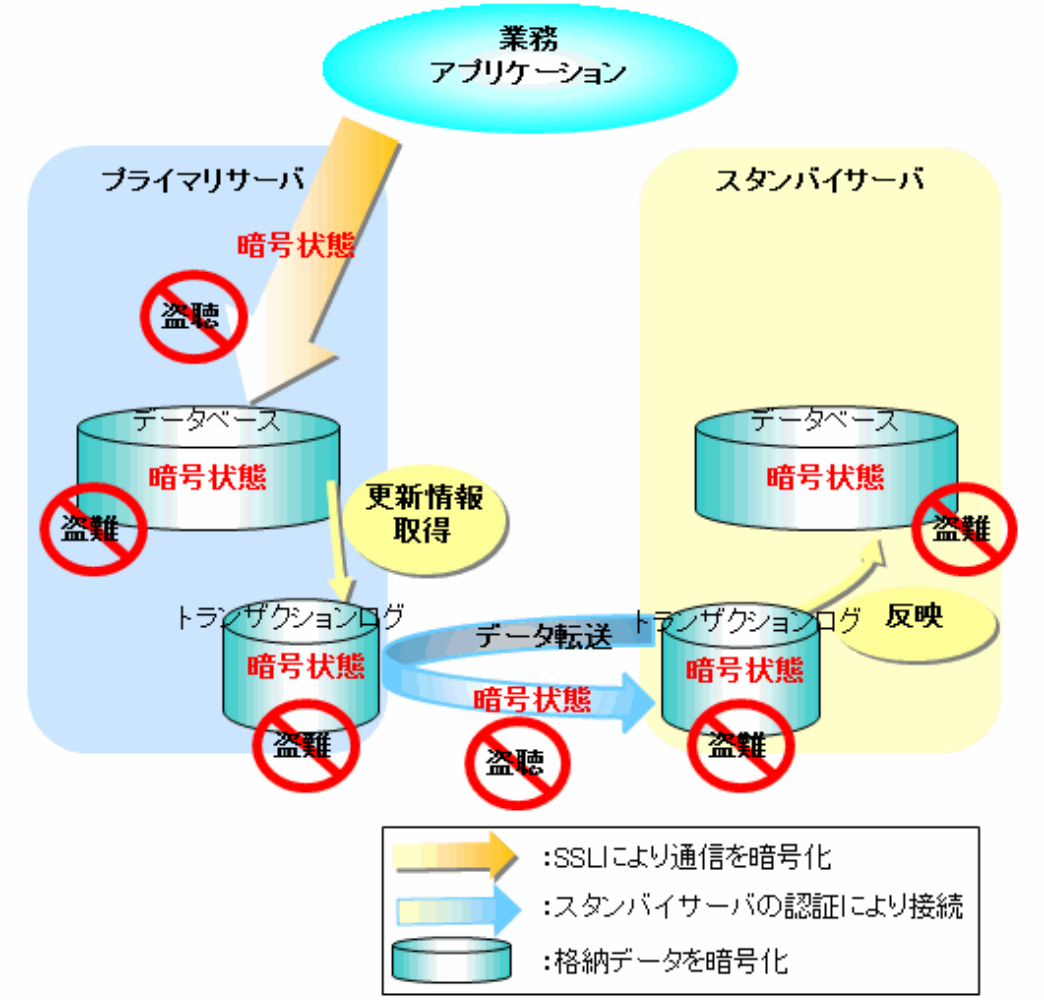
ポイント

業務用ネットワークが外部から切り離されている場合は、業務用ネットワークを裁定用ネットワークとして利用することが可能です。ただし、業務用ネットワークと裁定用ネットワークを兼用する場合は、ネットワークの負荷について考慮が必要です。

注意

ファイアウォールなどでポートが塞がれている(アクセス許可されていない)場合は、使用するポートに対し、アクセスできるように許可してください。ポートを開ける(アクセス許可を与える)方法については、ご使用のベンダーのドキュメントを参照してください。なお、ポートを開ける際には、十分セキュリティ上のリスクを考慮してください。

図1.8 スタンバイサーバの認証と転送するトランザクションログの暗号化によるセキュリティ



参照

SSLによる通信の暗号化の詳細は、“運用ガイド”の“Secure Sockets Layerによる安全な通信の構成”の“データベース多重化運用を行う場合”を参照してください。

1.4.1 スタンバイサーバの認証

外部のスタンバイサーバになりすましたサーバからの接続を、ユーザー名とパスワードで認証することによって防御します。

プライマリサーバのpg_hba.confファイルに、スタンバイサーバからの接続をクライアントからの接続として認証するように設定します。

参照

設定内容の詳細は、“PostgreSQL Documentation”の“Client Authentication”を参照してください。

1.4.2 スタンバイサーバに転送するトランザクションログの暗号化

スタンバイサーバの認証が破られて、悪意のあるユーザーがなりすましたスタンバイサーバでデータを参照してしまうような場合に、トランザクションログのデータを解読できないように暗号化して防御します。この暗号化には、透過的データの暗号化機能を使用します。

参照

詳細は、“運用ガイド”の“透過的データ暗号化による格納データの保護”を参照してください。

第2章 データベース多重化運用のセットアップ

本章では、データベース多重化運用のセットアップの手順および確認方法について、説明します。

データベースサーバでセットアップと操作を行うユーザー

データベースサーバのセットアップと操作は、インスタンス管理者ユーザーで行います。

裁定サーバでセットアップと操作を行うユーザー

裁定サーバを利用して自動縮退を行う運用の場合に、裁定サーバでセットアップと操作を行うユーザーは以下です。

- 任意のOSユーザー

ポイント

- Mirroring Controllerがデータベースのインスタンスに接続するユーザーとして任意のデータベーススーパーユーザーを選択することで、Mirroring Controllerのコマンドを操作するインスタンス管理者ユーザーとデータベーススーパーユーザーが異なる環境においても、データベース多重化運用を行うことが可能です。
- データベースのインスタンスへ接続するアプリケーション名は‘mc_agent’です。

システムの時刻合わせ

セットアップを開始する前に、OSの時刻合わせ機能などを使用してプライマリサーバ、スタンバイサーバおよび裁定サーバの時刻を合わせてください。

許容誤差範囲は、1秒程度です。

許容誤差範囲を超えるような時刻のずれがあると、トラブル発生時の調査に影響することがあります。

ICMPの設定

データベースサーバのハートビート監視はOSのpingコマンドを利用しているため、管理用ネットワークおよび裁定用ネットワークにおいてICMPが利用可能なようにネットワークの設定を行ってください。詳細は各オペレーティングシステムに従ってください。

セットアップ

以下にセットアップの手順を示します。ただし、裁定サーバの手順については、裁定サーバを利用して自動縮退を行う運用の場合にのみ実施します。また、プライマリサーバおよびスタンバイサーバの手順において、裁定サーバの利用有無によって手順を区別しています。

手順	作業項目			参照先
	プライマリサーバ	スタンバイサーバ	裁定サーバ	
1	インストール			2.1 インストール
2	データベースサーバの準備作業		裁定サーバの準備作業	2.2 セットアップの準備作業
3			裁定サーバの設定	2.3.1 裁定サーバの設定
4			ユーザーコマンドの作成	2.3.2 裁定サーバのユーザーコマンドの作成
5			裁定プロセスの起動	2.3.3 Mirroring Controller裁定プロセスの起動
6	データベース多重化運用のセットアップ			2.4.1 プライマリサーバのデータベース多重化運用のセットアップ
7	インスタンスの作成・設定・登録			2.4.2 プライマリサーバのインスタンスの作成・設定・登録

手順	作業項目			参照先
	プライマリサーバ	スタンバイサーバ	裁定サーバ	
8	ユーザーコマンドの作成			2.6 データベースサーバのユーザーコマンドの作成
9	Mirroring Controllerの起動			2.4.3 プライマリサーバのMirroring Controllerの起動
10		データベース多重化運用のセットアップ		2.5.1 スタンバイサーバのデータベース多重化運用のセットアップ
11		インスタンスの作成・設定・登録		2.5.2 スタンバイサーバのインスタンスの作成・設定・登録
12		ユーザーコマンドの作成		2.6 データベースサーバのユーザーコマンドの作成
13		Mirroring Controllerの起動		2.5.3 スタンバイサーバのMirroring Controllerの起動
14	ストリーミングレプリケーションの状態確認			2.7 ストリーミングレプリケーションの状態確認
15	接続状態の確認			2.8.1 データベースサーバでの接続状態の確認
16		接続状態の確認		2.8.1 データベースサーバでの接続状態の確認
17			接続状態の確認	2.8.2 裁定サーバでの接続状態の確認
18	アプリケーションの作成			2.9 アプリケーションの作成
19	動作確認			2.10 動作確認

以降で各手順の説明を行います。

参考

- 単一サーバでの運用をデータベース多重化運用に変更する場合のセットアップ手順も同様です。この場合、FUJITSU Enterprise Postgresのインストールやインスタンスの作成は省略してください。

  詳細は、“3.9.2 単体のサーバでの運用からデータベース多重化運用への変更”を参照してください。

- システムテストなどの目的のために、プライマリサーバとスタンバイサーバを擬似的に同一のサーバ内に構築できます。この場合も同様の手順でセットアップを行いますが、補足事項があります。

  セットアップする前に、“付録B プライマリサーバとスタンバイサーバを同一サーバ内に構築する場合の補足事項”を参照してください。

2.1 インストール

以下のマニュアルを参照して、インストールしてください。

参照

- FUJITSU Enterprise Postgresのインストールについては、“導入ガイド(サーバ編)”を参照してください。
- 裁定サーバへのサーバアシスタント機能のインストールについては、“導入ガイド(サーバアシスタント編)”を参照してください。

注意

裁定サーバはデータベースサーバと兼用しないでください。裁定サーバは、第三者としてデータベースサーバの状態確認とフェンシングを行うことを目的に設置します。データベースサーバと裁定サーバを兼用した場合、裁定サーバの効果を得ることはできません。

2.2 セットアップの準備作業

Mirroring Controllerをセットアップする前に行う準備作業について説明します。

2.2.1 データベースサーバの準備作業

2.2.1.1 バックアップ用のディスクの準備

Mirroring Controllerではバックアップ運用を行うことで、万が一すべてのサーバのディスクが破壊されるような事態となっても復旧できます。

バックアップの対象は、プライマリサーバです。しかし、切り替えや切り戻しによってスタンバイサーバも、プライマリサーバになる可能性があります。したがって、プライマリサーバとスタンバイサーバのそれぞれにバックアップ用のディスク装置を用意してください。バックアップは、バックアップを実施する時点のプライマリサーバで実施します。

2.3 裁定サーバのセットアップ

裁定サーバのセットアップについて説明します。

2.3.1 裁定サーバの設定

裁定サーバのデータベース多重化運用のセットアップについて説明します。

データベース多重化運用では、動作に必要なファイルをMirroring Controller裁定プロセスの管理ディレクトリで管理します。

1つの裁定プロセスに対して1つのMirroring Controller裁定プロセスの管理ディレクトリが対応します。

ポイント

1つの裁定サーバに、データベース多重化システムごとの裁定プロセスをそれぞれ起動することができます。

参照

- mc_arbコマンドの詳細は、“リファレンス”を参照してください。
- セットアップで編集する各パラメータの詳細については、“付録A パラメータ”を参照してください。

以下の手順で行います。

1. 裁定サーバに、裁定プロセスの起動や停止を行う任意のOSユーザーでログインします。
2. 環境変数の設定

   以下の環境変数を設定します。

   － PATH

   インストールディレクトリ/binを追加します。

   － MANPATH

   インストールディレクトリ/share/manを追加します。

－ LD_LIBRARY_PATH

インストールディレクトリ/libを追加します。

例

インストールディレクトリが“/opt/fsepv<*x*>assistant”の場合の環境変数の設定例を示します。“<*x*>”は製品のバージョンです。

sh、bashの場合

```
$ PATH=/opt/fsepv<x>assistant/bin:$PATH ; export PATH
$ MANPATH=/opt/fsepv<x>assistant/share/man:$MANPATH ; export MANPATH
$ LD_LIBRARY_PATH=/opt/fsepv<x>assistant/lib:$LD_LIBRARY_PATH ; export LD_LIBRARY_PATH
```

csh、tcshの場合

```
$ setenv PATH /opt/fsepv<x>assistant/bin:$PATH
$ setenv MANPATH /opt/fsepv<x>assistant/share/man:$MANPATH
$ setenv LD_LIBRARY_PATH /opt/fsepv<x>assistant/lib:$LD_LIBRARY_PATH
```

注意

LD_LIBRARY_PATHの設定後、FUJITSU Enterprise Postgres以外のコマンド(OSコマンドなど)を実行する場合は、LD_LIBRARY_PATHからインストールディレクトリ/libを削除してください。

3. 裁定サーバで必要なファイルを格納するMirroring Controller裁定プロセスの管理ディレクトリを作成します。

   Mirroring Controller裁定プロセスの管理ディレクトリは、ASCII文字で指定してください。

4. Mirroring Controller裁定プロセスが管理するMirroring Controllerのネットワークの構成をネットワーク定義ファイル(network.confファイル)に定義します。

   サンプルファイルをもとにして、Mirroring Controller裁定プロセスの管理ディレクトリにnetwork.confファイルを作成します。network.confファイルには、手順1の裁定プロセスの起動や停止を行うOSユーザーにのみ読み込み権限と書き込み権限を設定してください。

   手順1の裁定プロセスの起動や停止を行うOSユーザー以外にアクセス権限が与えられていた場合には、mc_arbコマンドは動作しません。これにより、手順1の裁定プロセスの起動や停止を行うOSユーザー以外がMirroring Controller裁定プロセスを操作することを防止します。

   サンプルファイルのパス

   ```
   /インストールディレクトリ/share/mcarb_network.conf.sample
   ```

   network.confファイルには、プライマリサーバとスタンバイサーバのIPアドレスまたはホスト名、およびポート番号を指定して、Mirroring Controller裁定プロセスが管理するMirroring Controllerのネットワークの構成を定義します。

   詳細は、“A.3 ネットワーク定義ファイル”を参照してください。

   以下に定義例を示します。

   例）
   サーバ識別子に“server1”および“server2”、ポート番号に“27541”を指定している場合の例を以下に示します。

   ```
   server1 192.0.3.100 27541
   server2 192.0.3.110 27541
   ```

5. Mirroring Controller裁定プロセスの制御に関する情報を裁定定義ファイル(arbitration.confファイル)に定義します。

   サンプルファイルをもとにして、Mirroring Controller裁定プロセスの管理ディレクトリにarbitration.confファイルを作成します。arbitration.confファイルには、手順1の裁定プロセスの起動や停止を行うOSユーザーにのみ読み込み権限と書き込み権限を設定してください。手順1の裁定プロセスの起動や停止を行うOSユーザー以外にアクセス権限が与えられていた場合には、mc_arbコマンドは動作しません。

   サンプルファイルのパス

```
/インストールディレクトリ/share/mcarb_arbitration.conf.sample
```

arbitration.confファイルには、以下のパラメータを設定します。

表2.1 設定するパラメータ

パラメータ	指定内容	備考
port	Mirroring Controller裁定プロセスのポート番号	0～65535の範囲を超えて指定できません。他のソフトウェアと重ならないように設定してください。また、他のプログラムによって一時的に割り当てられる可能性のあるエフェメラルポートを指定しないでください。
my_address	'データベースサーバのMirroring Controllerプロセスからの接続を受け付けるIPアドレス、またはホスト名' [設定例] my_address = ‘192.0.3.120’	IPアドレスはIPv4、IPv6の指定が可能です。 半角シングルクォートで囲んで指定してください。
syslog_ident	'プログラム名'	システムログ内のMirroring Controller裁定プロセスのメッセージを特定するために使用するプログラム名を半角シングルクォートで囲んで指定します。半角空白以外のASCII文字で指定してください。 デフォルトは、'MirroringControllerArbiter'です。
fencing_command	'フェンシングコマンドのファイルパス名' [設定例] fencing_command = '/arbiter/fencing_dir/execute_fencing.sh'	異常と判断したデータベースサーバをフェンシングするフェンシングコマンドのファイルパス名を指定します。 ファイルパス名は絶対パスで指定し、半角シングルクォートで囲んでください。 1024バイト未満の長さで指定してください。
fencing_command_timeout	フェンシングコマンドのタイムアウト時間(秒数)	指定された秒数以上のあいだ、コマンドの応答がない場合は、フェンシングに失敗したと判断し、フェンシングコマンドの実行プロセスに対して、シグナル(SIGTERM)を送信します。 1～2147483647の間の数値で指定してください。 デフォルトは20秒です。

参考

各パラメータの詳細およびその他のパラメータについては、“A.4.2 裁定定義ファイル”を参照してください。

2.3.2 裁定サーバのユーザーコマンドの作成

裁定サーバのユーザーコマンドは、フェンシングコマンドのみです。

フェンシングコマンドは、Mirroring Controllerが裁定処理を行った結果、不安定なデータベースサーバと判断した場合に、Mirroring Controller裁定プロセスから呼び出されるユーザーコマンドです。

フェンシングコマンドには、対象のOS/サーバを停止させるなどして、データベースサーバをクラスタシステムから隔離する処理を利用者が実装します。作成したフェンシングコマンドは、裁定定義ファイル内のパラメータに指定します。パラメータの詳細については、“A.4.2 裁定定義ファイル”を参照してください。

- 切り替え時にプライマリサーバをフェンシングする場合
  - プライマリサーバのMirroring Controller管理プロセスが相手サーバのMirroring Controller管理プロセスと通信できないようにします。
  - アプリケーションからプライマリサーバのインスタンスに接続できないようにします。
- 切り離し時にスタンバイサーバをフェンシングする場合
  - スタンバイサーバのMirroring Controller管理プロセスが相手サーバのMirroring Controller管理プロセスと通信できないようにします。
  - アプリケーションからスタンバイサーバのインスタンスに接続できないようにします。
  - スタンバイサーバがストリーミングレプリケーションを継続できないようにします。

参照

ユーザーコマンドの詳細については、“付録C ユーザーコマンド”を参照してください。

2.3.3 Mirroring Controller裁定プロセスの起動

Mirroring Controller裁定プロセスの起動について説明します。

Mirroring Controller裁定プロセスの起動は、裁定サーバにログインしたOSユーザーで、mc_arbコマンドをstartモードで実行して起動します。

例）

```
$ mc_arb start -M /mcarb_dir/arbiter1
```

2.4 プライマリサーバのセットアップ

プライマリサーバのセットアップについて、説明します。

2.4.1 プライマリサーバのデータベース多重化運用のセットアップ

プライマリサーバのデータベース多重化運用のセットアップについて、説明します。

データベース多重化機能では、動作に必要なファイルをMirroring Controller管理ディレクトリで管理します。

1つのインスタンスに対して1つのMirroring Controller管理ディレクトリが対応します。

注意

- Mirroring Controller管理ディレクトリは、FUJITSU Enterprise Postgresが管理するディレクトリとともに誤って削除されたり、FUJITSU Enterprise Postgresのリカバリとともに古いファイルに戻されたりすることがないようにFUJITSU Enterprise Postgresが管理するディレクトリ配下に配置しないでください。

参照

- FUJITSU Enterprise Postgresが管理するディレクトリの詳細は、“導入ガイド(サーバ編)”の“セットアップ”の“資源配置用のディレクトリの準備”を参照してください。
- mc_ctlコマンドの詳細は、“リファレンス”を参照してください。
- セットアップで編集する各パラメータの詳細については、“付録A パラメータ”を参照してください。

以下の手順で行います。

1. プライマリサーバにログインします。
2. データベース多重化機能で必要なファイルを格納するMirroring Controller管理ディレクトリを作成します。

   Mirroring Controller管理ディレクトリは、ASCII文字で指定してください。

   また、インスタンス管理者ユーザーに対する書き込み権限を付与してください。
3. Mirroring Controllerのプロセス間で連携するネットワークの構成をネットワーク定義ファイル(network.confファイル)に定義します。

   サンプルファイルをもとにして、Mirroring Controller管理ディレクトリにnetwork.confファイルを作成します。network.confファイルには、インスタンス管理者ユーザーにのみ読み込み権限と書き込み権限を設定してください。

   インスタンス管理者ユーザー以外にアクセス権限が与えられていた場合には、mc_ctlコマンドは動作しません。これにより、インスタンス管理者ユーザー以外がMirroring Controllerを操作することを防止します。

   サンプルファイルのパス

```
/インストールディレクトリ/share/mc_network.conf.sample
```

   network.confファイルには、プライマリサーバとスタンバイサーバのIPアドレスまたはホスト名、およびポート番号を指定して、Mirroring ControllerのプロセスとMirroring ControllerプロセスとMirroring Controller裁定プロセスが連携するためのネットワークの構成を定義します。

   詳細は、“A.3 ネットワーク定義ファイル”を参照してください。

   以下に定義例を示します。

   ハートビート異常検出時の動作の設定値によって、定義内容が異なります。

   裁定サーバによる自動縮退を選択した場合

   例）
   プライマリサーバ、スタンバイサーバのサーバ識別子に“server1”および“server2”、ポート番号に“27540”“27541”、また、Mirroring Controller裁定プロセスのサーバ識別子に“arbiter”、ポート番号に“27541”を指定している場合の例を以下に示します。

```
server1 192.0.2.100,192.0.3.100 27540,27541 server
server2 192.0.2.110,192.0.3.110 27540,27541 server
arbiter 192.0.3.120 27541 arbiter
```

   ポート番号は、プライマリサーバ、スタンバイサーバ、裁定サーバのどのサーバにおいても、他のソフトウェアと重ならないように設定してください。また、裁定サーバを利用して自動縮退を行う運用の場合は、裁定用ネットワークが管理用ネットワークの回線障害の影響を受けないネットワークを利用してください。

   サーバ種別が“server”の場合、IPアドレスまたはホスト名、およびポート番号を以下の順序で指定する必要があります。

   - 管理用ネットワークとして使用するデータベースサーバ側のIPアドレスまたはホスト名
   - 裁定用ネットワークとして使用するデータベースサーバ側のIPアドレスまたはホスト名
   - 管理用ネットワークとして使用するデータベースサーバ側のポート番号
   - 裁定用ネットワークとして使用するデータベースサーバ側のポート番号

   サーバ種別が“arbiter”の場合、裁定サーバのarbitration.confファイルのmy_addressパラメータで指定したIPアドレスまたはホスト名とportパラメータに指定したポート番号を指定する必要があります。

   裁定サーバによる自動縮退以外を選択した場合

   例）
   サーバ識別子に“server1”および“server2”、ポート番号に“27540”を指定している場合の例を以下に示します。

```
server1 192.0.2.100 27540
server2 192.0.2.110 27540
```

   WebAdminのように/etc/servicesファイルを用いて空きポート番号を検索して使用するプログラムが存在するため、プライマリサーバ側のポート番号を/etc/servicesファイルに登録しておいてください。

   なお、サービス名は、任意の名前で登録してください。

4. Mirroring Controllerの監視および制御に関する情報をサーバ識別子.confファイルに定義します。

サンプルファイルをもとにして、Mirroring Controller管理ディレクトリにサーバ識別子.confファイルを作成します。サーバ識別子.confファイルには、インスタンス管理者ユーザーにのみ読み込み権限と書き込み権限を設定してください。インスタンス管理者ユーザー以外にアクセス権限が与えられていた場合には、mc_ctlコマンドは動作しません。

サーバ識別子.confファイルのファイル名には、手順3でnetwork.confファイルに指定したサーバ識別子名を使用してください。

サンプルファイルのパス

```
/インストールディレクトリ/share/mc_server.conf.sample
```

サーバ識別子.confファイルには、以下のパラメータを設定します。

表2.2 設定するパラメータ

パラメータ	指定内容	備考
db_instance	'データ格納先ディレクトリ'	半角シングルクォートで囲んで、ASCII文字で指定してください。
db_instance_password	'インスタンス管理者ユーザーのユーザー名に対応するパスワード'	Mirroring Controllerからデータベースのインスタンスへの接続設定において、パスワード認証を行う場合に指定してください。 半角シングルクォートで囲んで、ASCII文字で指定してください。 本パラメータの指定値に'または¥を含む場合は、それぞれ¥'および¥¥と記載してください。
enable_hash_in_password	onまたはoff	db_instance_passwordの指定値に含まれている#をパスワードの文字とみなす場合はonを指定し、コメントとみなす場合はoffを指定します。 デフォルトはoffです。
syslog_ident	'プログラム名'	システムログ内のMirroring Controllerメッセージを特定するために使用するプログラム名を半角シングルクォートで囲んで、半角空白以外のASCII文字で指定してください。postgresql.confファイルのパラメータと同じプログラム名にすることで、Mirroring Controllerの出力内容を透過的に参照できるため、ログの参照が容易になります。
remote_call_timeout	管理通信のタイムアウト	Mirroring Controllerのエージェントプロセスがサーバ間で行う通信のタイムアウト値(ミリ秒)を指定します。OSのTCP接続のタイムアウト時間よりも短くなるように指定してください。
heartbeat_error_action	OS/サーバの生死監視においてハートビート異常を検出した場合の動作	arbitration：裁定サーバを利用して自動縮退を行います。 command：縮退の判断を行うユーザーコマンドを呼び出して自動縮退を行います。 message：メッセージを通知します。 fallback：無条件に自動縮退を行います。 プライマリサーバおよびスタンバイサーバで同じ設定にしてください。
heartbeat_interval	OS/サーバの生死監視における異常監視の間隔時間(ミリ秒数)	OS/サーバを監視するため、heartbeat_intervalに指定した間隔(ミリ秒)で異常の監視を行います。

パラメータ	指定内容	備考
heartbeat_timeout	OS/サーバの生死監視における異常監視のタイムアウト時間(秒数)	本パラメータの設定は、データベースプロセスの生死監視、ストリーミングレプリケーションの異常監視およびディスク異常監視のデフォルト値として使用されます。 監視時間に関する設定には、異常監視による縮退を最適化するために考慮することがあります。詳細は、“2.11.4.1 OS/サーバの異常監視のチューニング”を参照してください。
heartbeat_retry	OS/サーバの生死監視における異常監視のリトライ回数(回数)	
fencing_command	'フェンシングコマンドのファイルパス名' [設定例] fencing_command = '/mc/fencing_dir/execute_fencing.sh'	異常と判断したデータベースサーバをフェンシングするフェンシングコマンドのパス名を指定します。 ファイルパス名は絶対パスで指定し、半角シングルクォートで囲んでください。 本パラメータは、heartbeat_error_actionにcommandを設定した場合は、指定必須です。 1024バイト未満の長さで指定してください。
fencing_command_timeout	フェンシングコマンドのタイムアウト時間(秒数)	指定された秒数以上のあいだ、コマンドの応答がない場合は、フェンシングに失敗したと判断し、フェンシングコマンドの実行プロセスに対して、シグナル(SIGTERM)を送信します。 1～2147483647の間の数値で指定してください。 デフォルトは20秒です。
arbitration_timeout	Mirroring Controller裁定プロセスでの裁定処理のタイムアウト時間(秒数)	OS/サーバの生死監視時間 + 裁定定義ファイルのfencing_command_timeout以上の値を指定する必要があります。 指定された秒数以上のあいだ、応答がない場合には、プライマリサーバの切り替えやスタンバイサーバの切り離しは行われません。そのため、手動での縮退操作を行ってください。 1～2147483647の間の数値で指定してください。 本パラメータは、裁定サーバを利用しない運用では設定不要です。
arbitration_command	'裁定コマンドのファイルパス' [設定例] arbitration_command = '/mc/arbitration_dir/execute_arbitration_command.sh'	OS/サーバの生死監視において異常を検出した場合に実行する裁定コマンドのファイルパス名を指定します。 ファイルパス名は絶対パスで指定し、半角シングルクォートで囲んでください。 本パラメータは、heartbeat_error_actionにcommandを設定した場合は、指定必須です。 1024バイト未満の長さで指定してください。
arbitration_command_timeout	裁定コマンドのタイムアウト時間(秒数)	指定された秒数以上のあいだ、裁定コマンドの応答がない場合は、裁定コマンドの実行に失敗したと判断し、裁定コマンドの実行プロセスに対して、シグナル(SIGTERM)を送信します。

パラメータ	指定内容	備考
		1～2147483647の間の数値で指定してください。 本パラメータは、heartbeat_error_actionにcommandを設定した場合のみ、指定可能です。

参考

各パラメータの詳細およびその他のパラメータについては、“A.4.1 データベースサーバのサーバ定義ファイル”を参照してください。

2.4.2 プライマリサーバのインスタンスの作成・設定・登録

プライマリサーバのインスタンスの作成・設定・登録について、説明します。

参照

- pg_hba.confファイルの詳細は、“PostgreSQL Documentation”の“Client Authentication”を参照してください。
- postgresql.confファイルの詳細は、“A.1 プライマリサーバで設定するパラメータ”を参照してください。
- mc_ctlコマンドの詳細は、“リファレンス”を参照してください。

以下の手順で行います。

1. “導入ガイド(サーバ編)”の“セットアップ”を参照して、FUJITSU Enterprise Postgresのセットアップおよびインスタンスの作成を行います。

   データ格納先ディレクトリは、ASCII文字で指定してください。

   注意

   ー データベース多重化運用で異常が発生し、縮退運転になった場合に、スタンバイサーバに対して復旧作業が必要です。スタンバイサーバの復旧にpg_rewindコマンドを使用する場合には、いくつかの前提条件があります。initdbコマンド実行時にチェックサムを有効にすると、その前提条件の1つを満たすことができます(必須ではありません)。前提条件の詳細については“4.1.1.1.3 異常原因の特定とリカバリ”を参照してください。

2. 透過的データの暗号化機能を利用する場合は、格納データの暗号化の設定を行います。

   キーストア・ファイルを作成します。

   鍵管理システムをキーストアとして使用する場合は、鍵管理システムの接続情報を設定し、マスタ暗号化キーを宣言します。

   詳細は、“運用ガイド”の“透過的データ暗号化による格納データの保護”または“鍵管理システムをキーストアとして使用する場合の透過的データ暗号化の運用”を参照して、設定してください。

3. スタンバイサーバからの接続を認証するために、pg_hba.confファイルに以下のエントリーを追加します。本ファイルは、あとでスタンバイサーバに複写します。

```
# TYPE    DATABASE      USER      ADDRESS                    METHOD
  host    replication   fsep      スタンバイサーバのアドレス   認証方式
  host    replication   fsep      プライマリサーバのアドレス   認証方式
```

   プライマリサーバおよびスタンバイサーバのアドレスには、ログ転送用ネットワークに接続するIPアドレスを指定します。また、すべてのサーバがスタンバイサーバにもプライマリサーバにもなることができるので、データベース多重化システムを構成するすべてのサーバのアドレスについて、エントリーを追加してください。

ポイント

認証方式にtrust認証以外の方式を設定する場合

プライマリサーバがスタンバイサーバになった時に、プライマリサーバへの接続で自動認証を行うためにインスタンス管理者ユーザーのホームディレクトリに、.pgpassファイルを作成してreplicationデータベースに対するパスワードを指定します。これにより、インスタンス管理者ユーザーのOSユーザーとデータベースに登録したユーザーが同一となり、不特定のユーザーによる接続ではないことが検証できます。また、事前に設定したパスワードが認証に用いられて、自動的に接続できます。

注意

trust認証に設定すると、プライマリサーバにログイン可能なすべてのOSユーザーが接続できてしまうため、悪意のあるユーザーがいた場合には、誤ったトランザクションログを送信することで、スタンバイサーバのデータを破壊したり、業務システムをクラッシュさせたりしてしまいます。そのため、データベース多重化運用を行うシステムのセキュリティ要件に応じて、認証方式を決定してください。

なお、認証方式の詳細は、“PostgreSQL Documentation”の“Authentication Methods”を参照してください。

4. プライマリサーバのインスタンス管理者ユーザーがデータベースアプリケーションとして接続できるように設定します。

   Mirroring Controllerがインスタンスの異常を監視するために、インスタンス管理者ユーザーのユーザー名でインスタンスに接続できるようにするための設定です。postgresデータベースに接続できるように設定します。

   - パスワード認証を行う場合

     サーバ識別子.confファイルのdb_instance_passwordパラメータにインスタンス管理者ユーザーのユーザー名に対応するパスワードを指定してください。このパスワードを使用して、データベースのインスタンスへの接続を行います。db_instance_passwordパラメータにパスワードを指定しない場合、Mirroring Controllerからデータベースのインスタンスへの接続が失敗して、インスタンスのプロセス監視を正常に行えません。

   - パスワード認証を行わない場合

     db_instance_passwordパラメータへの指定は不要です。

     db_instance_passwordパラメータにインスタンス管理者ユーザーのユーザー名に対応するパスワードを指定しても、指定したパスワードは無視されます。

   - SSLによる証明書認証を行う場合

     サーバ識別子.confファイルのdb_instance_ext_pq_conninfoパラメータおよびdb_instance_ext_jdbc_conninfoパラメータにSSLに関する接続パラメータを指定してください。パラメータを指定しない場合、Mirroring Controllerからデータベースのインスタンスへの接続が失敗して、インスタンスのプロセス監視を正常に行えません。SSLによる証明書認証を行わない場合、パラメータの指定は不要です。

   設定例を以下に示します。

```
# TYPE    DATABASE        USER        ADDRESS         METHOD
  host    postgres        fsep        127.0.0.1/32    認証方式
```

注意

Mirroring ControllerはPostgreSQL JDBC 4.2ドライバを使用してデータベースのインスタンスへ接続を行います。そのため、認証方式は、JDBCドライバがサポートする方式を指定してください。JDBCドライバがサポートしない認証方式を指定した場合、Mirroring Controllerの起動が失敗します。JDBCドライバがサポートする認証方式については、“PostgreSQL JDBC Driver”のマニュアルを参照してください。

5. データベース多重化運用を行うためにpostgresql.confファイルに以下のパラメータを指定します。

   postgresql.confファイルはスタンバイサーバのインスタンス作成時に複製されます。したがって、スタンバイサーバで必要なパラメータについても設定します。

   postgresql.confファイルを編集したあと、インスタンスを再起動してください。

表2.3 設定するパラメータ

パラメータ	指定内容	備考
wal_level	replicaまたはlogical	ロジカルデコーディングも使用する場合にはlogicalを指定します。
max_wal_senders	2以上	Mirroring Controllerのクラスタシステムを構築する場合は、2を指定します。 また、クラスタシステムに非同期スタンバイサーバを追加で接続する場合には、これらのスタンバイサーバからの同時接続数を加算してください。
synchronous_standby_names	'スタンバイサーバ名'	スタンバイサーバを識別する任意の名前を半角シングルクォートで囲んで指定してください。 Mirroring Controller起動中は、本パラメータを変更しないでください。 Mirroring Controllerが管理できるスタンバイサーバは1つであるため、本パラメータに複数の名前を指定しないでください。
hot_standby	on	
wal_keep_size	WALの保存サイズ(メガバイト)	本パラメータの設定値を越える遅延が発生した場合、プライマリサーバはスタンバイサーバが今後必要とするWALセグメントを削除する可能性があります。 また、保守作業などでスタンバイサーバを停止させる場合、停止時間を考慮してWALセグメントが削除されない値を設定してください。 WALの保存サイズの見積もりの詳細は、“導入ガイド(サーバ編)”の“トランザクションログの容量の見積り”を参照してください。
wal_log_hints	on	スタンバイサーバの復旧時にpg_rewindコマンドを使用する場合には、本パラメータを設定するかまたはinitdbコマンド実行時にチェックサムを有効にしてください。
wal_sender_timeout	タイムアウト時間(ミリ秒数)	プライマリサーバ側でのトランザクションログの転送に異常が発生したと判断する時間を指定します。 データベースプロセスの生死監視時間の値と合わせることで、異常と判断する時間を統一できます。
archive_mode	on	アーカイブログモードを指定します。
archive_command	'インストールディレクトリ/bin/pgx_walcopy.cmd "%p" "バックアップデータ格納先ディレクトリ/archived_wal/%f"'	トランザクションログを保存するコマンドと格納先パス名を指定します。
backup_destination	バックアップデータ格納先ディレクトリ名	バックアップデータ格納先ディレクトリ名を指定します。 指定するディレクトリは、インスタンス管理者ユーザーのみがアクセスできるように権限を設定してください。

パラメータ	指定内容	備考
		他のサーバのバックアップデータを使ってリカバリを正しく行うために、すべてのサーバで同じパスになるように指定してください。
max_connections	インスタンスへのクライアントの同時接続数＋superuser_reserved_connectionsの値	指定値は、クライアントアプリケーションからの接続数とインスタンス管理のための接続数を制限するためにも使用されます。 詳細は、“導入ガイド（サーバ編）”の“initdbコマンドでインスタンスを作成した場合”および“PostgreSQL Documentation”の“Connections and Authentication”を参照してください。
superuser_reserved_connections	mc_ctl statusの同時実行数(注)＋2を加算	データベーススーパーユーザーからの接続のために予約されている接続数を指定します。 Mirroring Controllerプロセスからの接続数を加算します。また、加算した値はmax_connectionsパラメータにも反映してください。
wal_receiver_timeout	タイムアウト時間(ミリ秒数)	スタンバイサーバ側でのトランザクションログの受信に異常が発生したと判断する時間を指定します。 データベースプロセスの生死監視時間の値と合わせることで、異常と判断する時間を統一できます。
restart_after_crash	off	restart_after_crash パラメータにonを指定またはデフォルトの場合、一部のサーバプロセスがダウンした際に、クラッシュリカバリを含めたFUJITSU Enterprise Postgresの再起動に相当する動作が行われます。 しかし、データベース多重化運用による監視の機能で一部のサーバプロセスがダウンした時点で異常を検知してフェイルオーバが発生し、再起動途中のサーバプロセスが強制停止されます。このような意味なく再起動して途中でキャンセルされるという複雑な動作を抑止するためにoffを設定します。
synchronous_commit	onまたはremote_apply	トランザクションのコミット処理がクライアントに正常終了の応答を返す前に、スタンバイサーバへのWAL送信をどこまで行うかを指定します。 切り替えや切り替え直後のOSやサーバのダウンからのデータ損失を防ぐために、onまたはremote_applyを推奨します。
recovery_target_timeline	latest	切り替え発生時に、新スタンバイサーバ(旧プライマリサーバ)が新プライマリサーバに追随するために、latestを指定します。 本パラメータの指定は、プライマリサーバの切り替え後、旧プライマリサーバを新しくスタンバイサーバとして組み込む時に必要です。

注) mc_ctlコマンドのstatusモードでの同時実行数です。

2.4.3 プライマリサーバのMirroring Controllerの起動

プライマリサーバのMirroring Controllerの起動について説明します。

裁定サーバを利用して自動縮退を行う運用では、事前に裁定サーバでMirroring Controller裁定プロセスを起動してください。

1. Mirroring Controllerのプロセスを起動します。

自動切り替え/切り離しを有効化する場合

インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで実行します。

例)

```
$ mc_ctl start -M /mcdir/inst1
```

自動切り替え/切り離しを有効化しない場合

インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで-Fオプションを指定して実行します。

例)

```
$ mc_ctl start -M /mcdir/inst1 -F
```

注意

- 裁定サーバを利用して自動縮退を行う運用の場合、データベースサーバが裁定サーバに接続する時間が加算されるため、裁定サーバを利用しない運用に比べて、Mirroring Controllerの起動時間が長くかかる場合があります。
- 裁定サーバで設定したOS/サーバの生死監視のパラメータが、Mirroring ControllerのOS/サーバの生死監視のパラメータより大きい値の場合、Mirroring Controllerの起動に失敗することがあります。その場合、メッセージ通知の内容を確認し、裁定サーバまたはMirroring ControllerのOS/サーバの生死監視のパラメータを見直してください。
- 自動切り替え/切り離しを有効化してMirroring Controllerを起動した場合であっても、サーバ識別子.confファイルのheartbeat_error_actionの設定がmessageの場合、OS/サーバの生死監視での異常検出時は、切り替え/切り離しは自動で動作せず、メッセージ通知のみ行われます。
- セットアップ直後に、スタンバイサーバを誤ってプライマリサーバとして起動した場合や、切り替えが行われた後の旧プライマリサーバを復旧せずに、誤ってプライマリサーバとして起動した場合、通常はMirroring Controllerの起動が失敗しますが、万が一、管理用ネットワークが切断されていると起動を失敗させることができず、両サーバがプライマリサーバとなってしまう可能性があります。そのため、Mirroring Controllerの起動は、管理用ネットワークが接続していることを確認してから行ってください。

ポイント

- 裁定サーバを利用して自動縮退を行う運用では、事前に裁定サーバでMirroring Controller裁定プロセスを起動していない場合、mc_ctlコマンドが失敗します。ただし、事前にMirroring Controller裁定プロセスを起動できない場合であっても、mc_ctlコマンドに--async-connect-arbiterオプションを指定することで、Mirroring Controllerプロセスを起動することができます。
- 自動切り替え/切り離しはMirroring Controllerが起動したあとに、mc_ctlコマンドのenable-failoverモード、または、disable-failoverモードを使用することにより自動切り替え/切り離しを有効、または無効にすることが可能です。

2. pgx_dmpallコマンドを使用して、バックアップを取得します。

2.5 スタンバイサーバのセットアップ

スタンバイサーバのセットアップについて、説明します。

2.5.1 スタンバイサーバのデータベース多重化運用のセットアップ

スタンバイサーバのデータベース多重化運用のセットアップについて、説明します。

データベース多重化機能では、動作に必要なファイルをMirroring Controller管理ディレクトリで管理します。

1つのインスタンスに対して1つのMirroring Controller管理ディレクトリが対応します。

注意

- Mirroring Controller管理ディレクトリは、FUJITSU Enterprise Postgresが管理するディレクトリとともに誤って削除されたり、FUJITSU Enterprise Postgresのリカバリとともに古いファイルに戻されたりすることがないようにFUJITSU Enterprise Postgresが管理するディレクトリ配下に配置しないでください。
- 大きなデータベースのスタンバイサーバを作成するときには、業務システムの運用を停止するか、wal_keep_sizeパラメータに大きい値を指定するか、またはレプリケーションスロットを使用してください。

  pg_basebackupコマンドでスタンバイサーバを構築してから起動するまでに発生したWALはプライマリサーバで保持される必要がありますが、保持できるWALセグメントの数がwal_keep_sizeパラメータで制約されるためです。

  また、wal_keep_sizeパラメータの設定には、データベース多重化運用を安定化させるために考慮することがあります。詳細は、“2.11.1 データベース多重化運用を安定させるためのチューニング”を参照してください。

参照

- FUJITSU Enterprise Postgresが管理するディレクトリの詳細は、“導入ガイド(サーバ編)”の“セットアップ”の“資源配置用のディレクトリの準備”を参照してください。
- pg_basebackupコマンドの詳細については、“PostgreSQL Documentation”の“Reference”の“pg_basebackup”を参照してください。
- mc_ctlコマンドの詳細は、“リファレンス”を参照してください。
- セットアップで編集する各パラメータの詳細については、“付録A パラメータ”を参照してください。
- レプリケーションスロットの詳細は“PostgreSQL Documentation”の“Replication Slots”を参照してください。

以下の手順で行います。

1. スタンバイサーバにログインします。
2. データベース多重化機能で必要なファイルを格納するMirroring Controller管理ディレクトリを作成します。

   Mirroring Controller管理ディレクトリは、ASCII文字で指定してください。

   また、インスタンス管理者ユーザーに対する書き込み権限を付与してください。
3. プライマリサーバのnetwork.confファイルをコピーして配置します。

   プライマリサーバのセットアップで定義したnetwork.confファイルをコピーして、スタンバイサーバのMirroring Controller管理ディレクトリ配下に配置してください。

   インスタンス管理者ユーザーにのみ読み込み権限と書き込み権限を設定してください。インスタンス管理者ユーザー以外にアクセス権限が与えられていた場合には、mc_ctlコマンドは動作しません。これにより、インスタンス管理者ユーザー以外がMirroring Controllerを操作することを防止します。

   WebAdminのように/etc/servicesファイルを用いて空きポート番号を検索して使用するプログラムが存在するため、network.confファイルに指定したスタンバイサーバ側のポート番号を/etc/servicesファイルに登録してください。

   なお、サービス名は、任意の名前で登録してください。
4. プライマリサーバのサーバ識別子.confファイルをコピーして配置します。

   プライマリサーバのセットアップで定義したサーバ識別子.confファイルをコピーして、スタンバイサーバのMirroring Controller管理ディレクトリ配下に配置します。

インスタンス管理者ユーザーにのみ読み込み権限と書き込み権限を設定してください。インスタンス管理者ユーザー以外にアクセス権限が与えられていた場合には、mc_ctlコマンドは動作しません。

2.5.2 スタンバイサーバのインスタンスの作成・設定・登録

スタンバイサーバのインスタンスの作成・設定・登録について、説明します。

参照

- 各パラメータの詳細については、“付録A パラメータ”を参照してください。
- mc_ctlコマンドの詳細は、“リファレンス”を参照してください。

以下の手順で行います。

1. カーネルパラメータの設定を行います。

   詳細は“導入ガイド(サーバ編)”の“カーネルパラメータの設定”を参照してください。

2. 透過的データの暗号化機能を利用する場合は、格納データの暗号化の設定を行います。

   詳細は、“運用ガイド”の“透過的データ暗号化による格納データの保護”または“鍵管理システムをキーストアとして使用する場合の透過的データ暗号化の運用”を参照して、設定してください。

3. pg_basebackupコマンドを実行して、スタンバイサーバにプライマリサーバのインスタンスの複製を作成します。

   例)

```
$ pg_basebackup -D /database/inst1 -X fetch --waldir=/transaction/inst1 --progress --verbose -R --
dbname='application_name=スタンバイサーバ名' -h プライマリサーバのIPアドレス -p プライマリサーバのポート番号
```

注意

- pg_basebackupコマンドには、-Rオプションを指定して実行し、standby.signalファイルを作成してください。standby.signalファイルを作成しない場合、Mirroring Controllerはスタンバイサーバとして起動できません。
- プライマリサーバへの接続がパスワード認証を必要とする方式の場合、自動で認証が行われるようにしておく必要があります。pg_basebackupコマンドの-Rオプションを指定し、--dbnameオプションにpasswordパラメータを指定すると、pg_basebackupコマンドによりpostgresql.auto.confファイルのprimary_conninfoパラメータにパスワードが設定されて自動的に接続できるようになります。

  postgresql.auto.confファイルのprimary_conninfoパラメータにパスワードを設定しない場合は、インスタンス管理者ユーザーのホームディレクトリに.pgpassファイルを作成してreplicationデータベースに対するパスワードを設定してください。
- primary_conninfoパラメータは、postgresql.confファイルには設定せず、pg_basebackupコマンドを使用してpostgresql.auto.confファイルのみに設定してください。
- pg_basebackupコマンドの実行中に行われるトランザクションログファイルの収集において、考慮が必要な場合があります。
  - -Xオプションにfetchを指定してコマンドを実行する場合

    トランザクションログをバックアップの最後に収集する動作を指定して本コマンドを実行する場合は、バックアップ中に発生したトランザクションログがプライマリサーバで削除されないようにすることが必要です。そのため、postgresql.confファイルのwal_keep_sizeパラメータには十分な値を考慮して設定してください。
  - -Xオプションの省略や-Xオプションにstreamを指定してコマンドを実行する場合

    トランザクションログをストリームする動作を指定して本コマンドを実行すると、プライマリサーバでMirroring Controllerが起動している場合には、本コマンドによるストリーミングレプリケーションの接続を検知した契機で、接続を同期スタンバイサーバに変更してしまいます。そのため、プライマリサーバで業務を開始している場合にはそれらに影響を与えるため、プライマリサーバのMirroring Controllerプロセスのみを停止してから本コマンドを実行してください。

参照

standby.signalファイルの詳細については、“PostgreSQL Documentation”の“Hot Standby”を参照してください。

4. postgresql.confファイルに以下のパラメータを設定します。

表2.4 設定するパラメータ

パラメータ	指定内容	備考
synchronous_standby_names	'プライマリサーバ名'	プライマリサーバを切り替えたあとに、旧プライマリサーバを新しくスタンバイサーバとして組み込んだときに必要です。 Mirroring Controller起動中は、本パラメータを変更しないでください。 Mirroring Controllerが管理できるスタンバイサーバは1つであるため、本パラメータに複数の名前を指定しないでください。
streaming_wal_compression	-1から9までの数値	ストリーミングレプリケーションで、WALを圧縮して送信したい場合に指定します。 -1: zlibのデフォルトの圧縮レベルで圧縮します 0: 無圧縮 1-9:指定された圧縮レベルで圧縮します。9が高圧縮となります 本パラメータを指定しない場合は、0を指定したものとみなされます。

参照

streaming_wal_compressionについては、“運用ガイド”の“ストリーミングレプリケーションのWAL圧縮機能”を参照してください。.

2.5.3 スタンバイサーバのMirroring Controllerの起動

スタンバイサーバのMirroring Controllerの起動について説明します。

裁定サーバを利用して自動縮退を行う運用では、事前に裁定サーバでMirroring Controller裁定プロセスを起動してください。

1. スタンバイサーバのMirroring Controllerは、プライマリサーバのMirroring Controllerのプロセスが起動していることを確認してから、起動します。

自動切り替え/切り離しを有効化する場合

インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで実行します。

例）

```
$ mc_ctl start -M /mcdir/inst1
```

自動切り替え/切り離しを有効化しない場合

インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで-Fオプションを指定して実行します。

例）

```
$ mc_ctl start -M /mcdir/inst1 -F
```

2. Mirroring Controllerプロセスの状態を確認します。

   インスタンス管理者ユーザーでmc_ctlコマンドをstatusモードで実行します。“mirroring status”が切り替え可能状態(switchable)になっていることを確認してください。

   例）

```
$ mc_ctl status -M /mcdir/inst1
```

注意

- 裁定サーバを利用して自動縮退を行う運用の場合、データベースサーバが裁定サーバに接続する時間が加算されるため、利用しない運用に比べて、Mirroring Controllerの起動時間が長くかかる場合があります。
- 裁定サーバで設定したOS/サーバの生死監視のパラメータが、Mirroring ControllerのOS/サーバの生死監視のパラメータより大きい値の場合、Mirroring Controllerの起動に失敗することがあります。その場合、メッセージ通知の内容を確認し、裁定サーバまたはMirroring ControllerのOS/サーバの生死監視のパラメータを見直してください。
- 自動切り替え/切り離しを有効化してMirroring Controllerを起動した場合であっても、サーバ識別子.confファイルのheartbeat_error_actionの設定がmessageの場合、OS/サーバの生死監視での異常検出時は、切り替え/切り離しは自動で動作せず、メッセージ通知のみ行われます。
- セットアップ直後に、スタンバイサーバを誤ってプライマリサーバとして起動した場合や、切り替えが行われた後の旧プライマリサーバを復旧せずに、誤ってプライマリサーバとして起動した場合、通常はMirroring Controllerの起動が失敗しますが、万が一、管理用ネットワークが切断されていると起動を失敗させることができず、両サーバがプライマリサーバとなってしまう可能性があります。そのため、Mirroring Controllerの起動は、管理用ネットワークが接続していることを確認してから行ってください。

ポイント

- 裁定サーバを利用して自動縮退を行う運用では、事前に裁定サーバでMirroring Controller裁定プロセスを起動していない場合、mc_ctlコマンドが失敗します。ただし、事前にMirroring Controller裁定プロセスを起動できない場合であっても、mc_ctlコマンドに--async-connect-arbiterオプションを指定することで、Mirroring Controllerプロセスを起動することができます。
- 自動切り替え/切り離しはMirroring Controllerが起動したあとに、mc_ctlコマンドのenable-failoverモード、または、disable-failoverモードを使用することにより自動切り替え/切り離しを有効、または無効にすることが可能です。

2.6 データベースサーバのユーザーコマンドの作成

データベースサーバのユーザーコマンドの作成について説明します。

以下のユーザーコマンドは、Mirroring Controller管理プロセスから呼び出されます。

これらは必要に応じて利用者が作成します。

作成したユーザーコマンドはデータベースサーバのサーバ定義ファイル内のパラメータに指定します。パラメータの詳細については、“A.4.1 データベースサーバのサーバ定義ファイル“を参照してください。

ユーザーコマンドの種類

- フェンシングコマンド

  Mirroring Controllerが裁定処理を行った結果、不安定なデータベースサーバと判断した場合に、フェンシングを行うためのユーザーコマンドです。

- 裁定コマンド

  裁定サーバを設置しない場合に、裁定サーバの代わりに裁定処理を行うユーザーコマンドです。

- 状態遷移コマンド

  Mirroring Controllerがデータベースサーバの状態遷移を行った際に呼び出されるユーザーコマンドです。
  本コマンドには以下があります。

－ 切り替え後コマンド

スタンバイサーバがプライマリサーバに昇格した後に呼び出されるユーザーコマンドです。

－ 切り離し前コマンド

スタンバイサーバがクラスタシステムから切り離される前に呼び出されるユーザーコマンドです。
切り離し前コマンドがプライマリサーバ、スタンバイサーバの両方に指定されている場合、スタンバイサーバ側、プライマリサーバ側の順に呼び出されます。

スタンバイサーバを切り離す際にスタンバイサーバのインスタンスを強制停止させる設定を行っている場合、スタンバイ側の切り離し前コマンドの呼び出し後に、スタンバイサーバのインスタンスの停止が行われます。

－ 組み込み後コマンド

スタンバイサーバがクラスタシステムに組み込まれた後に呼び出されるユーザーコマンドです。
組み込み後コマンドがプライマリサーバ、スタンバイサーバの両方に指定されている場合、プライマリサーバ側、スタンバイサーバ側の順に呼び出されます。

ポイント

裁定サーバを利用して自動縮退を行う運用では、裁定サーバ側のみのフェンシングコマンドで要件を満たすことが可能な場合、データベースサーバ側でのフェンシングコマンドは不要です。また、データベースサーバ側のみのフェンシングコマンドで要件を満たすことが可能な場合には、裁定サーバ側では、実装を伴わない終了処理だけのフェンシングコマンドを作成してください。

表2.5 ユーザーコマンドの利用可否と呼び出し元のデータベースサーバ

ユーザーコマンド	OS/サーバの生死監視においてハートビート異常を検出した場合の動作				呼び出し元のデータベースサーバ	
	メッセージ出力	裁定サーバによる自動縮退	裁定コマンドによる自動縮退	無条件に自動縮退	プライマリサーバ	スタンバイサーバ
フェンシングコマンド	○(注1)	○(注2)	◎	×	○	○
裁定コマンド	×	×	◎	×	○	○
切り替え後コマンド	○	○	○	○	○	×
切り離し前コマンド	○	○	○	○	○	○(注3)
組み込み後コマンド	○	○	○	○	○	○(注3)

◎:必須

○:利用可能

×:利用不可

注1) mc_ctlコマンドで強制切り替え、または強制切り離しを実行した場合にのみ呼び出されます。

注2) データベースサーバ側でのフェンシングコマンドの作成は任意ですが、裁定サーバ側での作成は必須です。

注3) メッセージ出力または無条件に自動縮退する動作を選択している場合には、プライマリサーバからのみ呼び出されます。

参照

各ユーザーコマンドについては、“付録C ユーザーコマンド”を参照してください。

2.7 ストリーミングレプリケーションの状態確認

データベース多重化運用のセットアップを行う前に、前提となるストリーミングレプリケーション機能が正しくセットアップされていることを確認します。

以下の手順で行います。

1. プライマリサーバにおいて、統計情報ビューpg_stat_replicationにより、1行検索できることを確認します。

   例）
   psqlコマンドを使用した場合の出力例を以下に示します。

```
postgres=# select * from pg_stat_replication;
-[ RECORD 1 ]----+------------------------------
pid              | 10651
usesysid         | 10
usename          | fsep
application_name | standby
client_addr      | 192.0.2.210
client_hostname  |
client_port      | 55098
backend_start    | 2022-03-23 11:17:49.628793+09
backend_xmin     |
state            | streaming
sent_lsn         | 0/3000060
write_lsn        | 0/3000060
flush_lsn        | 0/3000060
replay_lsn       | 0/3000060
write_lag        |
flush_lag        |
replay_lag       |
sync_priority    | 1
sync_state       | sync
reply_time       | 2022-03-23 11:23:27.703366+09
```

2. 手順1の検索結果を確認します。

   意図したスタンバイサーバと同期モードでの接続が確立されていることを確認します。

表2.6 確認する項目

項目	確認内容
application_name	プライマリサーバのpostgresql.confファイルに指定するsynchronous_standby_namesパラメータの値と同じであること。
client_addr	スタンバイサーバのIPアドレスであること。
state	“streaming”であること。
sync_state	“sync”であること。

参照

統計情報ビューpg_stat_replicationの詳細は、“PostgreSQL Documentation”の“Server Administration”の“The Statistics Collector”を参照してください。なお、統計情報ビューpg_stat_replicationが将来変更される可能性がある点に注意してください。

2.8 接続状態の確認

データベースサーバまたは裁定サーバから接続状態を確認する方法を説明します。

2.8.1 データベースサーバでの接続状態の確認

Mirroring Controller裁定プロセス、プライマリサーバおよびスタンバイサーバのMirroring Controllerプロセスとの接続状態を、データベースサーバで確認する方法について説明します。

以下の手順で行います。

1. プライマリサーバとスタンバイサーバで、mc_ctlコマンドのstatusモードを--arbiterオプションを指定して実行します。

   例）
   mc_ctlコマンドに--arbiterオプション指定して実行した場合の状態表示例を以下に示します。

```
$ mc_ctl status --arbiter -M /mcdir/inst1

arbiter_id   host           status
----------------------------------------
arbiter      192.0.3.120    online
```

2. プライマリサーバとスタンバイサーバで、手順1で実行したmc_ctlコマンドのstatusモードの表示結果を確認します。

   確認する項目

   “status”の項目が、“online”になっていることを確認してください。

参照

mc_ctlコマンドの詳細は、“リファレンス”を参照してください。

2.8.2 裁定サーバでの接続状態の確認

Mirroring Controller裁定プロセス、プライマリサーバおよびスタンバイサーバのMirroring Controllerプロセスとの接続状態を、裁定サーバで確認する方法について説明します。

以下の手順で行います。

1. 裁定サーバでmc_arbコマンドをstatusモードで実行します。

   mc_arbコマンドを実行した場合の状態表示例を以下に示します。

   例）

```
$ mc_arb status -M /mcarb_dir/arbiter1

server_id    host           status
----------------------------------------
server1      192.0.3.100    online
server2      192.0.3.110    online
```

2. 手順1で実行したmc_arbコマンドの表示結果を確認します。

   確認する項目

   “status”の項目が、2行とも“online”になっていることを確認してください。

参照

mc_arbコマンドの詳細は、“リファレンス”を参照してください。

2.9 アプリケーションの作成

データベース多重化機能を使用したアプリケーションの作成方法および作成時の注意事項について、説明します。

2.9.1 アプリケーションの接続先サーバの設定

データベース多重化機能を使用した場合は、フェイルオーバが発生するとアプリケーションの接続先のサーバを切り替える必要があります。したがって、アプリケーションを作成するには、アプリケーションの接続先切り替え機能を使用して作成します。

参照

アプリケーションの接続先切り替え機能の詳細は、“アプリケーション開発ガイド”の“アプリケーションの接続先切り替え機能”を参照してください。

2.10 動作確認

環境設定を正しく実施できているかを確認するために、アプリケーションを起動して切り替えや組込みの動作確認を行ってください。

2.11 チューニング

データベース多重化運用におけるチューニング方法について、説明します。

2.11.1 データベース多重化運用を安定させるためのチューニング

データ更新を大量に行う場合、データベースへの書き込み負荷が大きくなるため、多重化の状態が不安定になることがあります。

したがって、postgresql.confファイルの以下のパラメータを編集することで多重化の状態を安定に維持できます。また、これらのパラメータを編集するには、“導入ガイド(サーバ編)”の“トランザクションログの容量の見積り”を参照してください。

表2.7 パラメータ

パラメータ	内容
wal_keep_size	詳細は、“2.4.2 プライマリサーバのインスタンスの作成・設定・登録”を参照してください。
max_wal_size	トランザクションログは、チェックポイントの契機で書き出されます。 本パラメータに指定した値の容量のトランザクションログが発生すると、チェックポイントが実行されます。 本パラメータに大きい値を指定した場合、クラッシュリカバリで必要となる時間が増加します。 本パラメータに小さい値を指定した場合、チェックポイントが多発するためプライマリサーバに接続するアプリケーションの性能に影響します。

2.11.2 スタンバイサーバでの問い合わせを安定させるためのチューニング

スタンバイサーバの参照系の業務で行われる問い合わせは、プライマリサーバで行われる業務によって取り消されることがあります。

取り消される可能性を少なくするために、postgresql.confファイルのmax_standby_archive_delayパラメータにできるだけ大きい値を指定してください。

参照

- 詳細は、“PostgreSQL Documentation”の“Handling Query Conflicts”を参照してください。
- max_standby_archive_delayパラメータの詳細は“PostgreSQL Documentation”の“Standby Servers”を参照してください。

2.11.3 スタンバイサーバでの問い合わせを安定させるためのチューニング(プライマリサーバで頻繁に更新する場合)

プライマリサーバで規則的に、かつ頻繁に更新する業務を行う場合、スタンバイサーバの参照系の業務で行われる問い合わせが取り消されやすくなります。この場合には、postgresql.confファイルの以下のどちらかのパラメータを編集してください。

表2.8 パラメータ

パラメータ	内容
hot_standby_feedback	onに設定することで、プライマリサーバで削除・更新したデータ領域の削除(vacuum)を抑止します。

パラメータ	内容
	これにより、スタンバイサーバでの問い合わせが取り消されなくなります。(注)
vacuum_defer_cleanup_age	指定した数のトランザクションが処理されるまでのあいだ、プライマリサーバで削除や更新したデータ領域の削除(vacuum)を遅らせます。これにより、スタンバイサーバでの問い合わせが取り消される確率を低減します。

注)vacuumが遅れるため、プライマリサーバのデータ格納先ディスクの容量が圧迫されます。
また、スタンバイサーバで実行される問い合わせとアクセスが競合すると、競合したことを示す内容のトランザクションログが転送されます。
したがって、アクセスの競合が発生しないようにmax_standby_archive_delayパラメータにできるだけ大きい値を指定してください。

参照

- hot_standby_feedbackパラメータの詳細は“PostgreSQL Documentation”の“Standby Servers”を参照してください。
- vacuum_defer_cleanup_ageパラメータの詳細は、“PostgreSQL Documentation”の“Primary Server”を参照してください。

2.11.4 最適な異常監視による縮退運転を行うためのチューニング

Mirroring Controllerには監視対象の資源へのアクセスにおいて、タイムアウト時間の超過やリトライ回数を越えたことで異常と判断する監視方法があります。これらの設定が適切でないと、誤検知のリスクや自動縮退の遅延を引き起こす場合があるため、適切な設計が必要となります。

例えば、異常監視に関するチューニングが適切に行われていない場合、以下のような事象が発生します。

- タイムアウト時間が短い場合

  冗長な縮退を引き起こして可用性が低下します。

- タイムアウト時間が長い場合

  業務継続に影響を与える障害が発生しても、自動縮退までの時間が長期化して業務停止を引き起こす可能性があります。

以降で説明するサーバ定義ファイルのパラメータをシステムに応じた値に編集することで縮退運転の最適化を行うことができます。また、これらのパラメータを編集するには、“A.4 サーバ定義ファイル”を参照してください。

2.11.4.1 OS/サーバの異常監視のチューニング

OS/サーバの異常監視のチューニングは、OS/サーバの生死監視においてハートビート異常を検出した場合の動作により異なります。

参照

OS/サーバの生死監視においてハートビート異常を検出した場合の動作については、“1.1.1 データベース多重化運用による監視”を参照してください。

2.11.4.1.1 裁定サーバを利用して自動縮退を行う運用の異常監視のチューニング

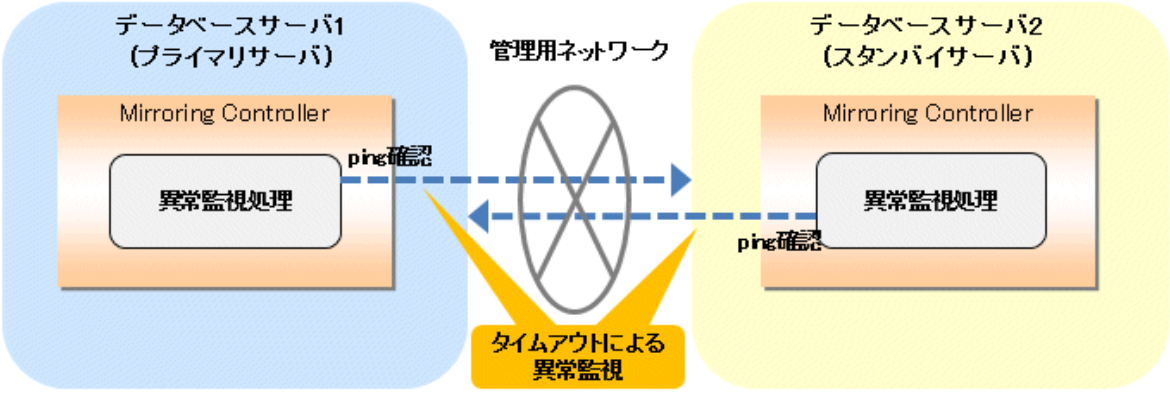
裁定サーバを利用して自動縮退を行う運用では、Mirroring Controller裁定プロセスがMirroring Controllerからの裁定依頼に即時に応答できるように、データベースサーバの異常監視を定期的に行っています。裁定サーバを利用した自動縮退は、以下のパラメータを編集することによって、OS/サーバの異常検知から自動縮退までの時間の最適化を行うことができます。

- データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータ
- 裁定定義ファイルのOS/サーバの異常監視のパラメータ
- 裁定処理とフェンシングに関するパラメータ

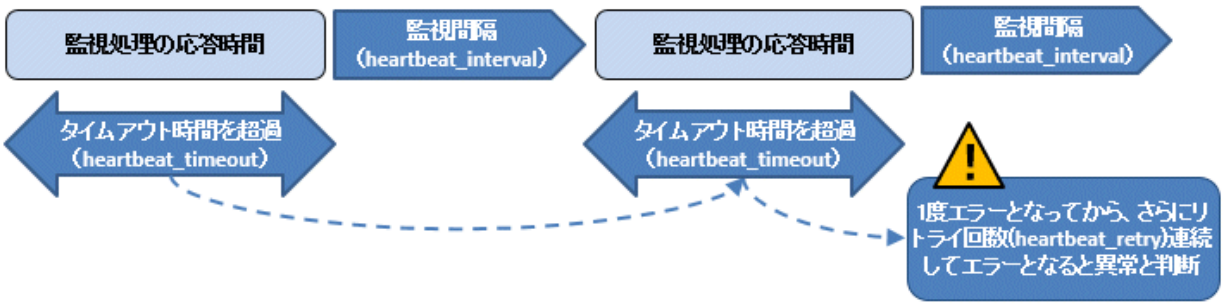
データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータ

表2.9 データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータ

パラメータ	指針
異常監視の監視間隔 (heartbeat_interval)	Mirroring Controllerによる異常監視がシステムに負荷を与えない範囲で設定しますが、通常は設定不要です (デフォルトは800ミリ秒)。
異常監視のタイムアウト時間 (heartbeat_timeout)	サーバや管理用ネットワークの能力に対して負荷が継続的に掛かる時間を考慮します。例えば、高負荷のバッチ業務を実行する場合や高多重のオンライン業務が連続的に発生するような場合が想定されます (デフォルトは1秒)。
異常監視のリトライ回数 (heartbeat_retry)	負荷が流動的なシステムなど、heartbeat_timeoutで指定した値を超える事象に対して安全値が必要な場合に設定しますが、通常は設定不要です (デフォルトは2回)。



タイムアウトによる異常監視の流れ



Mirroring Controllerが異常と判断するまでの時間の計算方法を以下に示します。

```
Mirroring Controllerの異常検出時間 ＝ ( heartbeat_timeout(秒数) + heartbeat_interval(ミリ秒) / 1000 ) ×
( heartbeat_retry(回数) + 1)
```

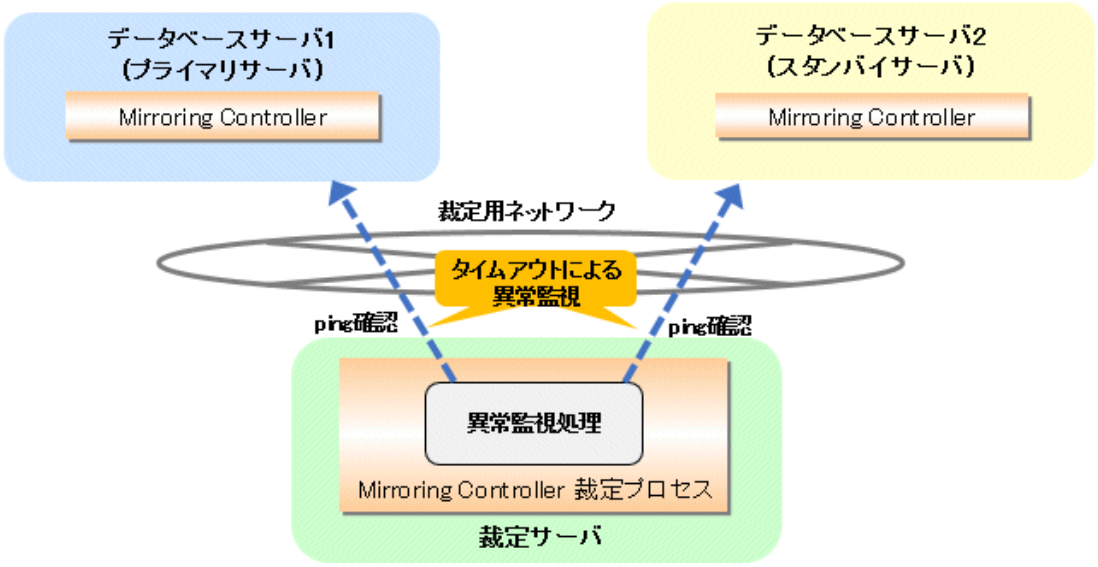
デフォルトの値を使用した場合の異常検出時間は以下となります。

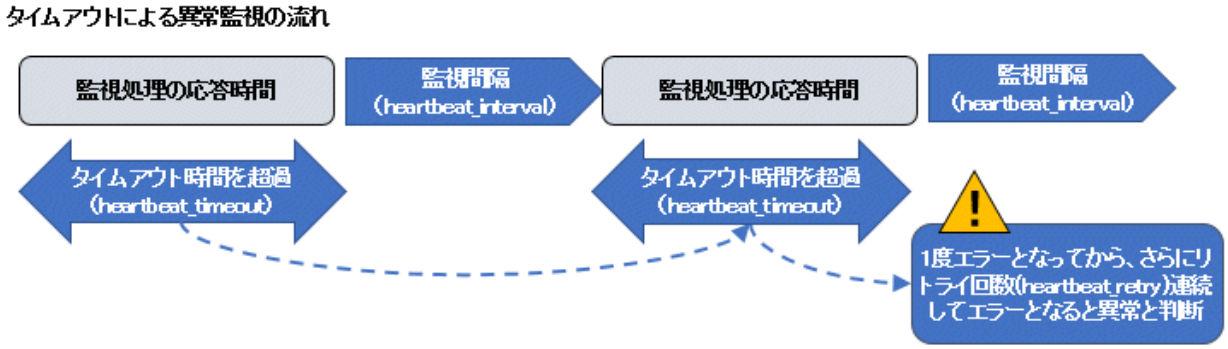
```
Mirroring Controllerの異常検出時間 ＝ ( 1 + 800 / 1000 ) × ( 2 + 1 )
                                   ＝ 5.4(秒)
```

裁定定義ファイルのOS/サーバの異常監視のパラメータ

表2.10 裁定定義ファイルのOS/サーバの異常監視のパラメータ

パラメータ	指針
異常監視の監視間隔 (heartbeat_interval)	Mirroring Controller裁定プロセスによる異常監視がシステムに負荷を与えない範囲で設定しますが、通常は設定不要です(デフォルトはデータベースサーバのサーバ定義ファイルのheartbeat_intervalの設定値です) (ミリ秒数)。
異常監視のタイムアウト時間 (heartbeat_timeout)	サーバや裁定用ネットワークの能力に対して負荷が継続的に掛かる時間を考慮します(デフォルトはデータベースサーバのサーバ定義ファイルのheartbeat_timeoutの設定値です) (秒数)。
異常監視のリトライ回数 (heartbeat_retry)	負荷が流動的なシステムなど、heartbeat_timeoutで指定した値を超える事象に対して安全値が必要な場合に設定しますが、通常は設定不要です(デフォルトはデータベースサーバのサーバ定義ファイルのheartbeat_retryの設定値です) (回数)。





Mirroring Controller裁定プロセスが異常と判断するまでの時間の計算方法を以下に示します。

```
Mirroring Controller裁定プロセスの異常検出時間 ＝ ( heartbeat_timeout(秒数) + heartbeat_interval(ミリ秒) / 1000 ) ×
( heartbeat_retry(回数) + 1)
```

デフォルトの値を使用した場合の異常検出時間は以下となります。

```
Mirroring Controller裁定プロセスの異常検出時間 ＝ ( 1 + 800 / 1000 ) × ( 2 + 1 )
                                ＝ 5.4(秒)
```

ポイント

裁定サーバを利用して自動縮退を行う運用の異常検出時間は以下のように計算できます。

```
異常検出時間 ＝ Max( Mirroring Controllerの異常検出時間, Mirroring Controller裁定プロセスの異常検出時間)
```

注意

裁定定義ファイルにheartbeat_intervalを設定する場合、データベースサーバのサーバ定義ファイルで指定したOS/サーバの異常監視のパラメータとの関係が以下の関係式を満たすように設計してください。

```
( 裁定定義ファイルのheartbeat_interval(ミリ秒) / 1000 ) <
( heartbeat_timeout(秒数) + heartbeat_interval(ミリ秒) / 1000 ) × heartbeat_retry(回数) + heartbeat_timeout(秒数)
```

裁定処理とフェンシングに関するパラメータ

表2.11 裁定処理とフェンシングに関するパラメータ

パラメータ	指針
裁定処理のタイムアウト時間 (データベースサーバのサーバ定義ファイルのarbitration_timeout)	Mirroring Controller裁定プロセスでの裁定処理に掛かる時間を考慮します。Mirroring Controller裁定プロセスの異常検出時間＋裁定定義ファイルのfencing_command_timeout以上の値を指定する必要があります (秒数)。
フェンシングのタイムアウト時間 (裁定定義ファイルのfencing_command_timeout)	フェンシングコマンドの実行に掛かる時間を考慮します (秒数)。

OS/サーバの異常検知から自動縮退までの流れ

裁定サーバを利用して自動縮退を行う運用のOS/サーバの異常検知から自動縮退が発生するまでの流れと関連するパラメータを示します。

<table>
<tr><th>OS/サーバの異常検知から自動縮退までの流れ</th><th>説明</th><th colspan="2">関連するパラメータ</th></tr>
<tr><td>(1) 異常検知</td><td>Mirroring Controllerがデータベースサーバの OS/サーバの異常を検知します。</td><td colspan="2">データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータ</td></tr>
<tr><td>(2) 裁定依頼</td><td>OS/サーバの異常を検知したサーバのMirroring Controllerが裁定サーバに対して相手サーバのOS/サーバの状態確認を依頼します。</td><td>-</td><td rowspan="3">データベースサーバのサーバ定義ファイルのarbitration_timeout</td></tr>
<tr><td>(3) 裁定処理</td><td>Mirroring Controller裁定プロセスが、相手サーバのOS/サーバの状態を確認します。ただし、データベースサーバのMirroring Controllerからの裁定依頼の前に、裁定サーバによるOS/サーバの異常監視の結果が確定している場合は、本処理は実施しません。</td><td>裁定定義ファイルのOS/サーバの異常監視のパラメータ</td></tr>
<tr><td>(4) フェンシング</td><td>Mirroring Controller裁定プロセスが相手サーバをOS/サーバの異常と判断した場合、その相手サーバをフェンシングし、クラスタシステムから隔離します。Mirroring Controller裁定プロセスがOS/サーバの状態を正常と判断した場合は、本処理および(6)は実施しません。</td><td>裁定定義ファイルのfencing_command_timeout</td></tr>
<tr><td>(5) 裁定結果の返却</td><td>裁定依頼を行ったデータベースサーバのMirroring Controllerに対して、裁定処理の結果を返却します。</td><td colspan="2">-</td></tr>
<tr><td>(6) 自動縮退</td><td>自動縮退を行います。(4)でフェンシングが失敗した場合は、本処理は実施しません。</td><td colspan="2">-</td></tr>
</table>

-:関連パラメータなし

注意

データベースサーバのサーバ定義ファイルにfencing_commandパラメータを指定している場合、裁定サーバでのフェンシングが正しく行われたときに、データベースサーバでフェンシングコマンドが呼び出されます。その場合、データベースサーバのサーバ定義ファイルのfencing_command_timeoutパラメータの値を見積もりに加算してください。

図2.1 プライマリサーバのMirroring ControllerでOS/サーバの異常を検知した場合

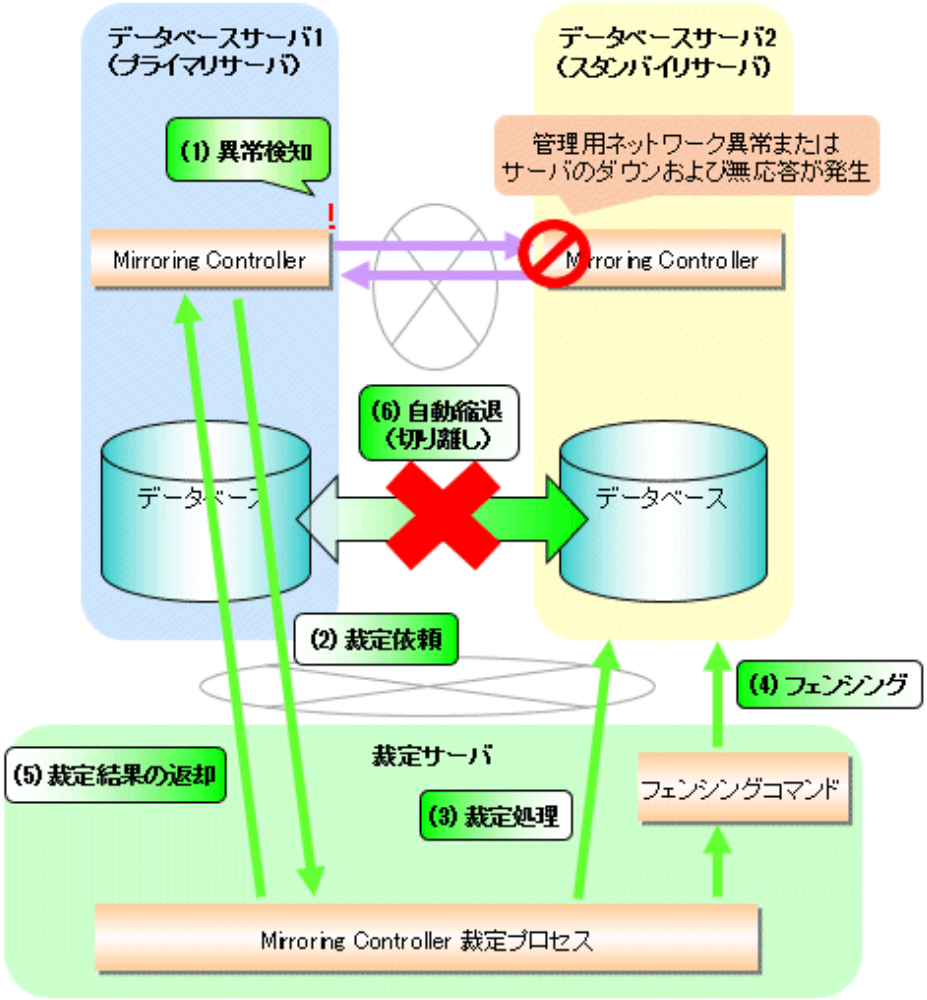
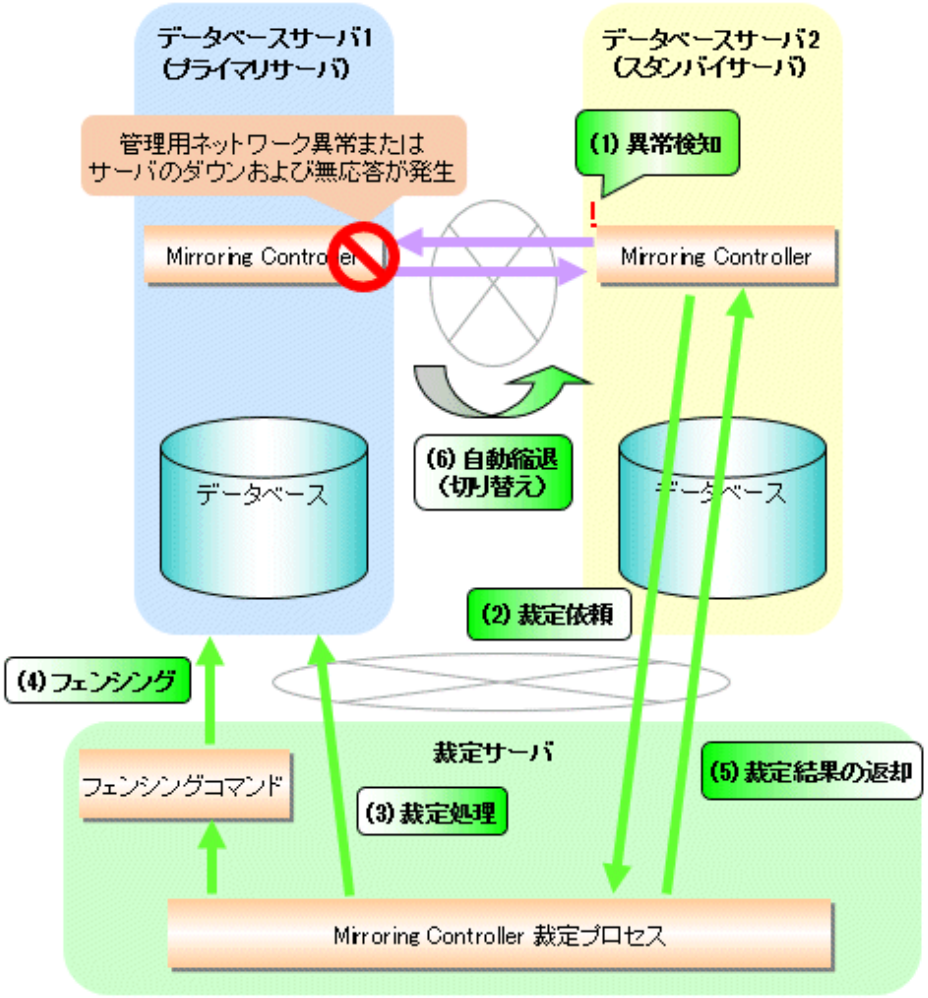


図2.2 スタンバイサーバのMirroring ControllerでOS/サーバの異常を検知した場合



2.11.4.1.2 縮退の判断を行うユーザーコマンドを呼び出して自動縮退を行う運用の異常監視のチューニング

縮退の判断を行うユーザーコマンドを呼び出して自動縮退を行う運用では、データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータおよび裁定処理とフェンシングに関連するパラメータを編集することによって、OS/サーバの異常検知から自動縮退までの時間の最適化を行うことができます。データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータの詳細は、"データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータ"を参照してください。

表2.12 裁定処理とフェンシングに関するパラメータ

パラメータ	指針
裁定処理のタイムアウト時間 (arbitration_command_timeout)	裁定コマンドの実行にかかる時間を考慮します (秒数)。
フェンシングのタイムアウト時間 (fencing_command_timeout)	フェンシングコマンドの実行にかかる時間を考慮します (秒数)。

OS/サーバの異常検知から自動縮退までの流れ

縮退の判断を行うユーザーコマンドを呼び出して自動縮退を行う運用のOS/サーバの異常検知から自動縮退が発生するまでの流れと関連するパラメータを示します。

OS/サーバの異常検知から自動縮退までの流れ	説明	関連するパラメータ
(1) 異常検知	Mirroring Controllerがデータベースサーバの OS/サーバの異常を検知します。	データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータ
(2) 裁定処理	裁定コマンドを実行し、相手サーバのOS/サーバの状態を確認します。	データベースサーバのサーバ定義ファイルの arbitration_command_timeout
(3) フェンシング	(2)でOS/サーバの異常と判断した場合、相手サーバをフェンシングし、クラスタシステムから隔離します。(2)で相手サーバのOS/サーバの状態を正常と判断した場合は、本処理および(4)は実施しません。	データベースサーバのサーバ定義ファイルの fencing_command_timeout
(4) 自動縮退	自動縮退を行います。(3)でフェンシングが失敗した場合は、本処理は実施しません。	-

-:関連パラメータなし

図2.3 プライマリサーバのMirroring ControllerでOS/サーバの異常を検知した場合

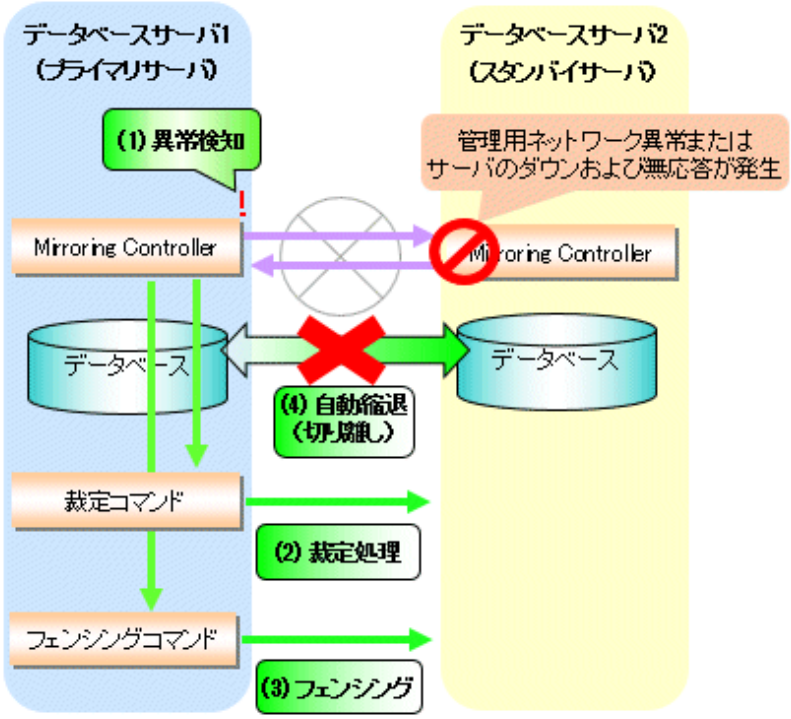
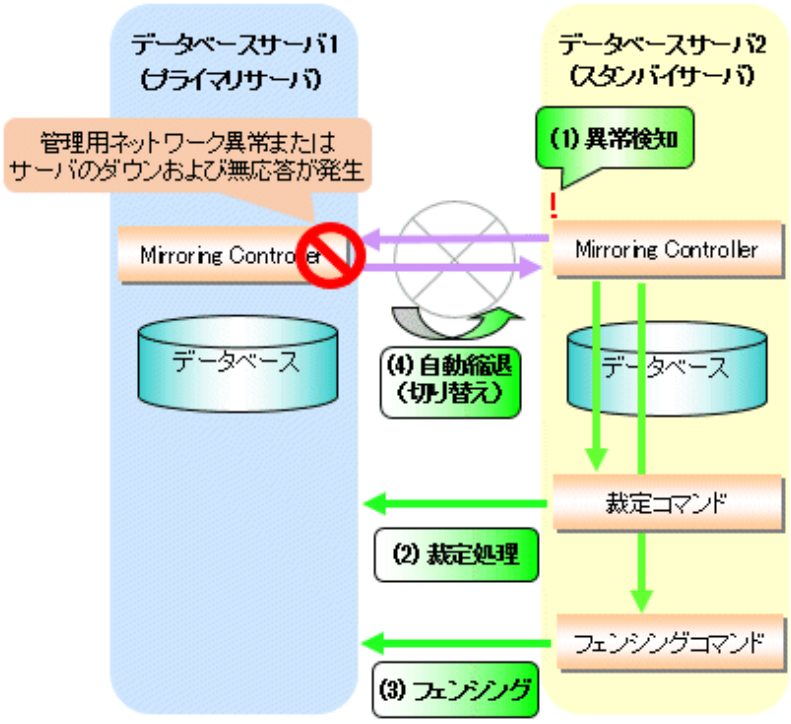


図2.4 スタンバイサーバのMirroring ControllerでOS/サーバの異常を検知した場合



2.11.4.1.3 メッセージを通知する運用の異常監視のチューニング

メッセージを通知する運用では、データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータを編集することによって、OS/サーバの異常検知時間の最適化を行うことができます。データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータの詳細は、"データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータ"を参照してください。なお、メッセージを通知する運用では、Mirroring Controllerが異常を検知した場合に、裁定処理、フェンシングおよび自動縮退は行わず、メッセージの通知のみ行います。

2.11.4.1.4 ハートビート異常で無条件に自動縮退を行う運用の異常監視のチューニング

ハートビート異常で無条件に自動縮退を行う運用では、データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータを編集することによって、OS/サーバの異常検知から自動縮退までの時間の最適化を行うことができます。データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータの詳細は、"データベースサーバのサーバ定義ファイルのOS/サーバの異常監視のパラメータ"を参照してください。なお、ハートビート異常で無条件に自動縮退を行う運用では、Mirroring Controllerが異常を検知した場合に、裁定処理とフェンシングは行わず、無条件に自動縮退を行います。

注意

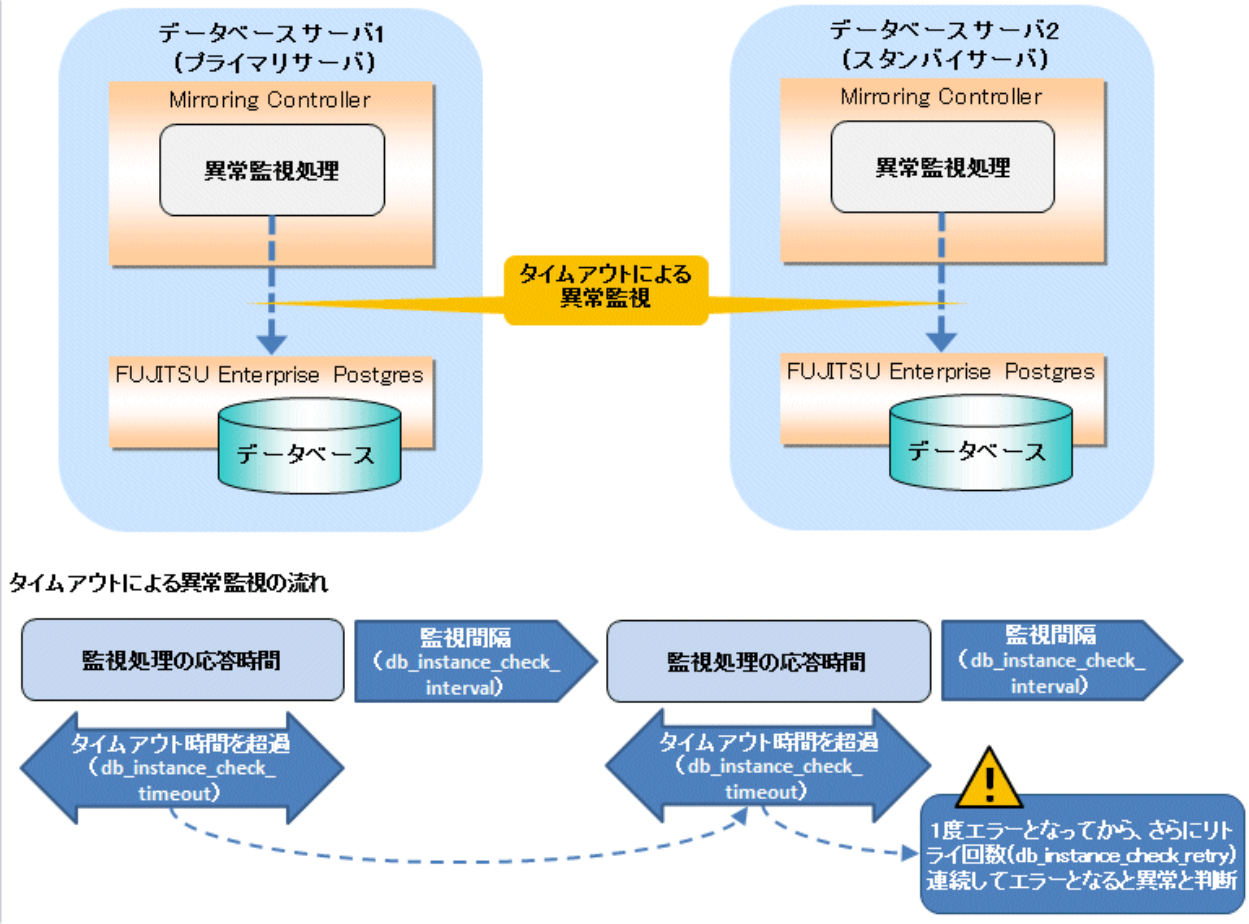
ハートビート異常で無条件に自動縮退を行う運用の注意事項は、"付録D ハートビート異常で即時に自動縮退を行うを選択した場合の注意事項"を参照してください。

2.11.4.2 データベースプロセスの異常監視のチューニング

データベースプロセスの異常監視は、データベースサーバのサーバ定義ファイルの以下のパラメータを編集することによって、最適化を行うことができます。

表2.13 データベースプロセスの異常監視のパラメータ

パラメータ	指針
異常監視の監視間隔 (db_instance_check_interval)	Mirroring Controllerによる異常監視がシステムに負荷を与えない範囲で設定しますが、通常は設定不要です(デフォルトはheartbeat_intervalの設定値です) (ミリ秒数)。
データベースプロセスの異常監視のタイムアウト時間 (db_instance_check_timeout)	データベースに対して負荷が継続的にかかる時間を考慮します。例えば、高負荷のバッチ業務を実行する場合や高多重のオンライン業務が連続的に発生するような場合が想定されます(デフォルトはheartbeat_timeoutの設定値です) (秒数)。
異常監視のリトライ回数 (db_instance_check_retry)	負荷が流動的なシステムなど、db_instance_check_timeoutで指定した値を超える事象に対して安全値が必要な場合に設定しますが、通常は設定不要です(デフォルトはheartbeat_retryの設定値です) (回数)。



異常と判断するまでの時間の計算方法を以下に示します。

```
異常検出時間 ＝ ( db_instance_check_timeout(秒数) ＋ db_instance_check_interval(ミリ秒) / 1000 ) ×
( db_instance_check_retry(回数) + 1 )
```

デフォルトの値を使用した場合の異常検出時間は以下となります。

```
異常検出時間 ＝ ( 1 + 800 / 1000 ) × ( 2 + 1 )
             ＝ 5.4(秒)
```

注意

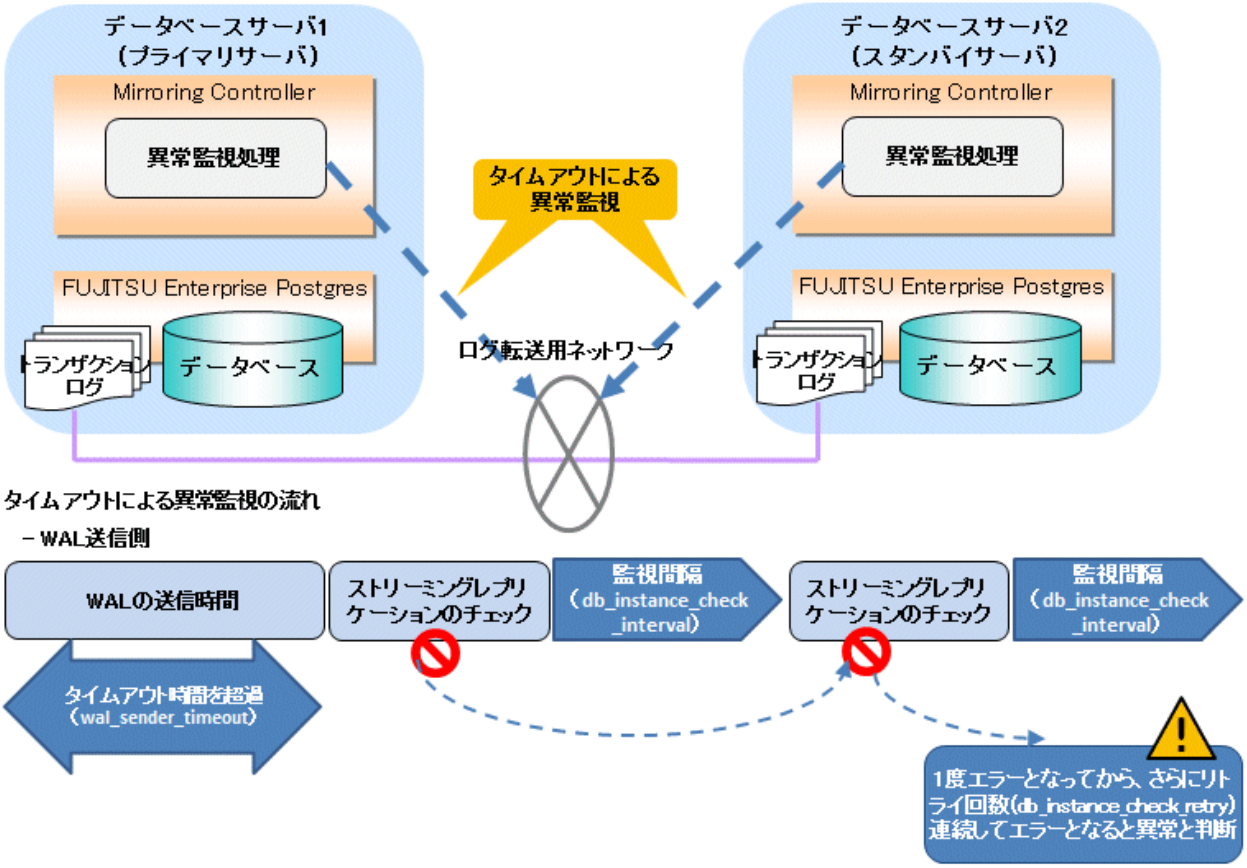
- サーバ識別子.confファイルのdb_instance_timeout_actionにmessageを指定した場合、db_instance_check_timeoutの指定値が短いと、データベースプロセスのダウンを無応答として検知し、自動縮退が発生するまでに時間がかかる場合があります。そのため、db_instance_check_timeoutには、適切なタイムアウト時間を指定してください。
- データベースに対する高負荷とインスタンスに接続できない事象が同時に発生した場合は、監視のリトライは行わずに異常と判断します。

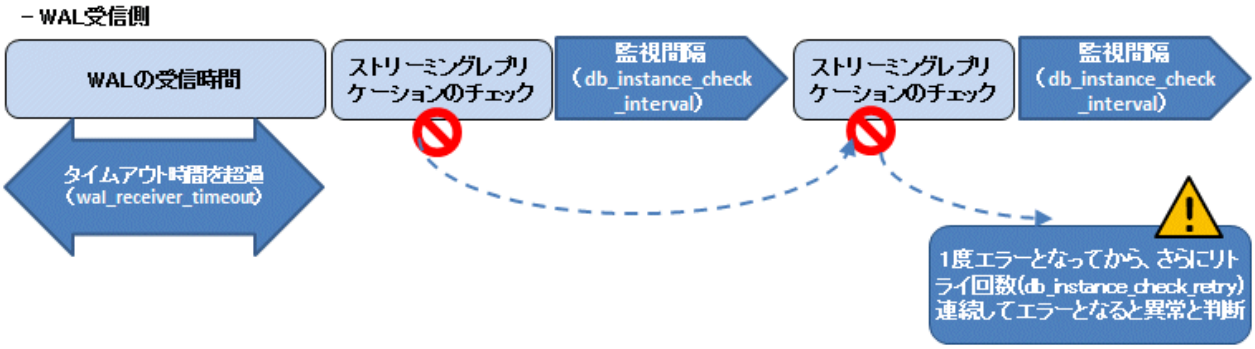
2.11.4.3 ストリーミングレプリケーションの異常監視のチューニング

ストリーミングレプリケーションの異常監視は、データベースサーバのサーバ定義ファイルの以下のパラメータを編集することによって、最適化を行うことができます。

表2.14 ストリーミングレプリケーションの異常監視のパラメータ

パラメータ	指針
異常監視の監視間隔 (db_instance_check_interval)	Mirroring Controllerによる異常監視がシステムに負荷を与えない範囲で設定しますが、通常は設定不要です(デフォルトはheartbeat_intervalの設定値です) (ミリ秒数)。
異常監視のリトライ回数 (db_instance_check_retry)	一時的にログ転送LAN異常が発生する場合など、安全値が必要な場合に設定しますが、通常は設定不要です(デフォルトはheartbeat_retryの設定値です) (回数)。
ストリーミングレプリケーションの異常監視のタイムアウト時間 (postgresql.conf内の wal_sender_timeoutおよび wal_receiver_timeout)	ログ転送用ネットワークの能力および負荷やデータベースに対する負荷が継続的に掛かる時間を考慮します。例えば、WALが大量に発生するようなデータ更新を行う業務が連続的に発生するような場合には誤検知を回避する設定が必要です (デフォルトは60秒)。





異常と判断するまでの時間の計算方法を以下に示します。

```
異常検出時間
＝ wal_sender_timeout(秒数) + ( db_instance_check_interval(ミリ秒) / 1000 × ( disk_check_retry(回数) + 1 ) )　または
＝ wal_receiver_timeout(秒数) + ( db_instance_check_interval(ミリ秒) / 1000 × ( disk_check_retry(回数) + 1 ) )
```

デフォルトの値を使用した場合の異常検出時間は以下となります。

```
異常検出時間 ＝ 60 + (800 / 1000 × (2 + 1))
             ＝ 62.4(秒)
```

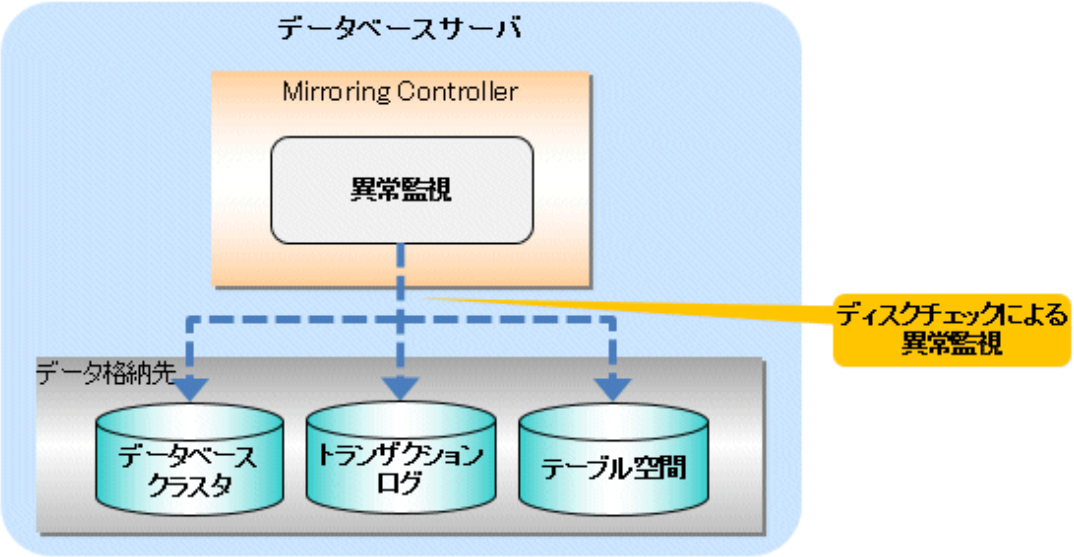
2.11.4.4 ディスクの異常監視のチューニング

ディスクの異常監視は、データベースサーバのサーバ定義ファイルの以下のパラメータを編集することによって、最適化を行うことができます。

表2.15 ディスクの異常監視のパラメータ

パラメータ	指針
異常監視の監視間隔 (disk_check_interval)	Mirroring Controllerによる異常監視がシステムに負荷を与えない範囲で設定します。ディスクアクセス時間より大きい値を設定してください(デフォルトはheartbeat_intervalの設定値です)(ミリ秒数)。
異常監視のリトライ回数 (disk_check_retry)	一時的なディスクの入出力障害が発生する可能性が想定される場合など、安全値が必要な場合に設定しますが、通常は設定不要です(デフォルトはheartbeat_retryの設定値です)(回数)。
異常監視のタイムアウト時間 (disk_check_timeout) (注1)	ディスク異常が発生した時点の次のdisk_check_intervalの開始時刻からタイムアウトによる異常と判断するまで許容する時間(デフォルトは2147483です)(秒数)。 0～2147483までの間の整数が指定可能です。
異常監視に利用するスレッド数の上限 (disk_check_max_threads) (注1)	ディスク監視用スレッド数の上限(デフォルトはJVMが利用可能なプロセッサ数です) 1～2147483647までの間の整数が指定可能ですが、マシンが利用可能なスレッド以上の値を設定するとシステムエラーになる可能性があります。 監視プロセスとは別に、mc_ctl statusコマンドを実行する場合に、mc_ctl statusごとに監視プロセスと同じ数量のスレッドが一時的に利用されます。disk_check_max_threadsを設定する際に、マシンのスレッド上限、利用予定のテーブル空間数、および同時に実行する可能性がある mc_ctl statusコマンドの数を考慮し、設定してください。

注1) PH23295を含む緊急修正を適用することで利用できます。



ディスクの異常監視ではディスクチェックを行い、異常検出時間またはdisk_check_timeoutに設定された時間内で、どちらか先に異常を検知した時点で縮退が実施されます。ただし、スタンバイサーバでdisk_check_timeoutによる異常検知時に、スタンバイサーバの切り離しを行うためには、shutdown_detached_synchronous_standbyにonを設定する必要があります。

異常と判断するまでの時間の計算方法を以下に示します。

```
異常検出時間 ＝ disk_check_interval（ミリ秒）/ 1000 × （ disk_check_retry(回数) ＋ 1 ）
```

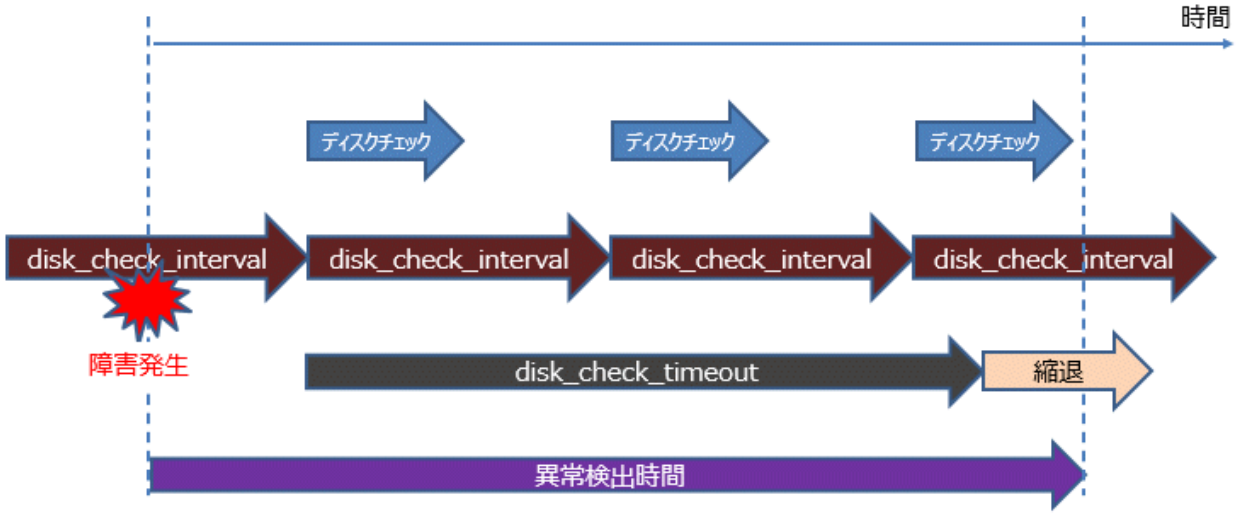
デフォルトの値を使用した場合の異常検出時間は以下となります。

```
異常検出時間 ＝ 800 / 1000  × （ 2 ＋ 1 ）
             ＝ 2.4(秒)
```

ディスクの異常監視を検出する例を以下に示します。

例1)

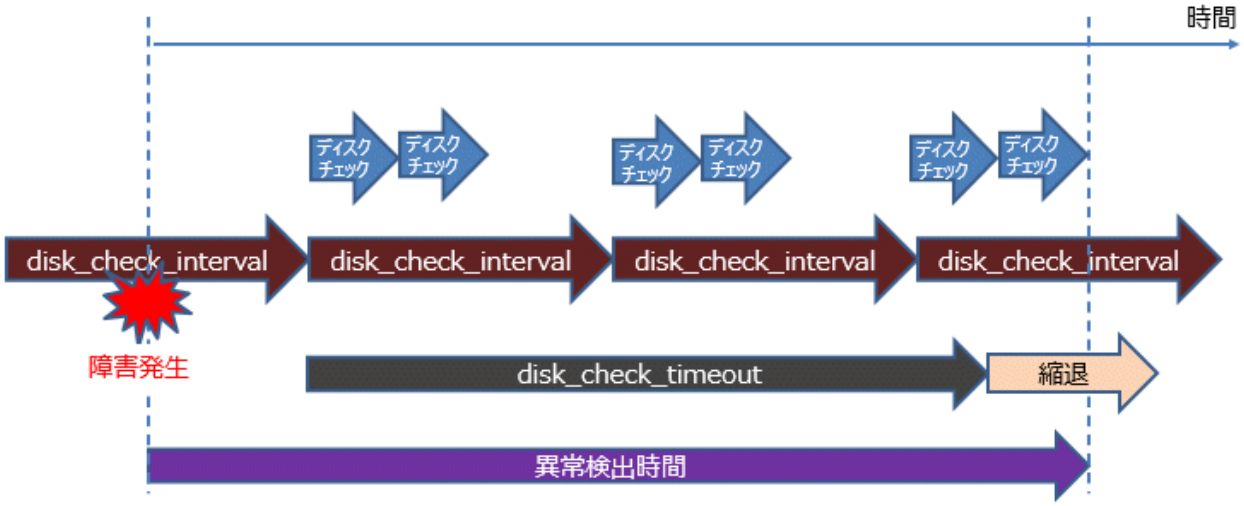
1つのスレッドが1つのディスクを監視、disk_check_retry = 2を設定。
ディスクに対するレスポンスが遅いため、ディスクアクセス時間内に読み込みまたは書き込みが完了できない場合。



例2)

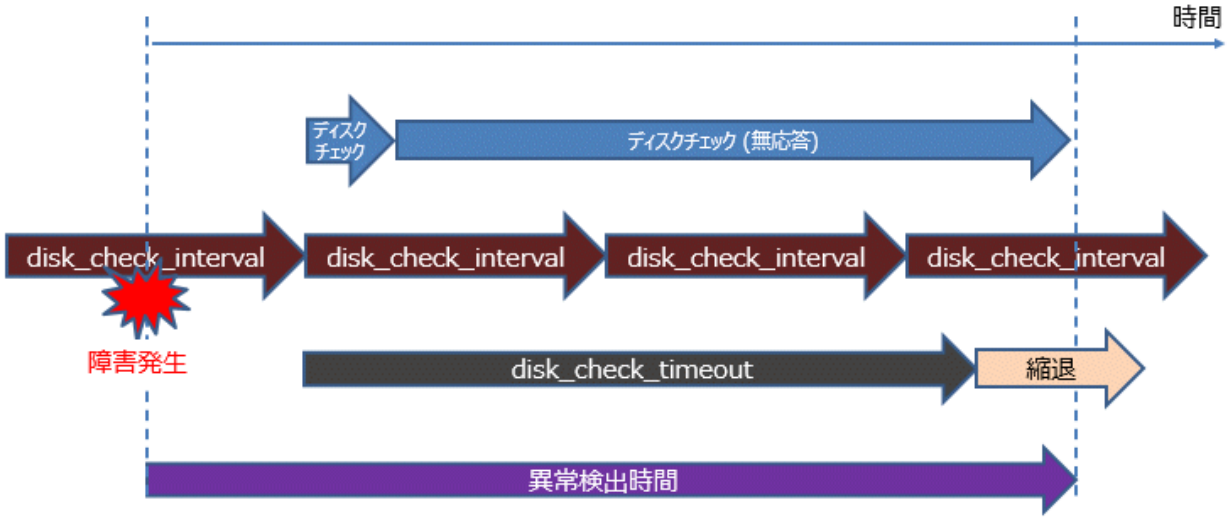
1つのスレッドが2つのディスクを監視、disk_check_retry = 2を設定。
ディスクに対するレスポンスが遅いため、ディスク1は、1回目、2回目にディスクアクセス時間内に読み込みまたは書き込みが完了できず、

3回目のリトライで読み込みまたは書き込みが成功。ディスク2はディスクアクセス時間内に読み込みまたは書き込みがすべて失敗した場合。



例3)

1つのスレッドが2つのディスクを監視、disk_check_retry = 2を設定。
ディスク2が無応答となり、タイムアウト時間内に監視結果が取得できない場合。



注意

- 上記のチューニングは、タイムアウトを検知してからプライマリサーバを切り替えるなどの操作を行うまでの時間に影響します。そのため、誤検知を行わない設計を行った上で、切り替え/切り離し時間を考慮した値に補正してください。
- OSやサーバの生死監視においてハートビート異常で即時に自動縮退を行う選択を行った場合には、スプリットブレインを発生させる危険性があります。詳細は、“付録D ハートビート異常で即時に自動縮退を行うを選択した場合の注意事項“を参照してください。

参考

監視対象の資源によっては、Mirroring Controllerがデータベースインスタンスへの接続やSQLアクセスにより異常監視を行うものがあります。この異常監視を行うための接続先データベース名や接続ユーザー名は、サーバ定義ファイル内のパラメータに従います。また、アプリケーション名は‘mc_agent’です。

2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定

データベースサーバのOSの起動・停止に合わせて、多重化したインスタンスとMirroring Controllerを自動的に起動・停止することができます。

注意

プライマリサーバとスタンバイサーバのMirroring Controllerの起動順序は保証しません。プライマリサーバの起動を確認したあとに、スタンバイサーバの起動を順次行って起動順序を保証してください。

また、データベースサーバのMirroring Controllerプロセスと裁定サーバのMirroring Controller裁定プロセスの起動順序は保証しません。裁定サーバを最初に起動できない場合には、mc_ctlコマンドのstartモードに--async-connect-arbiterオプションを指定して実行し、Mirroring Controllerプロセスを起動してください。

多重化したインスタンスとMirroring Controllerを起動する場合、時刻補正やネットワーク環境の開設などを待ち合わせるようにしてください。

以下の手順で行います。

1. unitファイルの作成

   以下に格納されているunitファイルのサンプルをコピーして、対象とするインスタンスにあわせて修正します。

   サンプルファイルのパス

   ```
   /インストールディレクトリ/share/mcoi.service.sample
   ```

   例）

   インストールディレクトリが“/opt/fsepv<*x*>server64”、インスタンス名を“inst1”とした場合の例を示します。“<*x*>”は製品のバージョンです。

   ```
   # cp /opt/fsepv<x>server64/share/mcoi.service.sample /usr/lib/systemd/system/mcoi_inst1.service
   ```

   unitファイルは、以下のオプションの下線部分を修正します。

セクション	オプション	指定値	説明
Unit	Description	FUJITSU Enterprise Postgres Mirroring Controller <インスタンス名>	機能概要を記述します。 対象とするインスタンス名を記述します。(注1)
Service	ExecStart	/bin/bash -c 'インストールディレクトリ/bin/mc_std start インストールディレクトリ Mirroring Controller管理ディレクトリ mc_ctl オプション'	サービスの起動時に実行するコマンドを指定します。 mc_ctlオプションには、-Mオプション以外のmc_ctlコマンドの実行時に追加したいオプションを指定します。 なお、このmc_ctlオプションの指定内容は、指定値のmc_stdコマンドからmc_ctlコマンドに引き継がれます。(注2)
	ExecStop	/bin/bash -c 'インストールディレクトリ/bin/mc_std stop インストールディレクトリ Mirroring Controller管理ディレクトリ mc_ctl オプション'	サービスの停止時に実行するコマンドを指定します。 mc_ctlオプションには、-Mオプション以外のmc_ctlコマンドの実行時に追加したいオプションを指定します。

セクション	オプション	指定値	説明
			ただし、停止時に--mc-onlyオプションを使いMirroring Controllerだけを停止するには、起動時に--mc-onlyオプションを使用して起動しておく必要があります。 なお、このmc_ctlオプションの指定内容は、指定値のmc_stdコマンドからmc_ctlコマンドに引き継がれます。
	User	ユーザー	インスタンス管理者のOSのユーザーアカウントを指定します。
	Group	グループ	インスタンス管理者のユーザーが所属するグループを指定します。

注1) ここで使用するインスタンス名とは、インスタンスを識別できる名称を指します。

インスタンスを識別するための命名規約は以下です。

- 16バイト以内
- 先頭の文字は、ASCIIアルファベット
- 先頭以外の文字は、ASCIIアルファベット、ASCII数字

注2) 裁定サーバを利用して自動縮退を行う運用の場合、最初に裁定サーバのMirroring Controller裁定プロセスを起動させた後にデータベースサーバのMirroring Controllerプロセスを起動させてください。裁定サーバを最初に起動させることができない場合には、--async-connect-arbiterオプションを指定して実行し、Mirroring Controllerプロセスを起動してください。

2. 自動起動・停止の有効化

OSのスーパーユーザーで、systemctlコマンドを使って自動起動・停止を有効にします。

例）

```
# systemctl enable mcoi_inst1.service
```

注意

Mirroring Controllerの自動起動・停止の設定を実施している場合、Mirroring Controllerの停止は、mc_ctlコマンドではなく、OSのスーパーユーザーで、systemctlコマンドを使用して停止してください。

例)

```
# systemctl stop mcoi_inst1.service
```

停止しない場合は、“運用ガイド”の“インスタンス停止失敗時の対処”を参照して、インスタンスを停止します。そのあと、mc_ctlコマンドに-eオプションを指定してMirroring Controllerを強制停止してください。

例)

```
$ mc_ctl stop -M /mcdir/inst1 -e
```

mc_ctlコマンドでMirroring Controllerを停止した場合、システムログに下記のメッセージが出力されますが、systemdによる自動停止が実行されたためであり、問題ありません。

メッセージ

```
FATAL: Mirroring Controllerの停止に失敗しました 対象サーバ：″{0}″ (MCA00043)
```

2.13 Mirroring Controller裁定プロセスの自動起動・停止の設定

裁定サーバのOSの起動・停止に合わせて、Mirroring Controller裁定プロセスを自動的に起動・停止することができます。

注意

Mirroring Controller裁定プロセスを起動する場合、時刻補正やネットワーク環境の開設などを待ち合わせるようにしてください。

以下の手順で行います。

1. unitファイルの作成

   以下に格納されているunitファイルのサンプルをコピーして、対象とするインスタンスにあわせて修正します。

   サンプルファイルのパス

```
/インストールディレクトリ/share/mcarboi.service.sample
```

   例）
   インストールディレクトリが“/opt/fsepv<*x*>assistant”、裁定プロセスの識別子名を“arbiter1”とした場合の例を示します。
   “<*x*>”は製品のバージョンです。

```
# cp /opt/fsepv<x>assistant/share/mcarboi.service.sample /usr/lib/systemd/system/mcarboi_arbiter1.service
```

   unitファイルは、以下のオプションの下線部分を修正します。

セクション	オプション	指定値	説明
Unit	Description	FUJITSU Enterprise Postgres Mirroring Controller Arbiter <裁定プロセスの識別子名>	機能概要を記述します。 対象とする裁定プロセスの識別子名を記述します。(注)
Service	ExecStart	/bin/bash -c 'インストールディレクトリ/bin/mc_arb_std start インストールディレクトリ Mirroring Controller裁定プロセスの管理ディレクトリ mc_arbオプション'	サービスの起動時に実行するコマンドを指定します。 mc_arbオプションには、-Mオプション以外の、mc_arbコマンドの実行時に追加したいオプションを指定します。 なお、このmc_arbオプションの指定内容は、指定値のmc_arb_stdコマンドからmc_arbコマンドに引き継がれます。
	ExecStop	/bin/bash -c 'インストールディレクトリ/bin/ mc_arb_std stop インストールディレクトリ Mirroring Controller裁定プロセスの管理ディレクトリ mc_arbオプション'	サービスの停止時に実行するコマンドを指定します。 mc_arbオプションには、-Mオプション以外のmc_arbコマンドの実行時に追加したいオプションを指定します。 なお、このmc_arbオプションの指定内容は、指定値のmc_arb_stdコマンドから

セクション	オプション	指定値	説明
			mc_arbコマンドに引き継がれます。
	User	ユーザー	OSのユーザーアカウントを指定します。
	Group	グループ	ユーザーが所属するグループを指定します。

注) ここで使用する裁定プロセスの識別子名とは、Mirroring Controller裁定プロセスを識別できる名称を指します。

Mirroring Controller裁定プロセスを識別するための命名規約は以下です。

・16バイト以内

・先頭の文字は、ASCIIアルファベット

・先頭以外の文字は、ASCIIアルファベット、ASCII数字

2. 自動起動・停止の有効化

OSのスーパーユーザーで、systemctlコマンドを使って自動起動・停止を有効にします。

例)

```
# systemctl enable mcarboi_arbiter1.service
```

2.14 バックアップ運用

データベース多重化運用におけるバックアップ運用について、説明します。

2.14.1 データベース多重化運用情報のバックアップ

Mirroring Controllerの設定を変更する場合には、データベースのバックアップに加えてMirroring Controllerの設定が失われないように、Mirroring Controller管理ディレクトリ配下の設定ファイルをバックアップしてください。

また、裁定サーバを利用して自動縮退を行う運用の場合は、Mirroring Controller裁定プロセスの管理ディレクトリ配下の設定ファイルもバックアップしてください。

2.14.2 データベースのバックアップ運用

データベース多重化運用を行うということは、ディスク故障に備えてスタンバイサーバにバックアップデータを取得していることと同じです。しかし、何らかの原因ですべてのサーバのディスクが破壊される可能性もあります。

このような場合に備えて、プライマリサーバ上でpgx_dmpallコマンドを実行してバックアップデータを取得してください。しかし、どのサーバがプライマリサーバとして動作するかは不定であるため、すべてのサーバでpgx_dmpallコマンドを定期的に実行するようにしてください。例えば、バックアップデータを取得するためのスクリプトを作成して運用管理ソフトウェアに設定します。

ポイント

スタンバイサーバでpgx_dmpallコマンドを実行すると、状況に一致していませんが、以下のエラーメッセージが出力されてコマンドは復帰値1を返却します。

すべてのサーバでこのようなエラーのみを無視するようなスクリプトを実行すれば、プライマリサーバのバックアップデータを取得することができます。

エラーメッセージ

```
ERROR:  リカバリーはすでに実行中です (10095)
```

注意

- プライマリサーバとして稼動していたサーバがそのバックアップデータとともに損壊する場合を考慮して、もう一方のサーバがプライマリサーバへと昇格したあとに、次のバックアップ予定を待たずに新プライマリサーバでバックアップすることを推奨します。
- プライマリサーバとスタンバイサーバで同じバックアップディレクトリ名を指定してください。異なるバックアップディレクトリ名を指定すると、他方のサーバのバックアップデータを使ってリカバリを行った場合に、正しく行えません。

参照

- バックアップは定期的に行う方が、リカバリ時間の短縮やディスク使用率の削減の効果があります。バックアップの詳細は、“運用ガイド”の“データベースのバックアップ”を参照してください。
- pgx_dmpallコマンドで採取したバックアップデータをもとにした復旧する方法は、“第4章 データベース多重化運用の異常時の対処”を参照してください。

第3章 データベース多重化運用の操作

本章では、データベース多重化運用をした場合に行う定期的な運用操作について説明します。

定期的な運用操作については、単一サーバでの運用操作と同じです。

参照

定期的な運用操作の詳細については、“運用ガイド”の“定期的な運用操作”を参照してください。

3.1 Mirroring Controller裁定プロセスの起動と停止

Mirroring Controller裁定プロセスの起動と停止について説明します。

3.1.1 Mirroring Controller裁定プロセスの起動

Mirroring Controller裁定プロセスが未起動の状態で、mc_arbコマンドをstartモードで実行して、Mirroring Controller裁定プロセスを起動します。

例）

```
$ mc_arb start -M /mcarb_dir/arbiter1
```

参照

mc_arbコマンドの指定方法の詳細については、“リファレンス”を参照してください。

3.1.2 Mirroring Controller裁定プロセスの停止

Mirroring Controller裁定プロセスが起動された状態で、mc_arbコマンドをstopモードで実行して、Mirroring Controller裁定プロセスを停止します。

例）

```
$ mc_arb stop -M /mcarb_dir/arbiter1
```

参照

mc_arbコマンドの指定方法の詳細については、“リファレンス”を参照してください。

注意

- サービス停止によって裁定サーバを停止する場合は強制停止になります。
- 裁定サーバのOSをシャットダウンする前に、必ずプライマリサーバ、スタンバイサーバのMirroring Controllerを停止するか、またはプライマリサーバ、スタンバイサーバのOSをシャットダウンしてください。

3.2 Mirroring Controllerの起動と停止

データベース多重化運用を行う場合は、mc_ctlコマンドを使用してインスタンスとMirroring Controllerを同時に起動・停止してください。

なお、インスタンスだけを起動・停止しないでください。

Mirroring Controllerの起動

Mirroring Controllerが未起動の状態で、mc_ctlコマンドをstartモードで実行して、Mirroring Controllerを起動します。

自動切り替え/切り離しを有効化する場合

mc_ctlコマンドのstartモードを実行してください。

例）

```
$ mc_ctl start -M /mcdir/inst1
```

インスタンスのみを起動・停止した場合には、以下の影響があります。

－ インスタンスのみを起動した場合

Mirroring Controllerが起動されるまでのあいだ、自動切り替え/自動切り離しなどの機能は動作できません。

－ インスタンスのみを停止した場合

Mirroring Controllerがインスタンスに異常が発生したと判断して、不必要な自動切り替えを行ってしまいます。

また、自動切り替えが正常に動作しなくなる場合があります。

自動切り替え/切り離しを有効化しない場合

mc_ctlコマンドのstartモードに-Fオプションを指定して実行してください。

例）

```
$ mc_ctl start -M /mcdir/inst1 -F
```

インスタンスのみを起動・停止した場合には、以下の影響があります。

－ インスタンスのみを起動した場合

Mirroring Controllerが起動されるまでのあいだ、“1.1 データベース多重化運用とは”に示す異常を検知できません。

－ インスタンスのみを停止した場合

Mirroring Controllerがインスタンスに異常が発生したと判断して、システムログにエラーを出力してしまいます。

ポイント

- Mirroring Controllerのプロセスのみを起動する場合には、mc_ctlコマンドのstartモードに--mc-onlyオプションを指定することで行えます。
- 自動切り替え/切り離しはMirroring Controllerが起動したあとに、mc_ctlコマンドのenable-failoverモード、または、disable-failoverモードを使用することにより自動切り替え/切り離しを有効、または無効にすることが可能です。
- 裁定サーバを利用して自動縮退を行う運用では、事前に裁定サーバでMirroring Controller裁定プロセスを起動していない場合、データベースサーバのMirroring Controllerプロセスの起動が失敗します。ただし、事前にMirroring Controller裁定プロセスを起動できない場合であっても、mc_ctlコマンドに--async-connect-arbiter オプションを指定することで、Mirroring Controllerプロセスを起動することができます。

注意

- 裁定サーバを利用して自動縮退を行う運用の場合、データベースサーバが裁定サーバに接続する時間が加算されるため、裁定サーバを利用しない運用に比べて、Mirroring Controllerの起動時間が長く掛かる場合があります。
- セットアップ直後に、スタンバイサーバを誤ってプライマリサーバとして起動した場合や、切り替えが行われた後の旧プラマリサーバを復旧せずに、誤ってプライマリサーバとして起動した場合、通常はMirroring Controllerの起動が失敗しますが、万が一、管理用ネッ

トワークが切断されていると起動を失敗させることができず、両サーバがプライマリサーバとなってしまう可能性があります。そのため、Mirroring Controllerの起動は、管理用ネットワークが接続していることを確認してから行ってください。

Mirroring Controllerの停止

Mirroring Controllerが起動された状態で、mc_ctlコマンドをstopモードで実行して、Mirroring Controllerを停止します。

例）

```
$ mc_ctl stop -M /mcdir/inst1
```

ポイント

Mirroring Controllerのプロセスのみを停止する場合には、mc_ctlコマンドのstopモードに--mc-onlyオプションを指定することで行えます。

注意

意図しない自動切り替えを発生させないためには、プライマリサーバのOSをシャットダウンする前に、必ずMirroring Controllerを停止するか、またはスタンバイサーバのOSをシャットダウンしてください。

参照

mc_ctlコマンドの指定方法の詳細については、“リファレンス”を参照してください。

3.3 データベース多重化運用の状態確認

3.3.1 データベースサーバでの状態確認

データベースサーバでの状態確認について説明します。

多重化したデータベースの状態は、mc_ctlコマンドをstatusモードで実行することで確認します。

また、Mirroring Controllerのメッセージを監視することで異常を検知できます。定期的に状態やメッセージを監視すると、自動切り替えに失敗した場合にも迅速に対処できます。

多重化したデータベースの状態確認

mc_ctlコマンドを実行すると、多重化の構成の内容、異常が発生したときに切り替えが可能な状態であるかの情報、および切り替えや切り離しの原因となった異常の箇所と内容を表示します。データベース多重化運用を開始したあとは、mc_ctlコマンドのstatusモードで多重化の状態を確認するようにしてください。

例）
mc_ctlコマンドを実行した場合の状態表示例を以下に示します。

```
$ mc_ctl status -M /mcdir/inst1

mirroring status
----------------
switchable
server_id  host_role       host           host_status  db_proc_status         disk_status
-------------------------------------------------------------------------------------------------------------
server1    primary         192.0.2.100    normal       normal                 normal
server2    standby         192.0.2.110    normal       normal                 normal
```

Mirroring Controller裁定プロセスとの接続状態の確認

裁定サーバを利用して自動縮退を行う運用では、--arbiterオプションを指定することで、Mirroring Controller裁定プロセスとの接続状態を確認することができます。statusの表示が“online”になっている場合は、データベースサーバから裁定サーバへの裁定依頼ができる状態です。裁定サーバを利用して自動縮退を行う運用では、--arbiterオプションを指定したstatusモードを定期的に実行し、statusの表示が“online”になっていることを確認してください。

例）
mc_ctlコマンドに--arbiterオプション指定して実行した場合の状態表示例を以下に示します。

```
$ mc_ctl status --arbiter -M /mcdir/inst1

arbiter_id   host           status
----------------------------------------
arbiter      192.0.3.120    online
```

データの同期状況の確認

プライマリサーバにおいて統計情報ビューpg_stat_replicationを参照することで、データの同期の状況を確認できます。ただし、監視プログラムを作成する場合には、pg_stat_replicationが将来変更される可能性がある点に注意してください。

例）
この例は、送信と受信後のトランザクションログの位置(sent_lsn, replay_lsn)が一致しており、同期が完全に行われている状態を示しています。

```
postgres=# select * from pg_stat_replication;
-[ RECORD 1 ]----+------------------------------
pid              | 10651
usesysid         | 10
usename          | fsep
application_name | standby
client_addr      | 192.0.2.210
client_hostname  |
client_port      | 55098
backend_start    | 2022-03-23 11:17:49.628793+09
backend_xmin     |
state            | streaming
sent_lsn         | 0/3000060
write_lsn        | 0/3000060
flush_lsn        | 0/3000060
replay_lsn       | 0/3000060
write_lag        |
flush_lag        |
replay_lag       |
sync_priority    | 1
sync_state       | sync
reply_time       | 2022-03-23 11:23:27.703366+09
```

参照

- mc_ctlコマンドの詳細は、“リファレンス”を参照してください。
- アプリケーションの互換性の維持に関する詳細は、“アプリケーション開発ガイド”の“アプリケーションの互換に関する注意事項”を参照してください。
- pg_stat_replicationの詳細は、“PostgreSQL Documentation”の“Server Administration”の“The Statistics Collector”を参照してください。

3.3.2 裁定サーバでの状態確認

裁定サーバでの状態確認について説明します。

Mirroring Controller裁定プロセスとプライマリサーバ、およびスタンバイサーバの接続状態は、mc_arbコマンドをstatusモードで実行することで確認します。

mc_arbコマンドを実行した場合の状態表示例を以下に示します。

例）

```
$ mc_arb status -M /mcarb_dir/arbiter1

server_id    host           status
----------------------------------------
server1      192.0.3.100    online
server2      192.0.3.110    online
```

3.4 プライマリサーバの手動切り替え

以下の場合には、プライマリサーバの切り替えを自動で行いません。

- 自動切り替え/切り離しを無効化している場合
- OS/サーバの生死監視においてハートビート異常でメッセージ通知を選択し、OS/サーバのダウンまたは無応答が発生した場合

上記の場合、プライマリサーバまたはスタンバイサーバ上でmc_ctlコマンドのswitchモードを実行して、プライマリサーバの切り替えを手動で行います。

例）

```
$ mc_ctl switch -M /mcdir/inst1
```

ポイント

自動切り替え/自動切り離しを有効化している場合、任意のタイミングでプライマリサーバの切り替えを行う利用方法が可能です。

3.5 スタンバイサーバの手動切り離し

自動切り替え/自動切り離しを有効化している場合と無効化している場合で、スタンバイサーバの切り離しを行うために実行する手順が異なります。

自動切り替え/切り離しを有効化している場合

スタンバイサーバ上で、mc_ctlコマンドをstopモードで実行します。

例）

```
$ mc_ctl stop -M /mcdir/inst1
```

自動切り替え/切り離しを無効化している場合

1. スタンバイサーバ上で、mc_ctlコマンドをstopモードで実行します。

   例）

   ```
   $ mc_ctl stop -M /mcdir/inst1
   ```

2. プライマリサーバ上で、postgresql.confファイルのsynchronous_standby_namesパラメータをコメントアウトします。
3. プライマリサーバ上で、pg_ctlコマンドをreloadモードで実行します。

   例）

   ```
   $ pg_ctl reload -D /database/inst1
   ```

ポイント

systemdを使用してMirroring Controllerの自動起動・停止の設定を実施している場合は、mc_ctlコマンドではなく、systemctlコマンドを使用してください。詳細は、“2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定”を参照してください。

3.6 ハートビート異常検出時の操作

OS/サーバの生死監視によるハートビート異常を検知した場合、以下のメッセージが出力されます。

```
監視の対象"サーバ(サーバ識別子名)"において異常を検出しました: 無応答:ping timeout (MCA00019)
```

サーバ識別子.confファイルのheartbeat_error_actionの設定がmessageの場合、自動切り替え/切り離しを有効化してMirroring Controllerを起動した場合であっても、ハートビート異常検出時は自動で切り替え/切り離し処理は行われません。そのため、ユーザーによる操作が必要です。

heartbeat_error_actionの設定がmessageの場合のハートビート異常検出時の操作を説明します。

1. ハートビート異常の原因を特定してください。原因として、以下のいずれかが考えられます。
   - 相手OS/サーバのダウンまたは無応答
   - 管理用ネットワークの異常
2. 手順1で特定した原因により、対処を行ってください。
   - 相手OS/サーバのダウンまたは無応答

     mc_ctlコマンドによる手動切り替えまたは手動切り離しを実施してください。
   - 管理ネットワークの異常

     “第4章 データベース多重化運用の異常時の対処”を参照し、データベース多重化システムの復旧を行ってください。

3.7 Mirroring Controllerのメッセージ監視

Mirroring Controllerが出力するメッセージは、データベースサーバと裁定サーバのそれぞれで出力されます。自動切り替えに失敗した場合などは、運用の継続に関わる重要なメッセージが出力されることがあるため、メッセージを監視するようにしてください。

裁定サーバを利用して自動縮退を行う運用では、データベースサーバと裁定サーバのそれぞれでメッセージを監視してください。

データベースサーバのメッセージ出力先

システムログに出力されます。

裁定サーバのメッセージ出力先

システムログに出力されます。

ポイント

- 重要と考えるメッセージ種別を監視するためには、事前にOSの設定が必要となる場合があります。OSのマニュアルなどを参照して、監視対象とするメッセージ種別のメッセージが出力されるかを確認して必要があれば設定してください。
- サーバ識別子.confファイルのheartbeat_error_actionの設定がmessageの場合、OS/サーバの生死監視による異常検出時はメッセージ出力のみを行い、切り替え/切り離し処理は自動で行われないため、ユーザーがメッセージを監視する必要があります。詳細は、“3.6 ハートビート異常検出時の操作”を参照してください。

データベースサーバでの表示形式

```
プログラム名[プロセスID]: メッセージ種別: メッセージ内容(メッセージ番号)
```

プログラム名は、データベースサーバのサーバ識別子.confファイルのsyslog_identパラメータで指定します。

以下に、Mirroring Controllerが出力するメッセージの種別、重要度、およびシステムログとの対応関係を示します。

表3.1 メッセージ種別、重要度、およびシステムログとの対応関係

メッセージ種別	重要度	意味	システムログ
INFO	通常	LOG、NOTICEに該当しない情報を提供します。	INFO
LOG		動作履歴を追跡するにあたり、特に重要なイベントとして認識する情報を提供します。 (例:自動切り替えが完了した など)	
NOTICE	注意	実行した、または自動実行された結果に対して、プログラム内でユーザーの指示に配慮を加えた内容を出力します。	NOTICE
WARNING	警告	やがて多重化状態が維持できなくなるなどの警告するべき内容を提供します。	WARNING
ERROR	エラー	FATAL、PANIC以外のエラーが発生したことを報告します。	ERROR
FATAL		多重化したデータベースシステムで何らかの異常を検出したため、異常が発生したシステムの復旧を開始する必要があることとその異常の内容や原因を報告します。	CRIT
PANIC		多重化したデータベースシステムのすべてで何らかの異常を検出したため、直ちに復旧を開始する必要があることとその異常の内容や原因を報告します。	ALERT

メッセージの重要度は以下の意味を持ちます。

- 通常

  通常の状態です。システムから通知されているメッセージが表示されます。特に対処を実施する必要はありません。

- 注意

  通常の状態ですが、注意すべきメッセージが表示されます。必要に応じてメッセージに記載されている“利用者の処置”にしたがって対処を実施してください。

- 警告

  エラーではありませんが、利用者の確認、作業を求めています。メッセージに記載されている“利用者の処置”を行ってください。

- エラー

  エラーです。メッセージに記載されている“利用者の処置”を行ってください。

裁定サーバでの表示形式

```
プログラム名[プロセスID]: メッセージ種別: メッセージ内容(メッセージ番号)
```

プログラム名は、裁定サーバのarbitration.confファイルのsyslog_identパラメータで指定します。

以下に、Mirroring Controllerが出力するメッセージの種別、重要度、および出力先ログとの対応関係を示します。

メッセージ種別	重要度	意味	システムログ
INFO	通常	LOG、NOTICEに該当しない情報を提供します。	INFO
LOG		動作履歴を追跡するにあたり、特に重要なイベントとして認識する情報を提供します。 (例:自動切り替えが完了した など)	
NOTICE	注意	実行した、または自動実行された結果に対して、プログラム内でユーザーの指示に配慮を加えた内容を出力します。	NOTICE

メッセージ種別	重要度	意味	システムログ
WARNING	警告	裁定処理ができなくなるなどの警告するべき内容を提供します。	WARNING
ERROR	エラー	FATAL、PANIC以外のエラーが発生したことを報告します。	ERROR
FATAL		裁定サーバの異常を検出したため、異常が発生したシステムの復旧を開始する必要があることとその異常の内容や原因を報告します。	CRIT
PANIC		裁定サーバの異常を検出したため、直ちに復旧を開始する必要があることとその異常の内容や原因を報告します。	ALERT

メッセージの重要度は以下の意味を持ちます。

- 通常

  通常の状態です。システムから通知されているメッセージが表示されます。特に対処を実施する必要はありません。

- 注意

  通常の状態ですが、注意すべきメッセージが表示されます。必要に応じてメッセージに記載されている“利用者の処置”にしたがって対処を実施してください。

- 警告

  エラーではありませんが、利用者の確認、作業を求めています。メッセージに記載されている“利用者の処置”を行ってください。

- エラー

  エラーです。メッセージに記載されている“利用者の処置”を行ってください。

3.8 サーバの保守

サーバの定期点検やOSを含むソフトウェア製品のアップデートを適用するような保守作業を行う場合は、事前に計画的にサーバを停止して作業を行う必要があります。

3.8.1 ローリングアップデート

データベース多重化運用では、業務を継続しながら構成されたサーバの保守作業を行うローリングアップデートを実施できます。

まず、スタンバイサーバに対して保守作業を実施したあと、スタンバイサーバをプライマリサーバに切り替えます。次に、スタンバイサーバに切り替わった旧プライマリサーバに対して保守作業を実施することで、業務を継続しながら保守作業を行うことができます。

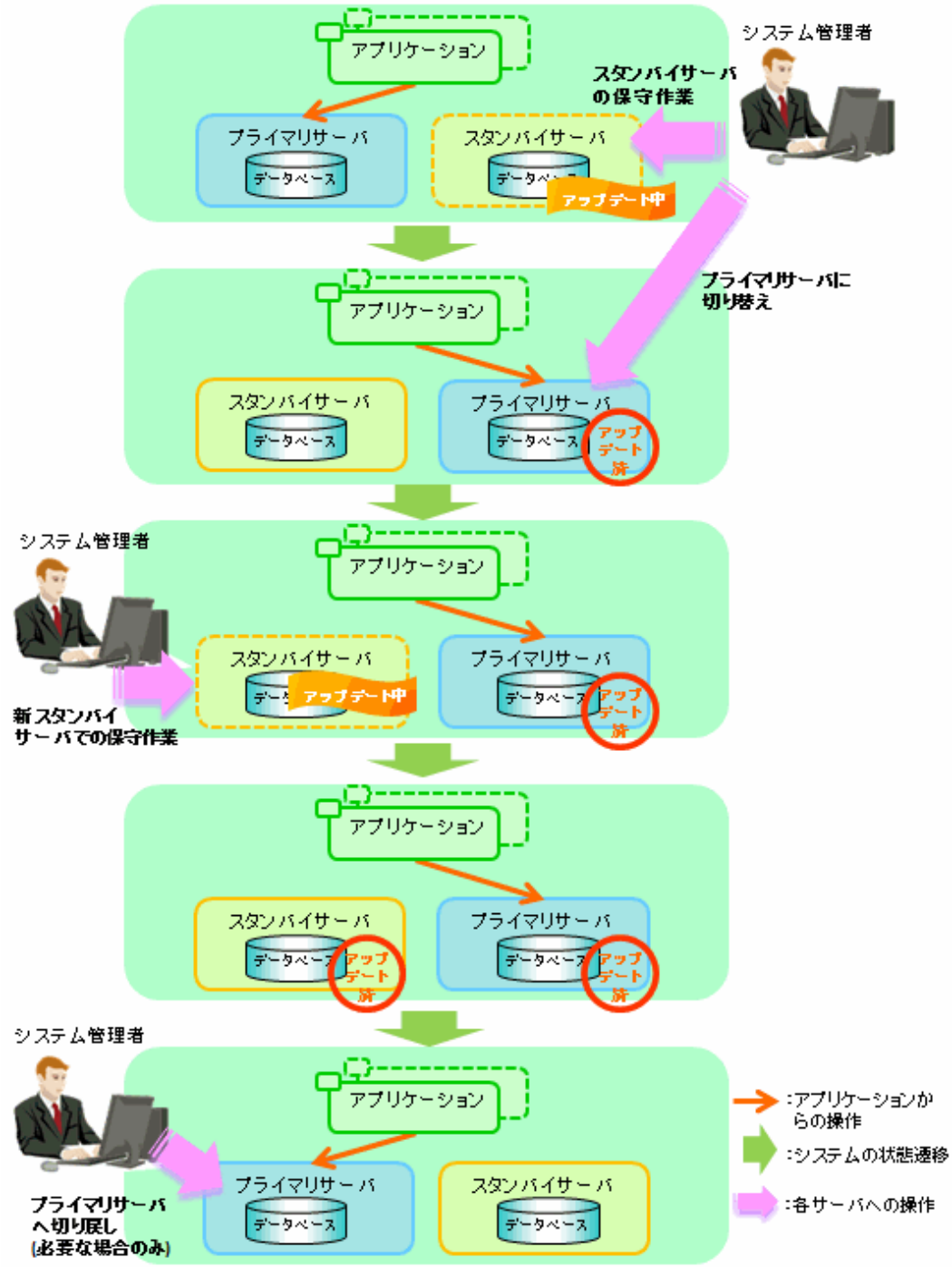
なお、裁定サーバはデータベースサーバの業務に影響なく保守が可能なため、ローリングアップデートでの考慮は不要です。

参照

スタンバイサーバへの保守作業で停止時間が長くなる場合は、“3.9.1 スタンバイサーバの停止を伴う変更”の“スタンバイサーバの停止時間”を参照してください。

以下は、ローリングアップデートのイメージです。

図3.1 ローリングアップデート



上図に対応して、以下の手順で行います。

スタンバイサーバの保守作業

1. スタンバイサーバにおいてサーバの保守作業を行うために、Mirroring Controllerを停止します。

   例）

```
$ mc_ctl stop -M /mcdir/inst1
```

2. 停止が完了したことを確認します。

   スタンバイサーバにおいてデータベースサーバのOSの起動・停止に合わせて、多重化したインスタンスおよびMirroring Controllerを自動的に起動・停止する設定にしていた場合は、自動的に起動・停止する設定を解除します。

参照

OSの起動・停止に合わせて、多重化したインスタンスおよびMirroring Controllerを自動的に起動・停止する設定の詳細は"2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定"を参照してください。

OSのスーパーユーザーで、systemctlコマンドを使用して自動起動・停止の無効化を行います。

例）
自動起動・停止するユニットを"mcoi_inst1.service"とした場合の例を以下に示します。

```
# systemctl disable mcoi_inst1.service
```

3. 保守作業を実施します。

4. スタンバイサーバにプライマリサーバのインスタンスの複製を作成します。

   pg_basebackupコマンドを実行して、スタンバイサーバのデータをプライマリサーバと同期して作成します。

   例）

```
$ pg_basebackup -D /database/inst1 -X fetch --waldir=/transaction/inst1 --progress --verbose -R --
dbname='application_name=スタンバイサーバ名' -h プライマリサーバのホスト名 -p プライマリサーバのポート番号
```

参照

スタンバイサーバにプライマリサーバのインスタンスを複製する手順は、スタンバイサーバをセットアップする手順と同じです。

"2.5.2 スタンバイサーバのインスタンスの作成・設定・登録"を参照して、リカバリを実施してください。

5. 多重化したインスタンスおよびMirroring Controllerを自動的に起動・停止する設定を確認します。

   手順2において、OSの起動・停止に合わせて、多重化したインスタンスおよびMirroring Controllerを自動的に起動・停止する設定を解除した場合は、再設定します。自動的に起動・停止する必要が無い場合は、この手順は省略できます。

   OSのスーパーユーザーで、systemctlコマンドを使用して自動起動・停止の有効化を行います。

   例）
   自動起動・停止するユニットを"mcoi_inst1.service"とした場合の例を以下に示します。

```
# systemctl enable mcoi_inst1.service
```

6. スタンバイサーバのMirroring Controllerを起動（組込み）します。

   本操作はスタンバイサーバでの保守作業を確定するうえで必要であるため、必ず実施してください。

   自動切り替え/切り離しを有効化する場合

   インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで実行します。

   例）

```
$ mc_ctl start -M /mcdir/inst1
```

   自動切り替え/切り離しを有効化しない場合

   インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで-Fオプションを指定して実行します。

   例）

```
$ mc_ctl start -M /mcdir/inst1 -F
```

ポイント

自動切り替え/切り離しはMirroring Controllerが起動したあとに、mc_ctlコマンドのenable-failoverモード、または、disable-failoverモードを使用することにより自動切り替え/切り離しを有効、または無効にすることが可能です。

プライマリサーバに切り替え

プライマリサーバの保守を行うために、プライマリサーバとスタンバイサーバのどちらかでmc_ctlコマンドをswitchモードで実行します。

例）

```
$ mc_ctl switch -M /mcdir/inst1
```

切り替えが完了した時点で、新プライマリサーバのpostgresql.confファイルのsynchronous_standby_namesパラメータが以下のようにコメントアウトされます。

例）

```
#synchronous_standby_names = 'primary'
```

新スタンバイサーバの保守作業

1. Mirroring Controllerを停止します。

   新スタンバイサーバ（切り替わる前のプライマリサーバ）において、mc_ctlコマンドをstopモードで実行します。

   systemdを使用してMirroring Controllerの自動起動・停止の設定を実施している場合は、mc_ctlコマンドではなく、systemctlコマンドを使用してください。詳細は、“2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定”を参照してください。

   例）

   ```
   $ mc_ctl stop -M /mcdir/inst1
   ```

2. 停止が完了したことを確認します。

   新スタンバイサーバにおいてデータベースサーバのOSの起動・停止に合わせて、多重化したインスタンスおよびMirroring Controllerを自動的に起動・停止する設定にしていた場合は、ここで自動的に起動・停止する設定を解除します。

   ### 参照

   OSの起動・停止に合わせて、多重化したインスタンスおよびMirroring Controllerを自動的に起動・停止する設定の詳細については“2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定”を参照してください。

   OSのスーパーユーザーで、systemctlコマンドを使用して自動起動・停止の無効化を行います。

   例）
   自動起動・停止するユニットを“mcoi_inst1.service”とした場合の例を以下に示します。

   ```
   # systemctl disable mcoi_inst1.service
   ```

3. 停止中の新スタンバイサーバにおいて、保守作業を実施します。

4. 新スタンバイサーバに新プライマリサーバのインスタンスの複製を作成します。

   pg_basebackupコマンドを実行して、新スタンバイサーバのデータを新プライマリサーバと同期して作成します。

   例）

   ```
   $ pg_basebackup -D /database/inst1 -X fetch --waldir=/transaction/inst1 --progress --verbose -R --
   dbname='application_name=スタンバイサーバ名' -h プライマリサーバのホスト名 -p プライマリサーバのポート番号
   ```

参照

新スタンバイサーバに新プライマリサーバのインスタンスを複製する手順は、新スタンバイサーバをセットアップする手順と同じです。"2.5.2 スタンバイサーバのインスタンスの作成・設定・登録"を参照して、リカバリを実施してください。

5. 多重化したインスタンスおよびMirroring Controllerを自動的に起動・停止する設定を確認します。

   手順2において、OSの起動・停止に合わせて、多重化したインスタンスおよびMirroring Controllerを自動的に起動・停止する設定を解除した場合は、ここで再設定します。自動的に起動・停止する必要が無い場合は、この手順は省略することができます。

   OSのスーパーユーザーで、systemctlコマンドを使用して自動起動・停止の有効化を行います。

   例）
   自動起動・停止するユニットを"mcoi_inst1.service"とした場合の例を以下に示します。

```
# systemctl enable mcoi_inst1.service
```

6. 保守作業が完了したあと、スタンバイサーバのpostgresql.confファイルを必要に応じて編集します。

   インスタンスの複製により、synchronous_standby_namesパラメータがプライマリサーバでの指定内容になるため、スタンバイサーバでの指定内容へ修正します。コメントアウトされている場合は、アンコメントしてください。

7. スタンバイサーバで、Mirroring Controllerを起動（組込み）します。

   自動切り替え/切り離しを有効化する場合

   インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで実行します。

   例）

```
$ mc_ctl start -M /mcdir/inst1
```

   自動切り替え/切り離しを有効化しない場合

   インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで-Fオプションを指定して実行します。

   例）

```
$ mc_ctl start -M /mcdir/inst1 -F
```

ポイント

自動切り替え/切り離しはMirroring Controllerが起動したあとに、mc_ctlコマンドのenable-failoverモード、または、disable-failoverモードを使用することにより自動切り替え/切り離しを有効、または無効にすることが可能です。

プライマリサーバの切り戻し

プライマリサーバとスタンバイサーバをもとのサーバ構成に戻して、以前のプライマリサーバで主たる業務を行いたい場合に実施してください。手順の詳細は、"4.1.1.3 プライマリサーバの切り戻し"を参照してください。

注意

保守作業が完了したあと、すみやかにバックアップを取得してください。

3.8.2 保守停止

定期点検などですべてのサーバを停止する場合に行います。Mirroring Controllerが起動しているサーバ上で、mc_ctlコマンドをstopモードで実行してインスタンスとMirroring Controllerを停止してください。

systemdを使用してMirroring Controllerの自動起動・停止の設定を実施している場合は、mc_ctlコマンドではなく、systemctlコマンドを使用してください。詳細は、"2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定"を参照してください。

その後、Mirroring Controller裁定プロセスが起動しているサーバ上でmc_arbコマンドをstopモードで実行し、Mirroring Controller裁定プロセスを停止してください。

Mirroring Controllerの停止

例）

```
$ mc_ctl stop -M /mcdir/inst1 -a
```

Mirroring Controller裁定プロセスの停止

例）

```
$ mc_arb stop -M /mcarb_dir/arbiter1
```

3.8.3 裁定サーバの保守

裁定サーバの保守は、データベースサーバの業務に影響することなく行うことができます。

裁定サーバの保守は、以下の手順で行います。

1. mc_arbコマンドをstopモードで実行し、Mirroring Controller裁定プロセスを強制停止します。

   例）

   ```
   $ mc_arb stop -M /mcarb_dir/arbiter1 -e
   ```

2. 保守作業を実施します。
3. mc_arbコマンドをstartモードで実行し、Mirroring Controller裁定プロセスを再起動します。

   例）

   ```
   $ mc_arb start -M /mcarb_dir/arbiter1
   ```

4. mc_arbコマンドをstatusモードで実行し、データベースサーバと接続していることを確認します。

   mc_arbコマンドを実行した場合の状態表示例を以下に示します。

   例）

   ```
   $ mc_arb status -M /mcarb_dir/arbiter1

   server_id    host            status
   ---------------------------------------
   server1      192.0.3.100     online
   server2      192.0.3.110     online
   ```

5. 手順4で実行したmc_arbコマンドの表示結果を確認します。

   確認する項目

   “status”の項目が、2行とも“online”になっていることを確認してください。

3.9 運用の変更

運用の変更には、以下の場合があります。

- スタンバイサーバの停止を伴う変更
- 単体のサーバでの運用からデータベース多重化運用への変更
- データベース多重化運用から単体のサーバでの運用への変更

- 裁定サーバを利用して自動縮退を行うデータベース多重化運用への変更
- パラメータの変更
- データベース多重化運用でのアンインストール

3.9.1 スタンバイサーバの停止を伴う変更

スタンバイサーバ停止時の操作

スタンバイサーバを停止した状態でプライマリサーバのインスタンスに対して保守作業を行う場合は、プライマリサーバのMirroring Controllerを停止し、プライマリサーバでpostgresql.confファイルのsynchronous_standby_namesパラメータをコメントアウトしたあと、pg_ctlコマンドをreloadモードで実行してから操作してください。

この操作を行わない場合、プライマリサーバで行うインスタンスへの操作が待ち状態になります。

参照

pg_ctlコマンドの詳細は、“リファレンス”を参照してください。

スタンバイサーバの停止時間

postgresql.confファイルのsynchronous_standby_namesパラメータを指定した状態で、スタンバイサーバのインスタンスを停止した場合、以下を考慮してください。

- postgresql.confファイルのwal_sender_timeoutパラメータ

  本パラメータに設定したタイムアウト時間を越えてスタンバイサーバを停止した場合、トランザクションログの受信に失敗した内容のエラーがプライマリサーバのシステムログに出力され、スタンバイサーバに転送するべきトランザクションログが蓄積されます。

- postgresql.confファイルのwal_keep_sizeパラメータ

  スタンバイサーバの停止中に本パラメータの設定値を超えるトランザクションログが発生した場合、トランザクションログが削除されることがあります。

  また、本パラメータの設定には、データベース多重化運用を安定化させるために考慮することがあります。詳細は、“2.11.1 データベース多重化運用を安定させるためのチューニング”を参照してください。

注意

保守作業が完了した後のスタンバイサーバの起動時点において、スタンバイサーバへの転送が保留されているトランザクションログが損失している場合に、スタンバイサーバの再構築が必要です。

“4.1.1.1.3 異常原因の特定とリカバリ”から“4.1.1.2 スタンバイサーバの組込み”までの対処を実施してください。

3.9.2 単体のサーバでの運用からデータベース多重化運用への変更

システムの高信頼化・負荷分散を目的として、単体のサーバで行っていた運用をデータベース多重化運用に切り替える場合の手順を示します。

この手順は、“第2章 データベース多重化運用のセットアップ”の章に記載されているセットアップに相当する手順を実施することで行います。

注意

データ格納先ディレクトリ名がASCII文字で構成されていない場合

アプリケーション業務を停止したあと、ASCII文字で構成されるディレクトリに以下の手順で移行してください。

1. プライマリサーバのデータベースのインスタンスを停止します。
2. データ格納先ディレクトリ名をASCII文字で構成されるディレクトリ名に変更します。

参照

格納データを暗号化している場合は、“運用ガイド”の“データベース多重化運用”を参照して、プライマリサーバとスタンバイサーバで暗号化のためのセットアップを実施してください。

1. 裁定サーバへのインストール

   本手順は、裁定サーバを利用して自動縮退を行う運用を行う場合にのみ実施してください。

   Mirroring Controller裁定プロセスを起動するサーバに対して、サーバアシスタント機能をインストールします。

   インストール手順については、“導入ガイド（サーバアシスタント編）”の“インストール”を参照してください。

2. スタンバイサーバへのインストール

   スタンバイサーバとして起動するサーバに対して、FUJITSU Enterprise Postgresをインストールします。

   インストール手順については、“導入ガイド（サーバ編）”の“インストール”を参照してください。

   データ格納先ディレクトリは、ASCII文字で指定してください。

3. アプリケーション業務の停止

   プライマリサーバに接続するアプリケーション業務を停止します。

4. プライマリサーバの設定変更

   スタンバイサーバとして起動するサーバからの接続を許可するため、プライマリサーバ上で“2.4.2 プライマリサーバのインスタンスの作成・設定・登録”の手順2以降の設定を行います。

5. 裁定サーバのセットアップ

   “2.3 裁定サーバのセットアップ”の手順を実施します。

   本手順は、裁定サーバを利用して自動縮退を行う運用を行う場合にのみ実施してください。

6. プライマリサーバのデータベース多重化運用のセットアップ

   “2.4.1 プライマリサーバのデータベース多重化運用のセットアップ”の手順を実施します。

7. スタンバイサーバのデータベース多重化運用のセットアップ

   “2.5.1 スタンバイサーバのデータベース多重化運用のセットアップ”の手順を実施します。

8. スタンバイサーバのインスタンスの作成とインスタンスの起動

   “2.5.2 スタンバイサーバのインスタンスの作成・設定・登録”の手順を実施して、スタンバイサーバのインスタンスの作成とインスタンスの起動を実施します。

上記の手順が完了したあと、“第2章 データベース多重化運用のセットアップ”の章に記載されている残りの記事を参照して、必要な設定や操作を実施するようにしてください。

3.9.3 データベース多重化運用から単体のサーバでの運用への変更

データベース多重化運用をやめて、単体のサーバで行う運用に変更する場合の手順を示します。

運用の変更は、データベースサーバでの操作と裁定サーバでの操作が必要です。

裁定サーバでの操作は、裁定サーバを利用して自動縮退を行う運用を行っている場合にのみ、実施してください。

データベースサーバでの操作

1. インスタンスを停止するサーバの決定と切り替え

   データベース多重化運用の対象から外して、インスタンスを停止するサーバを決定します。

   インスタンスを停止するサーバがプライマリサーバである場合は、mc_ctlコマンドをswitchモードで実行してスタンバイサーバをプライマリサーバに切り替えてください。

   切り替えが完了したあとのスタンバイサーバが、インスタンスを停止するサーバとなります。

インスタンスを停止するサーバがスタンバイサーバである場合は、切り替え操作は必要ありません。

例）

```
$ mc_ctl switch -M /mcdir/inst1
```

2. Mirroring Controllerおよびインスタンスの停止とファイル資源の削除

   手順1で決定したサーバにおいて、mc_ctlコマンドをstopモードで実行して、Mirroring Controllerとインスタンスを停止します。

   systemdを使用してMirroring Controllerの自動起動・停止の設定を実施している場合は、mc_ctlコマンドではなく、systemctlコマンドを使用してください。詳細は、"2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定"を参照してください。

   例）

```
$ mc_ctl stop -M /mcdir/inst1
```

   そのあと、以下のファイル資源を削除してください。

   - データ格納先ディレクトリ
   - Mirroring Controller管理ディレクトリ

   例）

```
$ rm -rf /database/inst1
$ rm -rf /mcdir/inst1
```

参照

データを安全に削除するには、"運用ガイド"の"セキュリティに関する注意事項"を参照してください。

3. アプリケーション業務の停止

   プライマリサーバに接続するアプリケーション業務を停止します。

4. プライマリサーバのMirroring Controllerおよびインスタンスの停止

   プライマリサーバでmc_ctlコマンドをstopモードで実行します。

   systemdを使用してMirroring Controllerの自動起動・停止の設定を実施している場合は、mc_ctlコマンドではなく、systemctlコマンドを使用してください。詳細は、"2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定"を参照してください。

   例）

```
$ mc_ctl stop -M /mcdir/inst1
```

5. プライマリサーバのインスタンスに対して設定されているデータベース多重化運用の設定を削除します。

   postgresql.confファイルのパラメータをデータベース多重化運用として設定する前の値に戻してください。

   Mirroring Controller管理ディレクトリのファイル資源は削除してください。

   バックアップ運用を行っている場合は、以下の資源を削除してください。

   - データベース多重化運用中に取得したMirroring Controller管理ディレクトリのバックアップデータ
   - データベース多重化運用中に取得したインスタンスのバックアップデータ

   また、postgresql.auto.confファイルにprimary_conninfoパラメータが設定されている場合は、ALTER SYSTEM RESET文を実行して、設定を削除してください。

   例）
   psqlコマンドを使用した場合の実行例を以下に示します。

```
postgres=# ALTER SYSTEM RESET primary_conninfo;
```

   これらの対処を行ったあと、単体運用を開始するときには、バックアップデータを採取するようにしてください。

参照

- データを安全に削除するには、“運用ガイド”の“セキュリティに関する注意事項”を参照してください。
- バックアップ運用の詳細は、“2.14 バックアップ運用”を参照してください。
- postgresql.confファイルのパラメータの詳細は、“付録A パラメータ”を参照してください。

ポイント

上記の手順において、単体のサーバとなるプライマリサーバのpostgresql.confファイルの変更手順が、設定ファイルの再読み込みで可能な場合には、アプリケーション業務を停止することなく運用を変更することができます。

その場合、プライマリサーバの停止は、mc_ctlコマンドのstopモードに--mc-onlyオプションを指定して、Mirroring Controllerのみを停止してください。

裁定サーバでの操作

1. mc_arbコマンドをstopモードで実行して、Mirroring Controller裁定プロセスを停止します。

   例）

```
$ mc_arb stop -M /mcarb_dir/arbiter1
```

2. Mirroring Controller裁定プロセスの管理ディレクトリを削除します。

   例）

```
$ rm -rf /mcarb_dir/arbiter1
```

3.9.4 裁定サーバを利用して自動縮退を行うデータベース多重化運用への変更

データベースサーバのMirroring Controllerのみの運用から、裁定サーバを利用して自動縮退を行うデータベース多重化運用に変更する場合の手順を示します。

運用の変更は、データベースサーバと裁定サーバでの操作が必要です。

裁定サーバでの操作

1. 裁定サーバのセットアップを行います。

   裁定サーバのセットアップ手順については、“2.3 裁定サーバのセットアップ”を参照してください。

データベースサーバでの操作

1. Mirroring Controllerが起動しているサーバ上で、mc_ctlコマンドをstopモードで実行し、プライマリサーバとスタンバイサーバのMirroring Controllerを停止してください。

   例）

```
$ mc_ctl stop -M /mcdir/inst1 -a --mc-only
```

2. プライマリサーバとスタンバイサーバのnetwork.confを編集し、裁定サーバの情報を追加してください。

   詳細は、“A.3 ネットワーク定義ファイル”を参照してください。

   以下にプライマリサーバのnetwork.confの定義例を示します。

例）
プライマリサーバ、スタンバイサーバのサーバ識別子に“server1”および“server2”、ポート番号に“27540”“27541”、また
Mirroring Controller裁定プロセスのサーバ識別子に“arbiter”、ポート番号に“27541”を指定している場合の例を以下に示します。

```
server1 192.0.2.100,192.0.3.100 27540,27541 server
server2 192.0.2.110,192.0.3.110 27540,27541 server
arbiter 192.0.3.120 27541 arbiter
```

注意

- ポート番号は、プライマリサーバ、スタンバイサーバ、裁定サーバのどのサーバにおいても、他のソフトウェアと重ならないように設定してください。また、管理用ネットワークと裁定用ネットワークのセグメントを兼用して構成しないでください。
- サーバ種別が“server”の場合、IPアドレスまたはホスト名、およびポート番号を2つずつ以下の順序で指定する必要があります。
  - 管理用ネットワークとして使用するデータベースサーバ側のIPアドレスまたはホスト名
  - 裁定用ネットワークとして使用するデータベースサーバ側のIPアドレスまたはホスト名
  - 管理用ネットワークとして使用するデータベースサーバ側のポート番号
  - 裁定用ネットワークとして使用するデータベースサーバ側のポート番号
- サーバ種別が“arbiter”の場合、IPアドレスまたはホスト名はarbitration.confファイルのmy_addressパラメータに指定したIPアドレスまたはホスト名、ポート番号はarbitration.confファイルのportパラメータに指定したポート番号を指定してください。

3. プライマリサーバとスタンバイサーバのサーバ識別子.confを編集し、裁定サーバを利用して自動縮退を行う運用に必要なパラメータを追加してください。

   裁定サーバを利用して自動縮退を行う運用の場合に必要なパラメータについては、“A.4.1 データベースサーバのサーバ定義ファイル”を参照してください。

4. プライマリサーバとスタンバイサーバでmc_ctlコマンドをstartモードで実行し、Mirroring Controllerのプロセスを起動します。

   例）

```
$ mc_ctl start -M /mcdir/inst1 --mc-only
```

共通の操作

1. データベースサーバまたは裁定サーバから接続状態を確認します。

   接続状態の確認手順については、“2.8 接続状態の確認”を参照してください。

3.9.5 パラメータの変更

Mirroring Controllerのサーバ定義ファイルおよびネットワーク定義ファイルを編集する場合には、Mirroring Controllerを停止してから行ってください。

Mirroring Controllerプロセスでダウンや無応答が発生した場合に、Mirroring Controller監視プロセスにより自動で再起動が行われ、定義ファイルが再読み込みされます。そのため、定義ファイルの編集が行われていると、意図しない動作となります。

3.9.6 データベース多重化運用でのアンインストール

データベース多重化運用を行っているサーバでFUJITSU Enterprise Postgresをアンインストールする手順について説明します。

データベースサーバと裁定サーバでの操作が必要です。

裁定サーバでの操作は、裁定サーバを利用して自動縮退を行う運用を行っている場合にのみ、実施してください。

データベースサーバでの操作

1. 多重化したインスタンスとMirroring Controllerの停止

   多重化したインスタンスとMirroring Controllerを停止します。

停止方法については、“3.2 Mirroring Controllerの起動と停止”を参照してください。

2. FUJITSU Enterprise Postgresのアンインストール

   “導入ガイド(サーバ編)”の“アンインストール”を参照して、FUJITSU Enterprise Postgresをアンインストールします。

裁定サーバでの操作

“導入ガイド(サーバアシスタント編)”の“アンインストール”を参照して、サーバアシスタント機能をアンインストールします。

第4章 データベース多重化運用の異常時の対処

本章では、データベース多重化運用で異常が発生した場合の対処について説明します。

データベース多重化運用では、異常を検知すると自動的にスタンバイサーバの切り替え/切り離しを行って、プライマリサーバだけで縮退運転を行います。この場合、切り替え/切り離しを行ったスタンバイサーバに対して復旧作業が必要です。

このほかにも、以下の場合が考えられます。

- 自動切り替えに失敗した場合
- 自動切り離しに失敗した場合
- すべてのサーバまたはインスタンスが停止した場合

4.1 縮退運転になった場合の対処

縮退運転になった場合、その原因が切り替え(フェイルオーバまたはスイッチオーバ)によるものか、切り離しによるものかによって、復旧作業が異なります。

mc_ctlコマンドをstatusモードで実行するか、またはシステムログを参照するかして、切り替えと切り離しのどちらの原因で縮退運転になったのかを確認してください。

例)
mc_ctlコマンドをstatusモードで実行した場合の例を以下に示します。
切り替えが発生した場合は、“mirroring status”にswitched(切替済：切り替えが終わって縮退運転の状態)が表示されます。

```
$ mc_ctl status -M /mcdir/inst1
mirroring status
----------------
switched

  :
```

例)
mc_ctlコマンドをstatusモードで実行した場合の例を以下に示します。
切り離しが発生した場合は、“mirroring status”にnot-switchable（切替不可：切り離しが行われて切り替えができない状態)が表示されます。

```
$ mc_ctl status -M /mcdir/inst1
mirroring status
----------------
not-switchable

  :
```

注意

縮退運転の状態からデータベース多重化運用への復旧を行うとき、業務を継続しているサーバでMirroring Controllerが何らかの異常を検知している場合には、“4.1.3 縮退運転中の異常時の対処”を行ってから、データベース多重化運用への復旧を行ってください。

4.1.1 切り替え発生後の縮退運転における運用操作

切り替え発生後の縮退運転における運用操作について、説明します。

注意

- プライマリサーバの異常によって切り替えが発生してからスタンバイサーバの組込みを行うまで、データベースは多重化構成ではありません。できるだけ早急に異常原因を取り除き、組込みを行ってください。

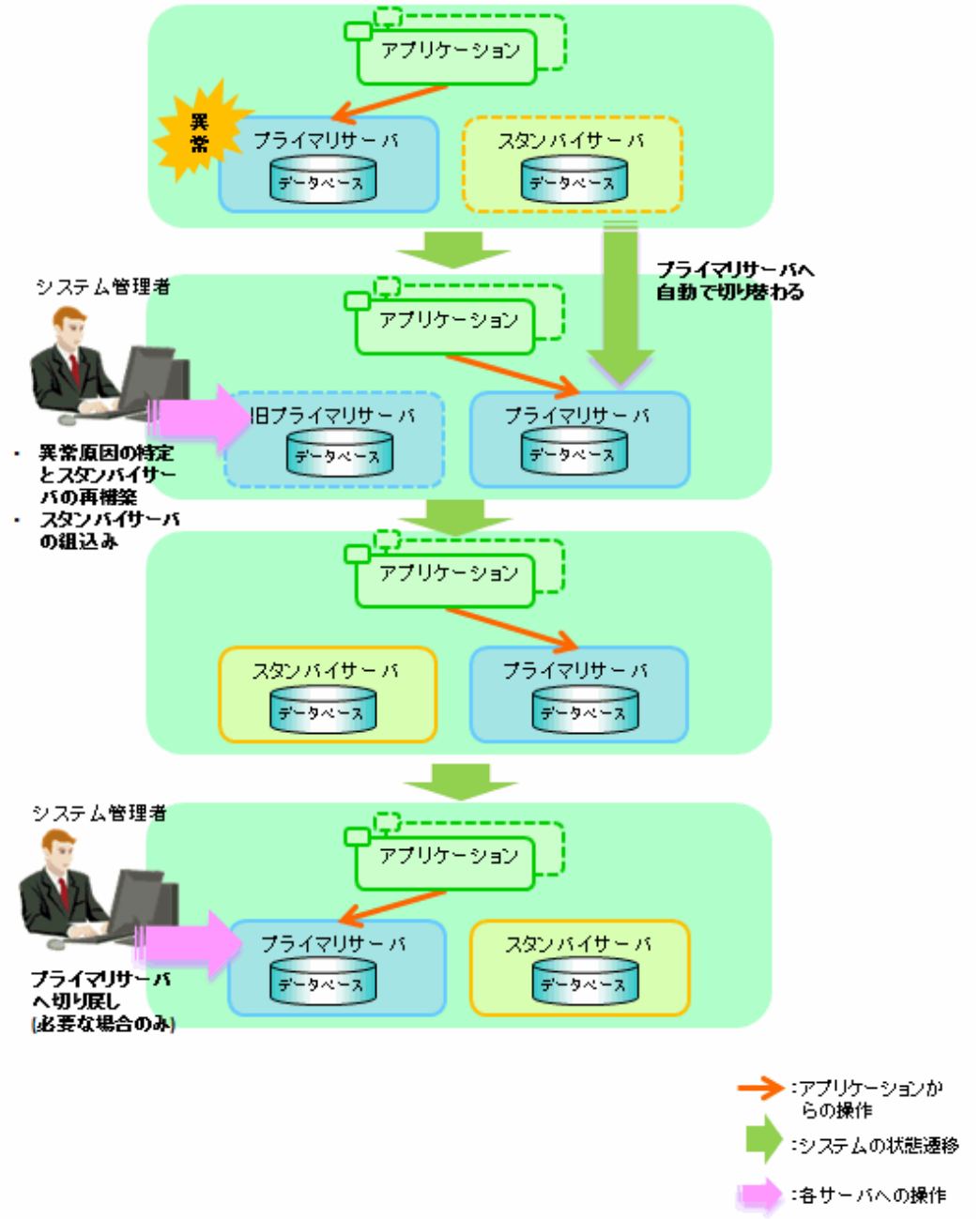
- スタンバイサーバで参照系の業務を行っていた場合、プライマリサーバに異常が発生してサーバが切り替わると、新しく切り替わったプライマリサーバに負荷が集中します。したがって、旧スタンバイサーバで行っている参照系の業務を一時停止して、旧プライマリサーバを新スタンバイサーバとして組込んだあと、新スタンバイサーバに対して参照系の業務を再開することを推奨します。
- 異常が発生した旧プライマリサーバを新スタンバイサーバとして組込む前に、新しく切り替わったプライマリサーバのインスタンスを停止した場合には、旧プライマリサーバのインスタンスから起動するとスプリットブレインが発生します。そのため、新プライマリサーバのインスタンスを起動してから、スタンバイサーバの再構築を行ってください。

切り替えが発生して縮退運転が行われている場合は、以下の運用操作を行って、スタンバイサーバを復旧してもとの状態に戻します。

- 異常原因の特定とスタンバイサーバの再構築
- スタンバイサーバの組込み
- プライマリサーバの切り戻し(必要な場合)

これらの運用操作の流れを以下の図に示します。

図4.1 運用操作の流れ



4.1.1.1 異常原因の特定とスタンバイサーバの再構築

以下の手順でリカバリを実施してください。

1. Mirroring Controllerの停止
2. Mirroring Controller管理ディレクトリのリカバリ
3. 異常原因の特定とリカバリ

4.1.1.1.1 Mirroring Controllerの停止

異常が発生した旧プライマリサーバに対して、mc_ctlコマンドをstopモードで実行します。

systemdを使用してMirroring Controllerの自動起動・停止の設定を実施している場合は、mc_ctlコマンドではなく、systemctlコマンドを使用してください。詳細は、“2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定”を参照してください。

例）

```
$ mc_ctl stop -M /mcdir/inst1
```

これにより、リカバリを実施する必要があるインスタンスも停止します。

注意

停止しない場合は、“運用ガイド”の“インスタンス停止失敗時の対処”を参照して、インスタンスを停止します。

そのあと、上記のコマンドに-eオプションを指定してMirroring Controllerを強制停止してください。

4.1.1.1.2 Mirroring Controller管理ディレクトリのリカバリ

バックアップデータからMirroring Controller管理ディレクトリ配下のファイルをコピーして、リカバリを実施してください。

4.1.1.1.3 異常原因の特定とリカバリ

プライマリサーバとスタンバイサーバのシステムログを参照して異常の原因を特定したあと、復旧します。

スタンバイサーバの復旧方法として以下の2種類のコマンドが使用できます。復旧内容や状況に応じて選択してください。

- pg_basebackupコマンド

  プライマリサーバのインスタンスのすべての資源を複製します。

- pg_rewindコマンド

  新プライマリサーバの変更されたファイルのみを複製します。そのため、本コマンドを使用して新スタンバイサーバを組み込むことでリカバリ時間を短縮することができます。ただし、本コマンドを利用して旧プライマリサーバを新スタンバイサーバとして構築する場合、以下のいずれかの条件を満たす必要があります。

  a. インスタンス作成時にチェックサムが有効化されている。または、

  b. インスタンス起動時にpostgresql.confファイルのwal_log_hintsパラメータが有効化されている。

  full_page_writesも有効でなければなりませんが、これはデフォルトで有効です。

参照

- pg_basebackupコマンドの詳細については、“PostgreSQL Documentation”の“Reference”の“pg_basebackup”を参照してください。
- pg_rewindコマンドの詳細については、“PostgreSQL Documentation”の“Reference”の“pg_rewind”を参照してください。

pg_rewindコマンドを実行して、旧プライマリサーバのデータを新プライマリサーバと同期してリカバリする例を以下に示します。

1. 新プライマリサーバでの未適用更新トランザクションログの適用を待つ

   新プライマリサーバで以下のSQLを実行し、結果がfalseになるまで待ちます。

```
# select pg_is_in_recovery();
```

例）

```
$ psql -h 新プライマリサーバのホスト名 -p 新プライマリサーバのポート番号 -d データベース名 -c "select
pg_is_in_recovery();"
```

接続するデータベースは、どのデータベースでも問題ありません。

注意

新プライマリサーバの昇格直後にpg_rewindを実行する場合、手順1および手順2の処理が必要になります。新プライマリサーバで更新系のSQLを実行できるようになり、チェックポイント処理が昇格後に実行されている場合は、手順1および手順2の処理は不要です。

2. タイムラインIDの更新

   チェックポイント処理を実行して、タイムラインIDを更新します。

```
$ psql -h 新プライマリサーバのホスト名 -p 新プライマリサーバのポート番号 -d データベース名 -c "checkpoint;"
```

   接続するデータベースは、どのデータベースでも問題ありません。

3. 旧プライマリサーバ(新スタンバイサーバ)に新プライマリサーバのインスタンスの複製を作成

   pg_rewindコマンドを実行して、新スタンバイサーバのデータを新プライマリサーバと同期して作成します。

   例）

```
$ pg_rewind -D /database/inst1 -R --source-server='user=ユーザー名 host=新プライマリサーバのホスト名 port=新プライマ
リサーバのポート番号 dbname=データベース名 application_name=新スタンバイサーバ名'
```

注意

- pg_rewindコマンドには、-Rオプションを指定して実行し、standby.signalファイルを作成してください。standby.signalファイルを作成しない場合、Mirroring Controllerはスタンバイサーバとして起動できません。
- プライマリサーバへの接続がパスワード認証を必要とする方式の場合、自動で認証が行われるようにしておく必要があります。pg_rewindコマンドの-Rオプションを指定し、--source-serverオプションにpasswordパラメータを指定すると、pg_rewindコマンドによりpostgresql.auto.confファイルのprimary_conninfoパラメータにパスワードが設定されて自動的に接続できるようになります。

  postgresql.auto.confファイルのprimary_conninfoパラメータにパスワードを設定しない場合は、インスタンス管理者ユーザーのホームディレクトリに.pgpassファイルを作成してreplicationデータベースに対するパスワードを設定してください。
- host、portおよびapplication_name以外に設定が必要な接続文字列がある場合は、primary_conninfoパラメータの設定に含めてください。
- primary_conninfoパラメータは、postgresql.confファイルには設定せず、pg_rewindコマンドを使用してpostgresql.auto.confファイルのみに設定してください。

4. 旧プライマリサーバ(新スタンバイサーバ)のpostgresql.confファイルにパラメータを設定

   スタンバイサーバ側に必要なパラメータをpostgresql.confファイルに設定します。

   postgresql.confファイルに設定するパラメータは、“表2.4 設定するパラメータ”を参照して設定してください。

参照

- standby.signalファイルの詳細については、“PostgreSQL Documentation”の“Hot Standby”を参照してください。
- primary_conninfoの詳細については、“PostgreSQL Documentation”の“Setting Up a Standby Server”を参照してください。

注意

新プライマリサーバは昇格によって新たなタイムラインに分岐されているため、旧プライマリサーバ(新スタンバイサーバ)が新プライマリサーバに追随するためには、recovery_target_timelineパラメータに'latest'を指定する必要があります。

4.1.1.2 スタンバイサーバの組込み

復旧した旧プライマリサーバをスタンバイサーバとして起動することをスタンバイサーバの組込みと呼びます。

旧プライマリサーバに対して、Mirroring Controllerとインスタンスを起動します。

自動切り替え/切り離しを有効化する場合

インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで実行します。

例)

```
$ mc_ctl start -M /mcdir/inst1
```

自動切り替え/切り離しを有効化しない場合

インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで-Fオプションを指定して実行します。

例)

```
$ mc_ctl start -M /mcdir/inst1 -F
```

ポイント

自動切り替え/切り離しはMirroring Controllerが起動したあとに、mc_ctlコマンドのenable-failoverモード、または、disable-failoverモードを使用することにより自動切り替え/切り離しを有効、または無効にすることが可能です。

4.1.1.3 プライマリサーバの切り戻し

スタンバイサーバを組込んだあとにプライマリサーバとスタンバイサーバをもとの構成に戻したい場合は、プライマリサーバの切り戻しを行います。

以前のプライマリサーバで主な業務を行いたい場合に実施してください。

以下の手順で行います。

1. プライマリサーバの切り戻し

   プライマリサーバとスタンバイサーバのどちらかでmc_ctlコマンドをswitchモードで実行します。

   例)

   ```
   $ mc_ctl switch -M /mcdir/inst1
   ```

   mc_ctlコマンドをswitchモードで実行したあとは、以下の状態になります。

   例)

   ```
   $ mc_ctl status -M /mcdir/inst1
   mirroring status
   ----------------
   switched
   server_id  host_role                  host         host_status   db_proc_status        disk_status
   ------------------------------------------------------------------------------------------------------
   server1    primary                    192.0.2.100   normal        abnormal(postmaster)  normal
   server2    none(inactivated primary)  192.0.2.110   normal        abnormal(postmaster)  normal
   ```

2. 旧プライマリサーバの停止

旧プライマリサーバでmc_ctlコマンドをstopモードで実行して、Mirroring Controllerとインスタンスを停止します。

systemdを使用してMirroring Controllerの自動起動・停止の設定を実施している場合は、mc_ctlコマンドではなく、systemctlコマンドを使用してください。詳細は、"2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定"を参照してください。

例)

```
$ mc_ctl stop -M /mcdir/inst1
```

3. 旧プライマリサーバ(新スタンバイサーバ)に新プライマリサーバのインスタンスの複製を作成

pg_basebackupコマンドを実行して、新スタンバイサーバのデータを新プライマリサーバと同期して作成します。

例)

```
$ pg_basebackup -D /database/inst1 -X fetch --waldir=/transaction/inst1 --progress --verbose -R --
dbname='application_name=スタンバイサーバ名' -h プライマリサーバのホスト名 -p プライマリサーバのポート番号
```

参照

新スタンバイサーバに新プライマリサーバのインスタンスを複製する手順は、新スタンバイサーバをセットアップする手順と同じです。

"2.5.2 スタンバイサーバのインスタンスの作成・設定・登録"を参照して、リカバリを実施してください。

4. スタンバイサーバの組込み

スタンバイサーバでMirroring Controllerとインスタンスを起動します。

自動切り替え/切り離しを有効化する場合

インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで実行します。

例)

```
$ mc_ctl start -M /mcdir/inst1
```

自動切り替え/切り離しを有効化しない場合

インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで-Fオプションを指定して実行します。

例)

```
$ mc_ctl start -M /mcdir/inst1 -F
```

ポイント

自動切り替え/切り離しはMirroring Controllerが起動したあとに、mc_ctlコマンドのenable-failoverモード、または、disable-failoverモードを使用することにより自動切り替え/切り離しを有効、または無効にすることが可能です。

4.1.2 切り離し発生後の縮退運転における運用操作

切り離し発生後の縮退運転における運用操作について、説明します。

注意

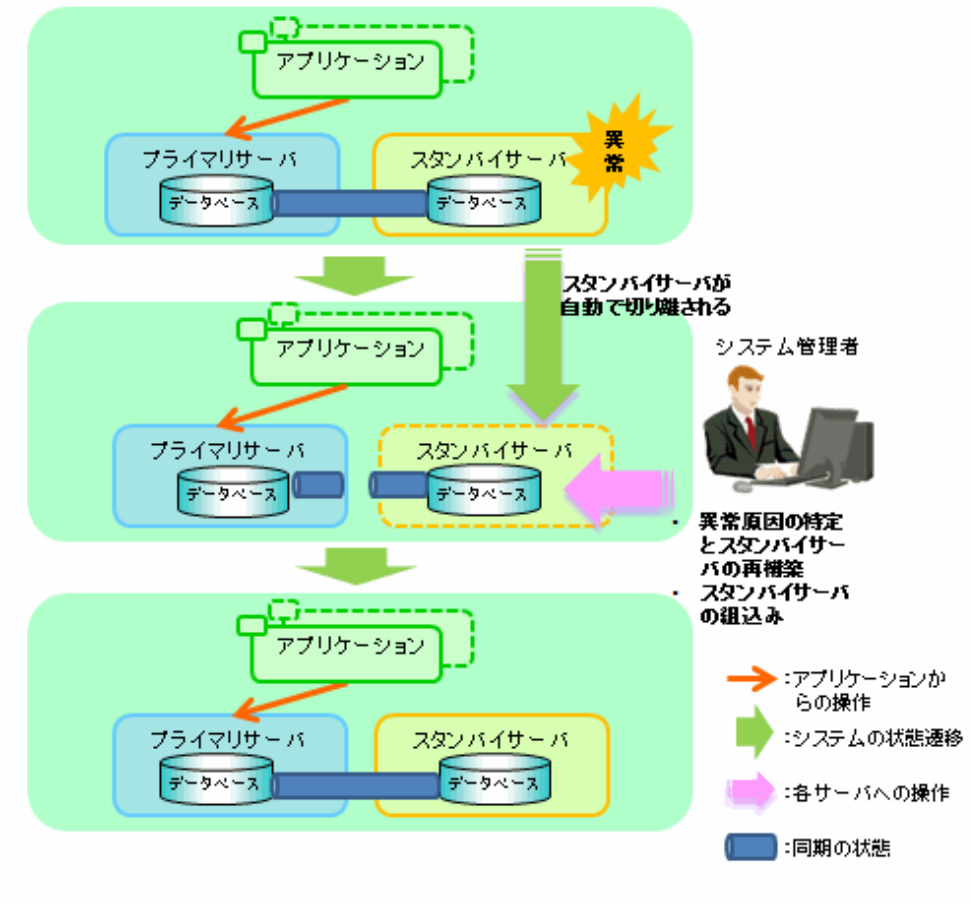
スタンバイサーバの異常によって切り離しが発生してからスタンバイサーバの組込みを行うまで、データベースは多重化構成ではありません。できるだけ早急に異常原因を取り除き、スタンバイサーバの組込みを行ってください。

切り離しが発生して縮退運転が行われている場合は、以下の運用操作を行ってスタンバイサーバを復旧してもとの状態に戻します。

- 異常原因の特定とスタンバイサーバの再構築
- スタンバイサーバの組込み

これらの運用操作の流れを以下の図に示します。

図4.2 運用操作の流れ



4.1.2.1 異常原因の特定とスタンバイサーバの再構築

以下の手順でリカバリを実施してください。

1. Mirroring Controllerの停止
2. Mirroring Controller管理ディレクトリのリカバリ
3. 異常原因の特定とリカバリ

4.1.2.1.1 Mirroring Controllerの停止

異常が発生したスタンバイサーバに対して、mc_ctlコマンドをstopモードで実行します。

systemdを使用してMirroring Controllerの自動起動・停止の設定を実施している場合は、mc_ctlコマンドではなく、systemctlコマンドを使用してください。詳細は、“2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定”を参照してください。

例）

```
$ mc_ctl stop -M /mcdir/inst1
```

これにより、リカバリを実施する必要があるインスタンスも停止します。

注意

停止しない場合は、“運用ガイド”の“インスタンス停止失敗時の対処”を参照して、インスタンスを停止します。

そのあと、上記のコマンドに-eオプションを指定してMirroring Controllerを強制停止してください。

4.1.2.1.2 Mirroring Controller管理ディレクトリのリカバリ

バックアップデータからMirroring Controller管理ディレクトリ配下のファイルをコピーして、リカバリを実施します。

4.1.2.1.3 異常原因の特定とリカバリ

プライマリサーバとスタンバイサーバのシステムログを参照して異常の原因を特定したあと、復旧します。

pg_basebackupコマンドを実行して、プライマリサーバのデータをスタンバイサーバと同期してリカバリします。

例）

```
$ pg_basebackup -D /database/inst1 -X fetch --waldir=/transaction/inst1 --progress --verbose -R --
dbname='application_name=スタンバイサーバ名' -h プライマリサーバのホスト名 -p プライマリサーバのポート番号
```

参照

このリカバリ手順は、スタンバイサーバをセットアップする手順と同じです。

“2.5.2 スタンバイサーバのインスタンスの作成・設定・登録”を参照して、リカバリを実施してください。

4.1.2.2 スタンバイサーバの組込み

スタンバイサーバのMirroring Controllerとインスタンスを起動して、スタンバイサーバを組込みます。

自動切り替え/切り離しを有効化する場合

インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで実行します。

例）

```
$ mc_ctl start -M /mcdir/inst1
```

自動切り替え/切り離しを有効化しない場合

インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで-Fオプションを指定して実行します。

例）

```
$ mc_ctl start -M /mcdir/inst1 -F
```

ポイント

自動切り替え/切り離しはMirroring Controllerが起動したあとに、mc_ctlコマンドのenable-failoverモード、または、disable-failoverモードを使用することにより自動切り替え/切り離しを有効、または無効にすることが可能です。

4.1.3 縮退運転中の異常時の対処

切り替えまたは切り離しによって縮退運転になった後に、業務を継続しているサーバで異常が発生した場合の対処方法について、説明します。

バックアップデータからの復旧が必要な場合

ディスク障害やユーザーの誤操作によるデータ破壊など、バックアップデータからデータベースの復旧が必要な場合には、以下を参照して、データベース多重化運用への復旧を行ってください。

- すべてのデータベースサーバまたはインスタンスが停止した場合の対処
- ユーザーの誤操作からのリカバリ

一時的な異常の場合

サーバの高負荷状態やシステムリソースの不足などの一時的な異常が発生した場合には、異常原因を取り除いてからMirroring Controllerを再起動し、以下を参照して、データベース多重化運用への復旧を行ってください。

- 切り替え発生後の縮退運転における運用操作
- 切り離し発生後の縮退運転における運用操作

参照

Mirroring Controllerの再起動については、"3.2 Mirroring Controllerの起動と停止"を参照して、Mirroring Controllerの停止および起動を行ってください。

4.2 自動切り替えに失敗した場合の対処

一時的なシステムリソース不足の発生のようにシステムの動作が不安定になった場合は、Mirroring Controllerが自動切り替えに失敗することがあります。

以下の方法により手動で切り替えを実施してください。

- "3.4 プライマリサーバの手動切り替え"の手順を参照して、手動で切り替えを実行する
- スタンバイサーバで、mc_ctlコマンドをswitchモードで--forceオプションを指定して実行し、強制的に切り替えを行う

  例）

```
$ mc_ctl switch -M /mcdir/inst1 --force
```

ポイント

- データベースサーバ間の通信ができない状況であっても、スタンバイサーバでmc_ctlコマンドのswitchモードに--forceオプションを指定して実行することで、プライマリサーバのフェンシングを行い、強制的に切り替え処理を行うことが可能です。
- 以下の場合にプライマリサーバのフェンシングを行わないため、事前にプライマリサーバのデータベースのMirroring Controllerとインスタンスを停止してください。
  - 強制切り替え実行時に--no-fencingオプションを指定
  - サーバ識別子.confファイルのheartbeat_error_actionの設定がmessageの場合に、フェンシングコマンドを利用する設定を行っていない(サーバ識別子.confファイルのfencing_commandパラメータを省略)
  - サーバ識別子.confファイルのheartbeat_error_actionの設定がfallbackの場合

参照

データベース多重化運用への復旧

データベース多重化運用へ復旧するには、"4.1.1.2 スタンバイサーバの組込み"および"4.1.1.3 プライマリサーバの切り戻し"を参照してください。

4.3 自動切り離しに失敗した場合の対処

メモリやディスクの空き領域などのシステムリソースが不足した場合などのようにシステムの動作が不安定になった場合は、Mirroring Controllerが自動で切り離しを実施できないことがあります。

以下の方法により手動で切り離しを実施してください。

- “3.5 スタンバイサーバの手動切り離し”を参照して、手動で切り離しを実行する
- プライマリサーバで、mc_ctlコマンドをdetachモードで実行し、強制的に切り離しを行う

  例）

```
$ mc_ctl detach -M /mcdir/inst1
```

ポイント

- データベースサーバ間での通信ができない状況であっても、プライマリサーバでmc_ctlコマンドのdetachモードを実行することで、スタンバイサーバのフェンシングを行い、強制的に切り離し処理を行うことが可能です。
- 以下の場合にスタンバイサーバのフェンシングを行わないため、事前にスタンバイサーバのデータベースのMirroring Controllerとインスタンスを停止してください。
  - 強制切り離し実行時に--no-fencingオプションを指定
  - サーバ識別子.confファイルのheartbeat_error_actionの設定がmessageの場合に、フェンシングコマンドを利用する設定を行っていない(サーバ識別子.confファイルのfencing_commandパラメータを省略)
  - サーバ識別子.confファイルのheartbeat_error_actionの設定がfallbackの場合

参照

データベース多重化運用への復旧

データベース多重化運用へ復旧するには、“4.1.2.2 スタンバイサーバの組込み”を参照してください。

4.4 すべてのデータベースサーバまたはインスタンスが停止した場合の対処

すべてのデータベースサーバまたはデータベースサーバ上のインスタンスが停止して、業務が継続できなくなった場合について、説明します。

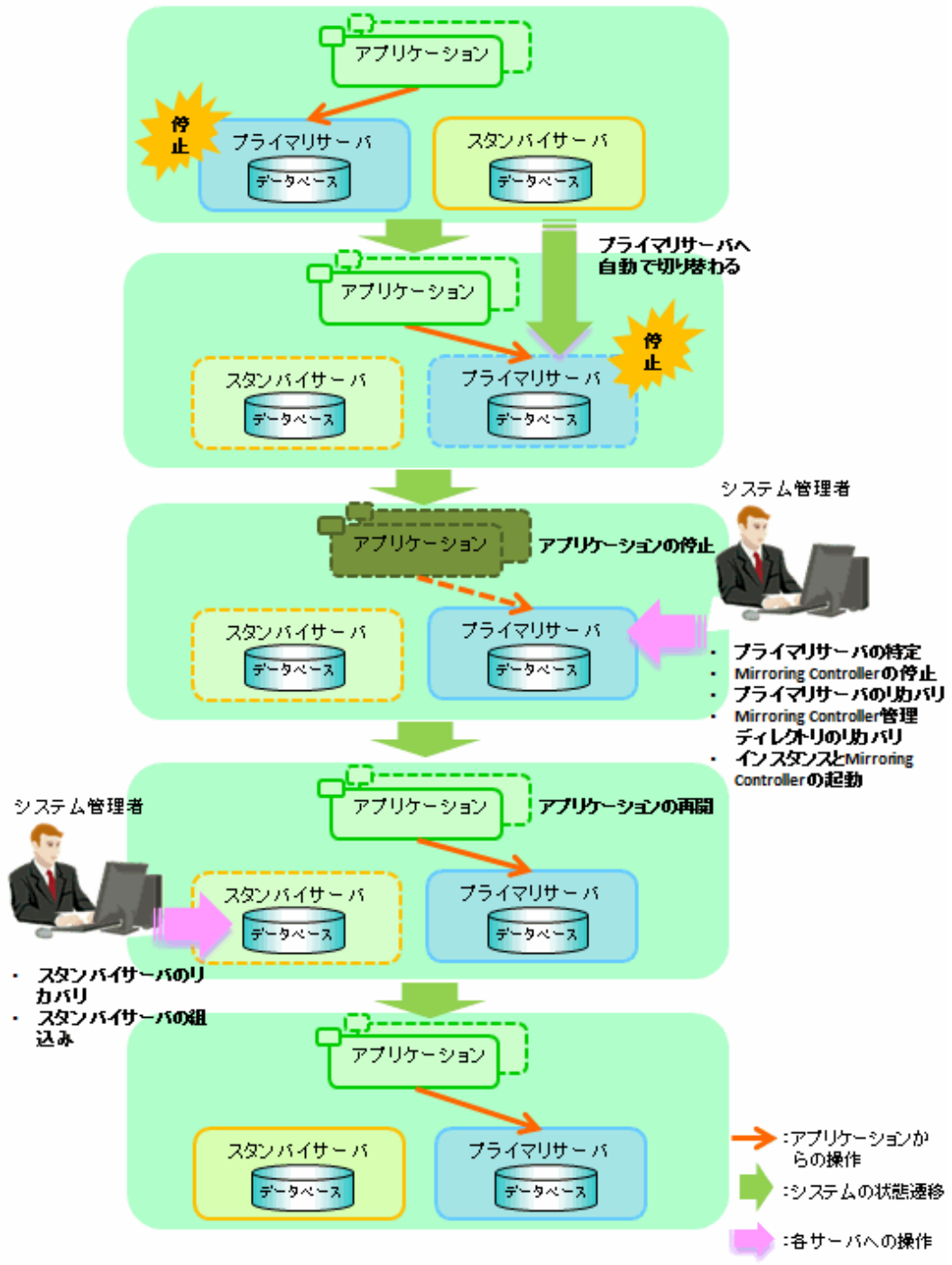
参照

データベース多重化運用への復旧

データベース多重化運用へ復旧するには、“4.1.1.2 スタンバイサーバの組込み”および“4.1.1.3 プライマリサーバの切り戻し”を参照してください。

これらの復旧操作の流れを以下の図に示します。

図4.3 復旧操作の流れ



以下の手順で実施してください。

1. アプリケーションの停止

   アプリケーションの実行を停止します。

2. プライマリサーバの特定

   以下のいずれかの方法で、停止する前に起動していたプライマリサーバを特定してください。

   ー 各サーバ上のシステムログを参照して、以下のメッセージが出力されているサーバを特定します。

   メッセージ

```
MirroringControllerOpen[30017]: LOG: プライマリサーバへの昇格処理が完了しました (MCA00062)
```

ー 各サーバ上でmc_ctlコマンドをstatusモードで実行して、none(inactivated primary)と表示されるサーバを検索します。

3. プライマリサーバのMirroring Controllerの停止

プライマリサーバに対して、mc_ctlコマンドをstopモードで実行します。

systemdを使用してMirroring Controllerの自動起動・停止の設定を実施している場合は、mc_ctlコマンドではなく、systemctlコマンドを使用してください。詳細は、"2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定"を参照してください。

例)

```
$ mc_ctl stop -M /mcdir/inst1
```

注意

Mirroring Controllerの強制停止

Mirroring Controllerが停止しない場合は、mc_ctlコマンドのstopモードに-eオプションを指定して実行します。

例)

```
$ mc_ctl stop -M /mcdir/inst1 -e
```

4. プライマリサーバのリカバリ

まず、"運用ガイド"の"異常時の対処"を参照して、異常原因の特定とリカバリを実施してください。

次に、バックアップデータをもとにしたpgx_rcvallコマンドによるリカバリの方法でプライマリサーバを復旧します。

"2.14.2 データベースのバックアップ運用"のpgx_dmpallコマンドによるバックアップ運用を行っている場合は、以下の手順でリカバリを実施してください。

a. プライマリサーバおよびスタンバイサーバの両方で以下の操作を行って、最新の日付を示すバックアップデータ、およびアーカイブログが存在するサーバを調べます。

- pgx_rcvallコマンドに-lオプションを付けて実行して、最新の日付を示すバックアップデータを特定します。
- 以下のように最新の日付を示すアーカイブログを特定します。

例)

```
$ ls -ltr <バックアップデータ格納先ディレクトリ>/*_wal
```

b. 最新のバックアップデータがスタンバイサーバに存在する場合は、バックアップ格納先ディレクトリごとプライマリサーバのバックアップ格納先ディレクトリに複写(注意1)して上書きします(注意2)。

c. 最新のアーカイブログおよびトランザクションログファイルがスタンバイサーバに存在する場合は、プライマリサーバのバックアップ格納先ディレクトリに複写(注意1)して上書きします(注意2)。

d. プライマリサーバ上で、pgx_rcvallコマンドをプライマリサーバのバックアップ格納先ディレクトリを指定して実行します。

注意

1. バックアップデータにシンボリックリンクが含まれることがあるため、シンボリックリンクが通常ファイルに変換されることがないようにtarコマンドなどを使用して複写してください。
2. バックアップ格納先ディレクトリのコピーを退避できる場合は、上書きしてしまわないように退避してください。

参照

pgx_rcvallコマンドの詳細については、"運用ガイド"の"異常時の対処"を参照してください。

5. Mirroring Controller管理ディレクトリのリカバリ

   バックアップデータからMirroring Controller管理ディレクトリ配下のファイルをコピーして、リカバリを実施します。

6. プライマリサーバのインスタンスとMirroring Controllerの起動

   自動切り替え/切り離しを有効化する場合

   インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで実行します。

   例）

```
$ mc_ctl start -M /mcdir/inst1
```

   自動切り替え/切り離しを有効化しない場合

   インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで-Fオプションを指定して実行します。

   例）

```
$ mc_ctl start -M /mcdir/inst1 -F
```

ポイント

自動切り替え/切り離しはMirroring Controllerが起動したあとに、mc_ctlコマンドのenable-failoverモード、または、disable-failoverモードを使用することにより自動切り替え/切り離しを有効、または無効にすることが可能です。

7. アプリケーションの再開

   アプリケーションの実行を再開します。

8. スタンバイサーバのMirroring Controllerの停止

   スタンバイサーバに対して、mc_ctlコマンドをstopモードで実行します。

   systemdを使用してMirroring Controllerの自動起動・停止の設定を実施している場合は、mc_ctlコマンドではなく、systemctlコマンドを使用してください。詳細は、“2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定”を参照してください。

   例）

```
$ mc_ctl stop -M /mcdir/inst1
```

9. スタンバイサーバのリカバリ

   “2.5.2 スタンバイサーバのインスタンスの作成・設定・登録”を参照して、プライマリサーバからスタンバイサーバを復旧（セットアップ）します。

10. スタンバイサーバの組込み

    スタンバイサーバ上で、インスタンスおよびMirroring Controllerを起動します。

    自動切り替え/切り離しを有効化する場合

    インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで実行します。

    例）

```
$ mc_ctl start -M /mcdir/inst1
```

    自動切り替え/切り離しを有効化しない場合

    インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで-Fオプションを指定して実行します。

    例）

```
$ mc_ctl start -M /mcdir/inst1 -F
```

ポイント

自動切り替え/切り離しはMirroring Controllerが起動したあとに、mc_ctlコマンドのenable-failoverモード、または、disable-failoverモードを使用することにより自動切り替え/切り離しを有効、または無効にすることが可能です。

4.5 ユーザーの誤操作からのリカバリ

ユーザーの誤操作によってデータ破壊が発生した場合に、インスタンスをリカバリする方法について、説明します。

たとえば、アプリケーションおよびコマンドによって、誤ったデータを意図しないデータに変更してしまった場合や削除してしまった場合などのように、誤操作によってデータを破壊してしまった場合は、プライマリサーバ上のデータをもとに戻して、スタンバイサーバと再同期する必要があります。

以下の手順で復旧します。

1. プライマリサーバの特定

   各サーバ上でmc_ctlコマンドをstatusモードで実行して、“primary”または“none(inactivated primary)”と表示されるサーバを検索します。

2. 誤操作の原因となったアプリケーションおよびコマンドの停止

   プライマリサーバで動作したアプリケーションおよびコマンドを停止します。これにより、誤ったデータによる影響を抑えます。

   また、スタンバイサーバで参照用のアプリケーションを実行している場合は、動作中のアプリケーションを停止します。

3. インスタンスおよびMirroring Controllerの停止

   プライマリサーバおよびスタンバイサーバのインスタンスとMirroring Controllerを停止します。

   例）

```
$ mc_ctl stop -a -M /mcdir/inst1
```

4. プライマリサーバのデータベースの復旧

   誤操作を行った時間よりも過去のリストアポイントの時点まで、バックアップデータをもとにしたpgx_rcvallコマンドによるリカバリ方法でデータベースを復旧します。

参照

pgx_rcvallコマンドによるリストアポイントの時点までの復旧手順の詳細は、“運用ガイド”の“ユーザーの誤操作からのリカバリ”を参照して、インスタンス停止状態でのデータベースのリカバリ手順のみを行ってください。

5. インスタンスおよびMirroring Controllerの起動

   プライマリサーバでインスタンスおよびMirroring Controllerを起動します。

   自動切り替え/切り離しを有効化する場合

   インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで実行します。

   例）

```
$ mc_ctl start -M /mcdir/inst1
```

   自動切り替え/切り離しを有効化しない場合

   インスタンス管理者ユーザーで、mc_ctlコマンドをstartモードで-Fオプションを指定して実行します。

   例）

```
$ mc_ctl start -M /mcdir/inst1 -F
```

ポイント

自動切り替え/切り離しはMirroring Controllerが起動したあとに、mc_ctlコマンドのenable-failoverモード、または、disable-failoverモードを使用することにより自動切り替え/切り離しを有効、または無効にすることが可能です。

6. 新スタンバイサーバの構築

   "2.5 スタンバイサーバのセットアップ"を参照して、プライマリサーバからスタンバイサーバを構築(セットアップ)します。

第5章 WebAdminを使用したMirroring Controllerの管理

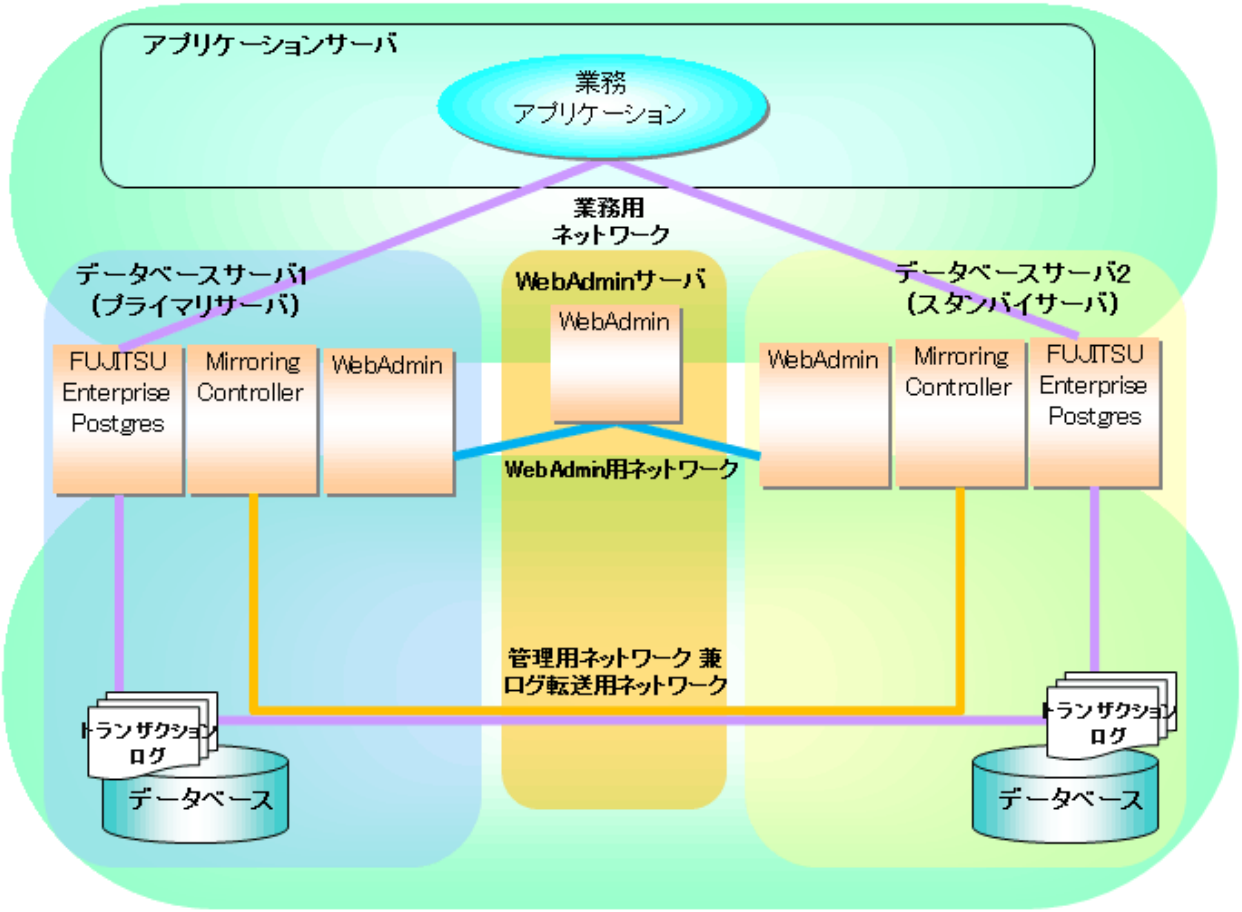
WebAdminを使って、ストリーミングレプリケーションクラスタでMirroring Controllerをセットアップおよび管理する方法を説明します。

エラーが起こった際に、Mirroring Controllerで、ストリーミングレプリケーションクラスタの監視と自動切り替え/切り離しを行うか、または同期レプリケーションの接続を切断することができます。

WebAdminを使って、既存のレプリケーションクラスタにMirroring Controllerをセットアップすることができます。同期スタンバイインスタンスまたは非同期スタンバイインスタンスのどちらにもセットアップが可能です。

WebAdminを使って、データベース多重化システムを構成したときのシステム構成を、以下に示します。

図5.1 WebAdminを利用したデータベース多重化運用のシステム構成



ポイント

- WebAdminを使ってレプリケーションクラスタにMirroring Controllerをセットアップすると、[ホスト名]に指定されたホスト名(IPアドレス)が割り当てられたネットワークを、管理用ネットワークとログ転送用ネットワークに使用します。
- 業務用ネットワークと異なるネットワークをログ転送用ネットワークとして使用する場合には、事前にレプリケーションクラスタを構築する際に、[ホスト名]に業務ネットワークと異なるログ転送用ネットワークが利用するホスト名を指定してください。

注意

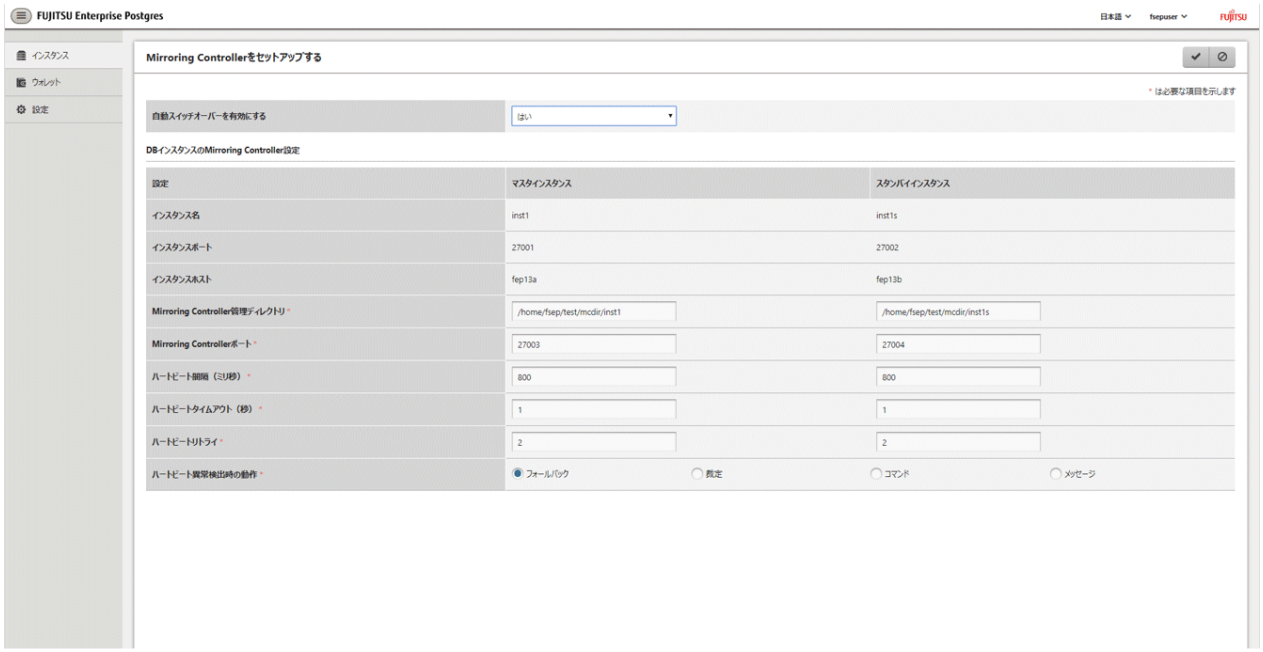
WebAdminで裁定サーバをセットアップする場合は、裁定サーバにWebAdminをインストールする必要があります。

5.1 Mirroring Controllerのセットアップ

ストリーミングレプリケーションクラスタにMirroring Controllerをセットアップする場合は、以下の手順で行ってください。

1. [インスタンス]タブで、Mirroring Controllerをセットアップするスタンバイインスタンスを選択します。
2. をクリックします。
3. セットアップするMirroring Controllerの情報を入力します。

   以下は、マスタインスタンス“inst1”とスタンバイインスタンス“inst1s”を持つレプリケーションクラスタにMirroring Controllerをセットアップする例です。



マスタインスタンスとスタンバイインスタンスのインスタンス名、ホストアドレスとポートが簡単に参照できるように表示されています。

Mirroring Controllerをセットアップするために、上のスクリーンショットに従って、マスタインスタンスとスタンバイインスタンスの欄に以下の項目を入力します。

- [自動スイッチオーバーを有効にする]: 自動切り替えと切り離し機能を有効にするかを設定します。“はい”を選択します。初期設定は“いいえ”です。
- [Mirroring Controller管理ディレクトリ]: Mirroring Controllerの構成ファイルが保存されるディレクトリです。[Mirroring Controller管理ディレクトリ]が指定されると、WebAdminは、対応するインスタンスのデータ格納パスに基づいて、入力されたディレクトリ内のMirroring Controllerの構成ファイルを検索します。Mirroring Controllerの構成ファイルが存在した場合、Mirroring Controllerに関する項目が自動的に表示されます。
- [Mirroring Controller ポート]: Mirroring Controllerのポート番号です。
- [ハートビート間隔(ミリ秒)]: 2回の連続するハートビートチェック間のミリ秒。初期設定は“800”です。
- [ハートビートタイムアウト(秒)]: ハートビートタイムアウトの秒数。初期設定は“1”です。
- [ハートビートリトライ]: フェイルオーバが起こる前のハートビート監視のリトライ数。初期設定は“2”です。
- [ハートビート異常検出時の動作]: OS/サーバの生死監視においてハートビート異常を検出した場合の動作。デフォルトは“フォールバック”です。

[ハートビート異常検出時の動作]に“裁定”を設定すると、さらに以下の項目が表示されます。

- [裁定用ネットワークのIPアドレス]: 裁定用ネットワークのIPアドレスです。

ー [Mirroring Controller裁定用ポート]: 裁定サーバと通信するMirroring Controllerのポート番号です。

また、以下の項目を選択できる[裁定サーバの設定]セクションも表示されます。[裁定サーバの設定]セクションの内容は、自動的に表示されません。

ー [場所]: 裁定サーバの場所です。設定に応じて“ローカル”または“リモート”を選択できます。

裁定サーバとWebAdminサーバが同じサーバに配置されている場合は“ローカル”を選択することができ、以下の項目が表示されます。

- [裁定プロセスの管理ディレクトリ]: 裁定サーバの構成ファイルが保存されるディレクトリです。
- [裁定サーバのホストまたはIPアドレス]: 裁定サーバのホスト名またはIPアドレスです。
- [裁定プロセス用ポート]: 裁定プロセス用のポート番号です。
- [フェンシングコマンドのファイルパス]: 異常が検出されたときにデータベースサーバをフェンシングするフェンシングコマンドのパスです。

“リモート”が設定されている場合、上記に加えて以下の項目も表示されます。

- [裁定サーバの設定]セクションに[OSクレデンシャル]が表示され、以下の情報を入力できます。

  [ユーザー名]: 裁定サーバにアクセスするためのユーザー名です。

  [パスワード]: 裁定サーバにアクセスするためのパスワードです。

- [裁定サーバ用のリモートWebAdmin]セクションに、以下の項目が表示されます。

  [リモートWebAdminアドレス]: 裁定サーバにインストールされているリモートWebAdminのIPアドレスです。

  [リモートWebAdminポート]: 裁定サーバにインストールされているWebAdminのポートです。

[ハートビート異常検出時の動作]に“コマンド”を設定すると、さらに以下の項目が表示されます。

ー [裁定コマンドのファイルパス]: 異常が検出されたときに実行する裁定コマンドのパスです。

ー [フェンシングコマンドのファイルパス]: 異常が検出されたときにデータベースサーバをフェンシングするフェンシングコマンドのパスです。

4. ✔をクリックしてMirroring Controllerをセットアップしてください。

5. セットアップが完了すると、Mirroring Controllerがマスタインスタンスとスタンバイインスタンスで起動します。

   [インスタンス]タブで、“inst1s”を選択します。以下のページが表示され、Mirroring Controllerのステータスも分かります。

Mirroring Controllerがセットアップされると、 ([Mirroring Controllerを編集する])ボタンと ([Mirroring Controllerの設定])ボタンが使用できます。

[ハートビート異常検出時の動作]が“裁定”の場合、このページに裁定ステータス(“online”または“offline”)、裁定サーバのIPアドレスおよび裁定プロセス用のポートが表示されます。

注意

OSクレデンシャル(インスタンス管理者のユーザー名とパスワード)に指定できない文字については、“付録E WebAdminで使用できない文字”を参照してください。

5.2 Mirroring Controllerのセットアップを編集する

“5.1 Mirroring Controllerのセットアップ”で設定した内容は、マスタインスタンスかスタンバイインスタンスのどちらかで編集することができます。

Mirroring Controllerの設定内容を編集する場合は、以下の手順で行ってください。

1. [インスタンス]タブで、Mirroring Controllerの設定内容を編集するインスタンスを選択します。
2. をクリックします。
3. Mirroring Controllerの情報を変更します。詳細は、“5.1 Mirroring Controllerのセットアップ”を参照してください。
4. をクリックしてMirroring Controllerの設定内容を更新します。
5. 完了すると、マスタインスタンスとスタンバイインスタンスでMirroring Controllerが起動します。

[Mirroring Controllerを編集する]ページを編集して保存すると、ページ上に表示されていないMirroring Controllerのその他のすべての設定がデフォルト値にリセットされます。

5.3 Mirroring Controllerの設定

WebAdminを使って、Mirroring Controllerの監視および制御に関する情報(“A.4.1 データベースサーバのサーバ定義ファイル”参照)やMirroring Controller裁定プロセスでの裁定および制御に関する情報(“A.4.2 裁定定義ファイル”参照)を設定することができます。マスタインスタンスかスタンバイインスタンスのどちらかで設定内容の参照や更新が可能です。

以下の手順で行ってください。

1. [インスタンス]タブで、Mirroring Controllerの設定を参照するインスタンスを選択します。
2. をクリックしてMirroring Controllerの設定を参照します。
3. をクリックしてMirroring Controllerの設定の編集ページを表示します。[Mirroring Controllerのセットアップ]で定義されたMirroring Controllerの設定は、このページでは読み取り専用です。詳細は、“5.1 Mirroring Controllerのセットアップ”を参照してください。設定の詳細や対応するパラメータ名については、“付録A パラメータ”を参照してください。

   以下の項目は、[ハートビート異常検出時の動作]のすべてに共通しています。

   - 対象DB
   - コアファイルのパス
   - システムログのファシリティ
   - システムログのID
   - 管理通信のタイムアウト時間(ミリ秒)
   - 監視プロセスの生死監視のタイムアウト時間(秒)
   - データベースプロセスの生死監視の間隔時間(ミリ秒)
   - データベースプロセスの生死監視のタイムアウト時間(秒)
   - データベースプロセスの生死監視のリトライ回数
   - データベースプロセスの無応答監視に対する動作
   - ディスクの異常監視の間隔(ミリ秒)
   - ディスクの異常監視のリトライ回数
   - テーブル空間格納ディレクトリの異常検知時の動作
   - 切り替え後コマンドのファイルパス
   - 昇格後コマンドのファイルパス
     (昇格後コマンドは、FUJITSU Enterprise Postgres 12で“切り替え後コマンド”に名称が変更となりましたが、Mirroring Controllerのサーバ定義ファイルでpost_promote_commandパラメータが使用されている場合に表示されます。)
   - 組み込み後コマンドのファイルパス
   - 切り離し前コマンドのファイルパス
   - 状態遷移コマンドのタイムアウト時間(秒)
   - 同期スタンバイ名の妥当性の確認

   [ハートビート異常検出時の動作]に“裁定”を設定すると、さらに以下の項目が表示されます。

   - 裁定処理のタイムアウト時間(秒)
   - 裁定プロセスとの接続監視の間隔時間(ミリ秒)
   - 裁定プロセスとの接続監視のリトライ回数
   - 裁定プロセスとの接続監視のタイムアウト時間(秒)
   - 裁定プロセスとの接続の試行間隔(ミリ秒)

- 裁定プロセスとの接続のタイムアウト時間(秒)
- フェンシングコマンドのファイルパス
- フェンシングコマンドのタイムアウト時間(秒)
- 切り離された同期スタンバイのインスタンスの動作

[ハートビート異常検出時の動作]に“裁定”が設定されている場合、以下の項目を選択できる[裁定サーバの設定]セクションも表示されます。

- コアファイルのパス
- システムログのファシリティ
- システムログのID
- フェンシングコマンドのタイムアウト時間(秒)
- ハートビート間隔（ミリ秒）
- ハートビートタイムアウト（秒）
- ハートビートリトライ回数

[ハートビート異常検出時の動作]に“コマンド”を設定すると、さらに以下の項目を使用できます。

- フェンシングコマンドのタイムアウト時間(秒)
- 裁定コマンドのタイムアウト時間(秒)
- 切り離された同期スタンバイのインスタンスの動作

[ハートビート異常検出時の動作]に“メッセージ”を設定すると、さらに以下の項目を使用できます。

- フェンシングコマンドのファイルパス
- フェンシングコマンドのタイムアウト時間(秒)

また、Mirroring Controllerがインスタンスに接続する際の下記の接続パラメータも設定できます。

- データベースのインスタンスへ接続するlibpq接続SSLパラメータ
- データベースのインスタンスへ接続するJDBC接続SSLパラメータ

4. をクリックしてMirroring Controllerの設定を更新します。

5.4 Mirroring Controllerの停止

WebAdminを使ってマスタインスタンスかスタンバイインスタンスのどちらかでMirroring Controllerを停止することができます。

Mirroring Controllerを停止する場合は、以下の手順で行ってください。

1. [インスタンス]タブで、Mirroring Controllerを停止するインスタンスを選択します。
2. をクリックします。
3. 確認ダイアログボックスで、[はい]ボタンをクリックしてください。

選択されたインスタンスでMirroring Controllerが停止します。Mirroring Controllerのステータスが更新され、確認メッセージが[メッセージ]欄に表示されます。

5.5 Mirroring Controllerの起動

WebAdminを使ってマスタインスタンスかスタンバイインスタンスのどちらかでMirroring Controllerを起動することができます。

Mirroring Controllerを起動する場合は、以下の手順で行ってください。

1. [インスタンス]タブで、Mirroring Controllerを起動するインスタンスを選択します。
2. をクリックします。
3. 確認ダイアログボックスで、希望するフェイルオーバモードを選択し、[はい]をクリックしてください。

選択されたインスタンスでMirroring Controllerが起動します。Mirroring Controllerのステータスが更新され、確認メッセージが[メッセージ]欄に表示されます。

5.6 フェイルオーバモードを無効にする

Mirroring Controllerでフェイルオーバモードを無効にするとマスタインスタンスとスタンバイインスタンス間の自動切り替えと切り離しが無効になります。

フェイルオーバモードを無効にする場合は、以下の手順で行ってください。

1. [インスタンス]タブで、インスタンスを選択します。
2. をクリックします。
3. 確認ダイアログボックスで、[はい]をクリックしてください。

Mirroring Controllerでフェイルオーバモードが無効になります。Mirroring Controllerのステータスが更新され、確認メッセージが[メッセージ]欄に表示されます。

5.7 フェイルオーバモードを有効にする

Mirroring Controllerでフェイルオーバモードを有効にするとマスタインスタンスとスタンバイインスタンス間の自動切り替えと切り離しが有効になります。

フェイルオーバを有効にする場合は、以下の手順で行ってください。

1. [インスタンス]タブで、インスタンスを選択します。
2. をクリックします。
3. 確認ダイアログボックスで、[はい]をクリックしてください。

Mirroring Controllerでフェイルオーバモードが有効になります。Mirroring Controllerのステータスが更新され、確認メッセージが[メッセージ]欄に表示されます。

5.8 Mirroring Controllerセットアップの削除

Mirroring Controllerセットアップを削除すると、マスタインスタンスとスタンバイインスタンスでのセットアップが削除されます。

1. [インスタンス]タブで、インスタンスを選択します。
2. をクリックします。
3. 確認ダイアログボックスで、[はい]をクリックしてください。

Mirroring Controllerセットアップがクラスタから削除されます。クラスタのステータスが更新され、確認メッセージが[メッセージ]欄に表示されます。

FUJITSU Enterprise Postgres 12以降では、Mirroring Controller管理ディレクトリと設定ファイルは削除されません。

5.9 フェイルオーバ後のステータス更新

Mirroring Controllerでフェイルオーバを行う場合、スタンバイインスタンスがスタンドアロンインスタンスに昇格します。Mirroring Controllerのセットアップがスタンバイインスタンスとマスタインスタンスの両方で削除されます。

フェイルオーバを起動させる方法の一例とWebAdminでMirroring Controllerを使うことで得られる結果を以下に示しています。

1. [インスタンス]タブで、マスタインスタンス“inst1”を選択します。
2. をクリックします。
3. 確認ダイアログボックスで、“このインスタンスはMirroring Controllerに監視されています。インスタンスを停止すると、クラスタフェイルオーバが発生するかもしれません”という警告が表示されます。
4. 停止モードを選択して[はい]をクリックします。

   サーバでは、以下が行われます。

   a. マスタインスタンスが停止します。
   b. Mirroring Controllerでフェイルオーバが起動します。
   c. Mirroring Controllerのセットアップがスタンバイインスタンスとマスタインスタンスの両方で削除されます。
   d. スタンバイインスタンスがスタンドアロンインスタンスに昇格します。
5. インスタンスをWebAdminでリフレッシュすると、インスタンスの最新のステータスが表示されます。

注意

フェイルオーバを行うとMirroring Controllerのセットアップがスタンバイインスタンスとマスタインスタンスの両方で削除されます。そのため、再度WebAdminを使用したMirroring Controllerの管理を行うためには、スタンバイインスタンスの作成およびMirroring Controllerのセットアップを実施してください。

スタンバイインスタンスの作成の詳細については、“運用ガイド”の“スタンバイインスタンスの作成”を参照してください。

Mirroring Controllerのセットアップの詳細については、“5.1 Mirroring Controllerのセットアップ”を参照してください。

5.10 管理用ネットワーク兼ログ転送用ネットワークの異常時の対処

サーバの高負荷状態やシステムリソースの不足などの原因により、管理ネットワーク兼ログ転送用ネットワークが一時的に通信異常となる場合があります。これにより、サーバが健全な状態にもかかわらず誤ってスプリットブレインを発生させる危険性があります。

スプリットブレインとは、一時的に両サーバがプライマリサーバとして動作してしまうために、両サーバでデータの更新業務が行われる現象をいいます。

WebAdminを利用したスプリットブレインの検知方法

以下の条件の場合にスプリットブレインが発生している可能性があります。“付録D ハートビート異常で即時に自動縮退を行うを選択した場合の注意事項”の“スプリットブレインの検出方法”および“スプリットブレインからの復旧方法”を参照して、対処を実施してください。

1. [インスタンス]タブで、スタンバイインスタンスを選択する。かつ、
2. [インスタンスの種類]に“Standalone”と表示される。かつ、
3. [インスタンス]タブで、マスタインスタンスを選択する。かつ、
4. [インスタンスの種類]に“Standalone”と表示される。

注意

管理用ネットワークは、Mirroring Controllerが各サーバの状態を確認しあうための重要なネットワークです。

また、ログ転送用ネットワークは、データの鮮度を維持するための重要なネットワークです。

したがって、これらにはOSやネットワークドライバベンダが提供するネットワークの冗長化機能のチャンネルボンディング機能を使用して、障害に強いネットワーク構成にしてください。

5.11 裁定サーバを利用して自動縮退を行う運用

WebAdminによるデータベース多重化運用では、裁定サーバを利用した自動縮退を行うことも可能です。その場合には、WebAdminでMirroring Controllerをセットアップした後に、データベースサーバと裁定サーバでの操作が必要です。

裁定サーバでの操作

Mirroring Controllerコマンドを利用した裁定サーバのセットアップを行います。

1. 裁定サーバのセットアップを行います。

   裁定サーバのセットアップ手順については、“第2章 データベース多重化運用のセットアップ”から“2.3 裁定サーバのセットアップ”を参照してください。

データベースサーバでの操作

WebAdminによるMirroring Controllerのセットアップ後に設定を一部変更します。

1. WebAdminでMirroring Controllerをセットアップしてください。

   詳細は、“5.1 Mirroring Controllerのセットアップ”を参照してください。

2. マスタインスタンスとスタンバイインスタンスのMirroring ControllerをWebAdminにより停止してください。

   詳細は、“5.4 Mirroring Controllerの停止”を参照してください。

3. マスタインスタンスとスタンバイインスタンスのネットワーク定義ファイルを編集し、裁定サーバの情報を追加してください。

   ネットワーク定義ファイルは、Mirroring Controllerのセットアップ時に指定した“Mirroring Controller管理ディレクトリ”に存在するnetwork.confです。詳細は、“A.3 ネットワーク定義ファイル”を参照してください。

   以下にnetwork.confの定義例を示します。

   例）
   裁定用ネットワークとして使用するデータベースサーバ側のポート番号に“27541”、Mirroring Controller裁定プロセスのサーバ識別子に“arbiter”、ポート番号に“27541”を指定する場合の例を以下に示します。

```
dbsvm27500 192.0.2.100,192.0.3.100 27540,27541 server
dbsvs27500 192.0.2.110,192.0.3.110 27540,27541 server
arbiter 192.0.3.120 27541 arbiter
```

注意

- ポート番号は、データベースサーバ、裁定サーバのどのサーバにおいても、他のソフトウェアと重ならないように設定してください。また、管理用ネットワークと裁定用ネットワークのセグメントを兼用して構成しないでください。
- サーバ種別が“server”の場合、IPアドレスまたはホスト名、およびポート番号を2つずつ以下の順序で指定する必要があります。
  - 管理用ネットワークとして使用するデータベースサーバ側のIPアドレスまたはホスト名
  - 裁定用ネットワークとして使用するデータベースサーバ側のIPアドレスまたはホスト名
  - 管理用ネットワークとして使用するデータベースサーバ側のポート番号
  - 裁定用ネットワークとして使用するデータベースサーバ側のポート番号
- サーバ種別が“arbiter”の場合、IPアドレスまたはホスト名は裁定サーバのarbitration.confファイルのmy_addressパラメータに指定したIPアドレスまたはホスト名、ポート番号は裁定サーバのarbitration.confファイルのportパラメータに指定したポート番号を指定してください。

4. マスタインスタンスとスタンバイインスタンスのサーバ定義ファイルを編集し、裁定サーバを利用して自動縮退を行う運用に必要なパラメータを追加してください。

   サーバ定義ファイルは、Mirroring Controllerのセットアップ時に指定した“Mirroring Controller管理ディレクトリ”に存在する“インスタンス名”“インスタンスポート”.confです。

   裁定サーバを利用して自動縮退を行う運用は、heartbeat_error_actionパラメータにarbitrationの指定が必要です。

   その他のパラメータについては、“A.4.1 データベースサーバのサーバ定義ファイル”を参照してください。

5. マスタインスタンスとスタンバイインスタンスのMirroring ControllerをWebAdminにより起動します。

   詳細は、“5.5 Mirroring Controllerの起動”を参照してください。

共通の操作

1. データベースサーバまたは裁定サーバからMirroring Controllerのコマンドにより接続状態を確認します。

   接続状態の確認手順については、“2.8 接続状態の確認”を参照してください。

付録A　パラメータ

データベース多重化運用で必要となる各設定ファイルおよびパラメータについて、説明します。

参照

postgresql.confファイルの詳細は、“PostgreSQL Documentation”の“Server Configuration”を参照してください。

A.1　プライマリサーバで設定するパラメータ

プライマリサーバのpostgresql.confファイルに指定するパラメータの内容を以下に示します。

表A.1 postgresql.confファイル

パラメータ	設定値	説明
wal_level	replicaまたはlogical	トランザクションログの出力レベルを指定します。 ロジカルデコーディングも使用する場合にはlogicalを指定します。
max_wal_senders	2以上	Mirroring Controllerのクラスタシステムを構築する場合は、2を指定します。 また、クラスタシステムに非同期スタンバイサーバを追加で接続する場合には、これらのスタンバイサーバからの同時接続数を加算してください。
synchronous_standby_names	'スタンバイサーバ名'	スタンバイサーバを識別する任意の名前を半角シングルクォートで囲んで指定します。 Mirroring Controller起動中は、本パラメータを変更しないでください。 Mirroring Controllerが管理できるスタンバイサーバは1つであるため、本パラメータに複数の名前を指定しないでください。
hot_standby	on	スタンバイサーバ上で問い合わせを実行できるかどうかを指定します。 onを指定してください。
wal_keep_size	WALの保存サイズ(メガバイト)	本パラメータの設定値を越える遅延が発生した場合、プライマリサーバはスタンバイサーバが今後必要とするWALセグメントを削除する可能性があります。 また、保守作業などでスタンバイサーバを停止させる場合、停止時間を考慮してWALセグメントが削除されない値を設定してください。 WALの保存サイズの見積もりの詳細は、“導入ガイド(サーバ編)”の“トランザクションログの容量の見積り”を参照してください。
wal_log_hints	on	スタンバイサーバの復旧時にpg_rewindコマンドを使用する場合には、本パラメータを設定するかまたはinitdbコマンド実行時にチェックサムを有効にしてください。

パラメータ	設定値	説明
wal_sender_timeout	タイムアウト時間(ミリ秒数)	プライマリサーバ側でのトランザクションログの受信プロセス(walreceiver)が異常な状態になったと判断する時間を指定します。 指定値は、スタンバイサーバのpostgresql.confファイルに設定するwal_receiver_status_intervalパラメータの値より大きくする必要があります。 データベースプロセスの生死監視時間と合わせることで、異常と判断する時間を統一できます。
wal_receiver_timeout	タイムアウト時間(ミリ秒数)	スタンバイサーバ側でのトランザクションログの受信に異常が発生したと判断する時間を指定します。 データベースプロセスの生死監視時間と合わせることで、異常と判断する時間を統一できます。
archive_mode	on	アーカイブログモードを指定します。
archive_command	'インストールディレクトリ/bin/pgx_walcopy.cmd "%p" "バックアップデータ格納先ディレクトリ/archived_wal/%f"'	トランザクションログを保存するコマンドと格納先パス名を指定します。
backup_destination	バックアップデータ格納先ディレクトリ名	バックアップデータ格納先ディレクトリ名を指定します。 指定するディレクトリは、インスタンス管理者ユーザーのみがアクセスできるように権限を設定してください。 他のサーバのバックアップデータを使ってリカバリを正しく行うために、すべてのサーバで同じパスになるように指定してください。
listen_addresses	プライマリサーバのIPアドレス、ホスト名、または‘*’	プライマリサーバのIPアドレスまたはホスト名を指定します。ログ転送用ネットワークに接続するIPアドレスまたは対応するホスト名を指定します。 指定内容は、クライアントアプリケーションからの接続を許可する場合にも使用されます。 任意のクライアントやスタンバイサーバからの受付を行う場合は、‘*’を指定することができます。 詳細は“PostgreSQL Documentation”の“Connections and Authentication”を参照してください。
max_connections	インスタンスへのクライアントの同時接続数＋superuser_reserved_connectionsの値	指定値は、クライアントアプリケーションからの接続数とインスタンス管理のための接続数を制限するためにも使用されます。 詳細は、“導入ガイド（サーバ編）”の“initdbコマンドでインスタンスを作成した場合”、および、“PostgreSQL Documentation”の“Connections and Authentication”を参照してください。

パラメータ	設定値	説明
superuser_reserved_connections	mc_ctl statusの同時実行数(注)＋2を加算	データベーススーパーユーザーからの接続のために予約されている接続数を指定します。 Mirroring Controllerプロセスからの接続数を加算します。また、加算した値はmax_connectionsパラメータにも反映してください。
restart_after_crash	off	restart_after_crashパラメータにonを指定またはデフォルトの場合、一部のサーバプロセスがダウンした際に、クラッシュリカバリを含めたEnterprise Postgresの再起動に相当する動作が行われます。 しかし、データベース多重化運用による監視の機能で一部のサーバプロセスがダウンした時点で異常を検知してフェイルオーバが発生し、再起動途中のサーバプロセスが強制停止されます。このような意味なく再起動して途中でキャンセルされるという複雑な動作を抑止するためにoffを設定します。
synchronous_commit	onまたはremote_apply	トランザクションのコミット処理がクライアントに正常終了の応答を返す前に、スタンバイサーバへのWAL送信をどこまで行うかを指定します。 切り替えや切り替え直後のOSやサーバのダウンからのデータ損失を防ぐために、onまたはremote_applyを推奨します。
recovery_target_timeline	latest	切り替え発生時に、新スタンバイサーバ(旧プライマリサーバ)が新プライマリサーバに追随するために、latestを指定します。 本パラメータの指定は、プライマリサーバの切り替え後、旧プライマリサーバを新しくスタンバイサーバとして組み込む時に必要です。

注) mc_ctlコマンドのstatusモードでの同時実行数です。

A.2 スタンバイサーバで設定するパラメータ

スタンバイサーバで設定するファイルおよびパラメータの内容について、説明します。

postgresql.confファイルの以下のパラメータを編集したあと、インスタンスを起動してください。

表A.2 postgresql.confファイル

パラメータ	設定値	説明
wal_level	replicaまたはlogical	トランザクションログの出力レベルを指定します。 ロジカルデコーディングも使用する場合にはlogicalを指定します。
max_wal_senders	2以上	Mirroring Controllerのクラスタシステムを構築する場合は、2を指定します。 また、クラスタシステムに非同期スタンバイサーバを追加で接続する場合には、これらのスタンバイサーバからの同時接続数を加算してください。

パラメータ	設定値	説明
synchronous_standby_names	'プライマリサーバ名'	プライマリサーバを識別する任意の名前を半角シングルクォートで囲んで指定します。 この名前は、プライマリサーバを切り替えたあとに、旧プライマリサーバを新しくスタンバイサーバとして組み込むときに必要です。 Mirroring Controller起動中は、本パラメータを変更しないでください。 Mirroring Controllerが管理できるスタンバイサーバは1つであるため、本パラメータに複数の名前を指定しないでください。
hot_standby	on	スタンバイサーバ上で問い合わせを実行できるかどうかを指定します。 onを指定してください。
wal_keep_size	WALの保存サイズ(メガバイト)	本パラメータの設定値を越える遅延が発生した場合、プライマリサーバはスタンバイサーバが今後必要とするWALセグメントを削除する可能性があります。 また、保守作業などでスタンバイサーバを停止させる場合、停止時間を考慮してWALセグメントが削除されない値を設定してください。 WALの保存サイズの見積もりの詳細は、“導入ガイド(サーバ編)”の“トランザクションログの容量の見積り”を参照してください。
wal_log_hints	on	スタンバイサーバの復旧時にpg_rewindコマンドを使用する場合には、本パラメータを設定するかまたはinitdbコマンド実行時にチェックサムを有効にしてください。
wal_sender_timeout	タイムアウト時間(ミリ秒数)	プライマリサーバ側でのトランザクションログの受信プロセス(walreceiver)が異常な状態になったと判断する時間を指定します。 指定値は、スタンバイサーバのpostgresql.confファイルに設定するwal_receiver_status_intervalパラメータの値より大きくする必要があります。 データベースプロセスの生死監視時間と合わせることで、異常と判断する時間を統一できます。
wal_receiver_timeout	タイムアウト時間(ミリ秒数)	スタンバイサーバ側でのトランザクションログの受信に異常が発生したと判断する時間を指定します。 データベースプロセスの生死監視時間と合わせることで、異常と判断する時間を統一できます。
backup_destination	バックアップデータ格納先ディレクトリ名	バックアップデータ格納先ディレクトリ名を指定します。 指定するディレクトリは、インスタンス管理者ユーザーのみがアクセスできるように権限を設定してください。 他のサーバのバックアップデータを使ってリカバリを正しく行うために、すべてのサーバで同じパスになるように指定してください。
archive_mode	on	アーカイブログモードを指定します。
archive_command	'インストールディレクトリ/bin/pgx_walcopy.cmd "%p" "バックアップ	トランザクションログを保存するコマンドと格納先パス名を指定します。

パラメータ	設定値	説明
	データ格納先ディレクトリ/archived_wal/%f'''	
listen_addresses	スタンバイサーバのIPアドレス、ホスト名、または‘*’	スタンバイサーバのIPアドレスまたはホスト名を指定します。ログ転送用ネットワークに接続するIPアドレスまたは対応するホスト名を指定します。 指定内容は、クライアントアプリケーションからの接続を許可する場合にも使用されます。 任意のクライアントやスタンバイサーバからの受付を行う場合は、‘*’を指定することができます。 詳細は“PostgreSQL Documentation”の“Connections and Authentication”を参照してください。
max_connections	インスタンスへのクライアントの同時接続数＋superuser_reserved_connectionsの値	指定値は、クライアントアプリケーションからの接続数とインスタンス管理のための接続数を制限するためにも使用されます。 詳細は、“導入ガイド（サーバ編）”の“initdbコマンドでインスタンスを作成した場合”、および、“PostgreSQL Documentation”の“Connections and Authentication”を参照してください。
superuser_reserved_connections	mc_ctl statusの同時実行数(注)＋2を加算	データベーススーパーユーザーからの接続のために予約されている接続数を指定します。 Mirroring Controllerプロセスからの接続数を加算します。また、加算した値はmax_connectionsパラメータにも反映してください。
restart_after_crash	off	restart_after_crash パラメータにonを指定またはデフォルトの場合、一部のサーバプロセスがダウンした際に、クラッシュリカバリを含めたEnterprise Postgresの再起動に相当する動作が行われます。 しかし、データベース多重化運用による監視の機能で一部のサーバプロセスがダウンした時点で異常を検知してフェイルオーバが発生し、再起動途中のサーバプロセスが強制停止されます。このような意味なく再起動して途中でキャンセルされるという複雑な動作を抑止するためにoffを設定します。
synchronous_commit	onまたはremote_apply	トランザクションのコミット処理がクライアントに正常終了の応答を返す前に、スタンバイサーバへのWAL送信をどこまで行うかを指定します。 切り替えや切り替え直後のOSやサーバのダウンからのデータ損失を防ぐために、onまたはremote_applyを推奨します。
primary_conninfo	‘ストリーミングレプリケーションの接続先情報’	ストリーミングレプリケーションの接続先情報を半角シングルクォートで囲んで指定します。本パラメータのデフォルト値は、インスタンスのセットアップのpg_basebackupを実行する手順において、自動的にpostgresql.auto.confに設定されます。
recovery_target_timeline	latest	切り替え発生時に、新スタンバイサーバ(旧プライマリサーバ)が新プライマリサーバに追随するために、latestを指定します。

パラメータ	設定値	説明
		本パラメータの指定は、プライマリサーバの切り替え後、旧プライマリサーバを新しくスタンバイサーバとして組み込む時に必要です。
streaming_wal_compression	-1から9までの数値	ストリーミングレプリケーションで、WALを圧縮して送信したい場合に指定します。 -1: zlibのデフォルトの圧縮レベルで圧縮します 0: 無圧縮 1-9:指定された圧縮レベルで圧縮します。9が高圧縮となります 本パラメータを指定しない場合は、0を指定したものとみなされます。

注) mc_ctlコマンドのstatusモードでの同時実行数です。

A.3 ネットワーク定義ファイル

データベースサーバと裁定サーバのそれぞれに定義する、ネットワーク定義ファイル(network.confファイル)について説明します。プライマリサーバとスタンバイサーバは同じ内容を定義してください。

データベース多重化運用では、network.confファイルに、以下に関するネットワークの構成を定義します。

- Mirroring Controllerのプロセス間の連携
- Mirroring ControllerプロセスとMirroring Controller裁定プロセスの連携

network.confファイルに定義する項目

［指定形式］

```
サーバ識別子 ホスト名[, ホスト名] ポート番号[, ポート番号] [サーバ種別]
または、
サーバ識別子 IPアドレス[, IPアドレス] ポート番号[, ポート番号] [サーバ種別]
```

サーバ識別子、IPアドレスまたはホスト名、ポート番号、およびサーバ種別を半角空白で区切って指定します。

各項目の説明を以下に示します。

表A.3 network.confファイル

項目	説明
サーバ識別子	サーバを識別する任意の識別子を指定します。最大長は64バイトです。半角空白または#以外のASCII文字で指定してください。
IPアドレスまたはホスト名	データベースサーバ間の通信を行う管理用ネットワークおよびデータベースサーバと裁定サーバ間の通信を行う裁定用ネットワークに接続する、IPアドレスまたは対応するホスト名を指定します。 IPアドレスまたはホスト名をカンマ区切りで2つ指定する場合は、カンマの後にスペースを入れないでください。 ホスト名は、半角空白以外のASCII文字で指定してください。
ポート番号	ポート番号は、0 ～ 65535の範囲を超えて指定できません。他のソフトウェアと重ならないように設定してください。また、他のプログラムによって一時的に割り当てられる可能性のあるエフェメラルポートを指定しないでください。なお、本パラメータに指定したポート番号は、サービスファイルに設定する必要があります。 ポート番号をカンマ区切りで2つ指定する場合は、カンマの後ろにスペースを入れないでください。
サーバ種別	サーバ種別は、データベースサーバを示す“server”、または裁定サーバを示す“arbiter”を指定してください。なお、データベースサーバを示す“server”は省略可能です。

データベースサーバでの定義内容

データベースサーバに定義するnetwork.confの定義内容を説明します。

ハートビート異常検出時の動作の設定値によって、定義内容が異なります。

裁定サーバによる自動縮退を選択した場合

- 管理用ネットワークと裁定用ネットワークに関する定義を行います。
- サーバ種別(データベースサーバまたは裁定サーバ)に応じて、IPアドレスまたはホスト名、およびポート番号を下表のとおりに指定します。

サーバ種別	IPアドレスまたはホスト名		ポート番号	
	1つ目	2つ目	1つ目	2つ目
server	管理用ネットワークとして使用するIPアドレスまたはホスト名	裁定用ネットワークとして使用するIPアドレスまたはホスト名(注)	管理用ネットワークとして使用するポート番号	裁定用ネットワークとして使用するポート番号(注)
arbiter	裁定サーバ側のIPアドレスまたはホスト名 裁定サーバのarbitration.confのmy_addressパラメータに指定した値と同じ値を指定してください。	指定不要	裁定サーバ側のポート番号 裁定サーバのarbitration.confのportパラメータに指定した値と同じ値を指定してください。	指定不要

注)自サーバの定義でなければ省略可能ですが、省略した場合、プライマリサーバとスタンバイサーバそれぞれでnetwork.confファイルの作成が必要です。

設定例)

[IPv4の場合]

```
server1 192.0.2.100,192.0.3.100 27540,27541 server
server2 192.0.2.110,192.0.3.110 27540,27541 server
arbiter 192.0.3.120             27541       arbiter
```

[IPv6の場合]

```
server1 2001:258:8404:1217:250:56ff:fea7:559f,2001:258:8404:1217:250:56ff:fea8:559f 27540,27541 server
server2 2001:258:8404:1217:250:56ff:fea7:55a0,2001:258:8404:1217:250:56ff:fea8:55a0 27540,27541 server
arbiter 2001:258:8404:1217:250:56ff:fea8:55a0                                       27541       arbiter
```

裁定サーバによる自動縮退以外を選択した場合

- 管理用ネットワークに関する定義を行います。
- プライマリサーバ、スタンバイサーバで同じ内容を定義します。
- サーバ種別がデータベースサーバの行のみを定義します。
- IPアドレスまたはホスト名、ポート番号は1つだけ指定します。

IPアドレスまたはホスト名		ポート番号	
1つ目	2つ目	1つ目	2つ目
管理用ネットワークとして使用するIPアドレスまたはホスト名	指定不要	管理用ネットワークとして使用するポート番号	指定不要

設定例)

[IPv4の場合]

```
server1 192.0.2.100 27540
server2 192.0.2.110 27540
```

[IPv6の場合]

```
server1 2001:258:8404:1217:250:56ff:fea7:559f 27540
server2 2001:258:8404:1217:250:56ff:fea7:55a0 27540
```

裁定サーバでの定義内容

裁定サーバに定義するnetwork.confの定義内容を説明します。

- 裁定用ネットワークに関する定義を行います。
- サーバ種別がデータベースサーバの行のみを定義します。
- IPアドレスまたはホスト名には、データベースサーバ側のnetwork.confファイルのデータベースサーバの行に指定した2つ目のIPアドレスまたはホスト名と同じ値を指定してください。
- ポート番号には、データベースサーバ側のnetwork.confファイルのデータベースサーバの行に指定した2つ目のポート番号と同じ値を指定してください。

設定例)

[IPv4の場合]

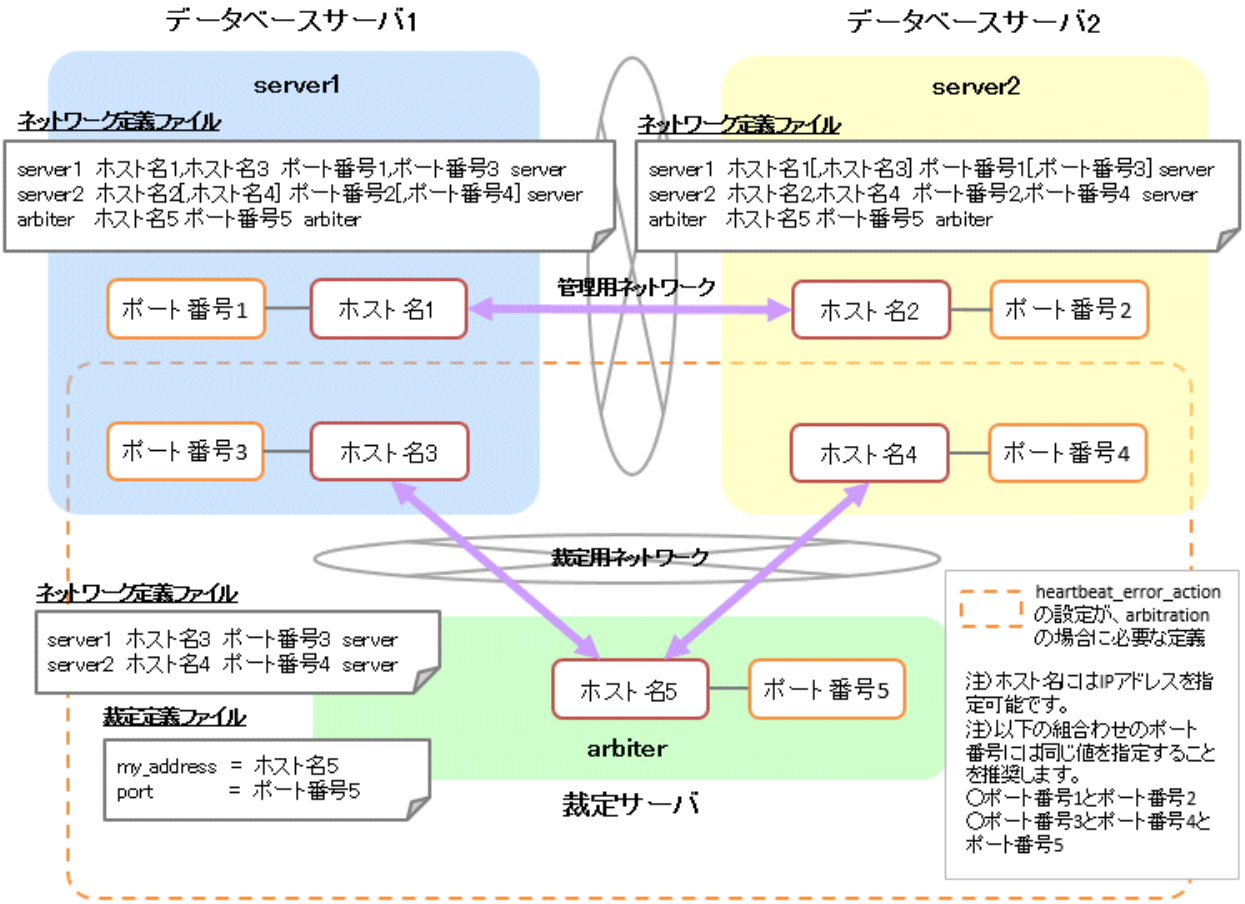
```
server1 192.0.3.100 27541
server2 192.0.3.110 27541
```

[IPv6の場合]

```
server1 2001:258:8404:1217:250:56ff:fea8:559f 27541
server2 2001:258:8404:1217:250:56ff:fea8:55a0 27541
```

ネットワークに関する定義の関係について

ネットワーク定義ファイル(network.confファイル)や裁定定義ファイル(arbitration.confファイル)に指定するホスト名および、IPアドレスまたはポート番号の関係については、以下の図を参照してください。



A.4 サーバ定義ファイル

A.4.1 データベースサーバのサーバ定義ファイル

Mirroring Controllerの監視および制御に関する情報をサーバ識別子.confファイルに定義します。サーバ識別子の最大長は、64バイトです。半角空白以外のASCII文字で指定してください。

プライマリサーバとスタンバイサーバの環境が異なる場合は、環境に合わせて異なる内容を定義してください。

表A.4 サーバ識別子.confファイル

パラメータ	設定値	説明
db_instance	'データ格納先ディレクトリ' [設定例] db_instance = '/database1/inst1'	監視対象のインスタンスを特定するデータ格納先ディレクトリを半角シングルクォートで囲んで指定してください。 ASCII文字で指定してください。
target_db	postgres、template1のいずれか	データベースインスタンスに接続するデータベースの名前を指定します。 デフォルトはpostgresです。
db_instance_username	'データベースのインスタンスへ接続するユーザー名'	データベースのインスタンスへ接続するユーザー名を指定します。 ASCII文字で指定してください。 本パラメータはデータベース管理者ユーザーとMirroring Controllerの起動を行うOSユーザーが異なる場合に指定します。

パラメータ	設定値	説明
		データベーススーパーユーザーを半角シングルクォートで囲んで指定してください。 ユーザー名の最大長は63バイトです。 デフォルトはMirroring Controllerの起動を行うOSユーザーです。
db_instance_password	'インスタンス管理者ユーザーのユーザー名に対応するパスワード'	Mirroring Controllerがデータベースのインスタンスへ接続するためのパスワードを半角シングルクォートで囲んで指定します。 ASCII文字で指定してください。 本パラメータは、Mirroring Controllerからデータベースのインスタンスへの接続設定において、パスワード認証を行う場合に指定が必要です。 パスワード認証を行わない場合に本パラメータを指定すると、無視されます。 本パラメータの指定値に'または¥を含む場合は、それぞれ¥'および¥¥と記載してください。
enable_hash_in_password	onまたはoff	db_instance_passwordの指定値に含まれている#をパスワードの文字とみなす場合はonを指定し、コメントとみなす場合はoffを指定します。 デフォルトはoffです。
core_file_path	'コアファイルを出力するディレクトリ'	コアファイルを出力するディレクトリを半角シングルクォートで囲んで指定します。 ASCII文字で指定してください。 本パラメータを省略すると、Mirroring Controller管理ディレクトリを指定したものとみなされます。
syslog_facility	LOCAL0、LOCAL1、LOCAL2、LOCAL3、LOCAL4、LOCAL5、LOCAL6、LOCAL7のいずれかを指定します。	syslogへのログ取得が有効である場合、このパラメータの値がsyslogの"facility"に使用されます。 デフォルトは、LOCAL0です。
syslog_ident(注1)	'プログラム名'	システムログ内のMirroring Controllerメッセージを特定するために使用するプログラム名を半角シングルクォートで囲んで指定します。半角空白以外のASCII文字で指定してください。 デフォルトは、'MirroringControllerOpen'です。
remote_call_timeout	管理通信のタイムアウト	Mirroring Controllerのエージェントプロセスがサーバ間で行う通信のタイムアウト値(ミリ秒)を指定します。 OSのTCP接続のタイムアウト時間(注2)よりも短くなるように0～2147483647の間の数値で指定してください。 0はタイムアウトの制限がないことを示します。 デフォルトは、70000ミリ秒(70秒)です。
agent_alive_timeout	Mirroring Controllerプロセスの生死監視のタイムアウト時間(秒数)	指定された秒数以上の時間応答がない場合は、Mirroring Controllerプロセスを再起動します。 0または2～2147483647の間の数値で指定します。0を指定した場合、タイムアウトの制限がないことを示します。

パラメータ	設定値	説明
		デフォルトは、0秒です。
heartbeat_error_action	OS/サーバの生死監視においてハートビート異常を検出した場合の動作	arbitration：裁定サーバを利用して自動縮退を行います。 command：縮退の判断を行うユーザーコマンドを呼び出して自動縮退を行います。 message：メッセージを通知します。 fallback：無条件に自動縮退を行います。 デフォルトはarbitrationです。 プライマリサーバおよびスタンバイサーバで同じ設定にしてください。
heartbeat_interval	OS/サーバの生死監視における異常監視の間隔時間(ミリ秒数)	OSやサーバを監視するため、heartbeat_intervalに指定した間隔で、異常の監視を行います。 異常を検出した場合には、heartbeat_error_actionに指定した動作に従います。heartbeat_error_actionにarbitrationを指定した場合、OS/サーバの監視の異常検知時間は、裁定サーバを利用しない運用に比べ、最大でarbitration_timeoutの指定値の分だけ長くなります。 1～2147483647の間の数値で指定してください。 指定した値はdb_instance_check_intervalとdisk_check_intervalのデフォルトとして使用されます。 デフォルトは、800ミリ秒です。
heartbeat_timeout	OS/サーバの生死監視における異常監視のタイムアウト時間(秒数)	指定された秒数以上のあいだ、応答がない場合に、プライマリサーバの切り替えやスタンバイサーバの切り離しが必要となる異常が発生したと見なします。 異常を検出した場合には、heartbeat_error_actionに指定した動作に従います。heartbeat_error_actionにarbitrationを指定した場合、OS/サーバの監視の異常検知時間は、裁定サーバを利用しない運用に比べ、最大でarbitration_timeoutの指定値の分だけ長くなります。 1～2147483647の間の数値で指定してください。 指定した値はdb_instance_check_timeoutのデフォルトとして使用されます。 デフォルトは、1秒です。
heartbeat_retry	OS/サーバの生死監視における異常監視のリトライ回数(回数)	プライマリサーバの切り替えやスタンバイサーバの切り離しが必要となる異常を検知した場合のリトライ回数を指定します。指定回数＋1回以上連続して異常を検知した場合に切り替えや切り離しを行います。 異常を検出した場合には、heartbeat_error_actionに指定した動作に従います。heartbeat_error_actionにarbitrationを指定した場合、OS/サーバの監視の異常検知時間は、裁定サーバを利用しない運用に比べ、最大でarbitration_timeoutの指定値の分だけ長くなります。指定回数＋1回以上連続して異常を検知した場合に切り替えや切り離しを行います。 0～2147483647の間の数値で指定してください。

パラメータ	設定値	説明
		指定した値はdb_instance_check_retryとdisk_check_retryのデフォルトとして使用されます。 デフォルトのリトライ回数は、2回です。
db_instance_check_interval	データベースプロセスの生死監視の間隔時間(ミリ秒数)	データベースプロセスを監視するため、db_instance_check_intervalに指定した間隔で生死監視を行います。 本パラメータの設定は、ストリーミングレプリケーションの異常監視にも使用されます。 1～2147483647の間の数値で指定してください。 デフォルトは、heartbeat_intervalの設定値です。
db_instance_check_timeout	データベースプロセスの生死監視のタイムアウト時間(秒数)	指定された秒数以上のあいだ、応答がない場合に、プライマリサーバの切り替えやスタンバイサーバの切り離しが必要となる異常が発生したと見なします。 1～2147483647の間の数値で指定してください。 デフォルトは、heartbeat_timeoutの設定値です。
db_instance_check_retry	データベースプロセスの生死監視のリトライ回数(回数)	プライマリサーバの切り替えやスタンバイサーバの切り離しが必要となる異常を検知した場合のリトライ回数を指定します。指定回数＋1回以上連続して異常を検知した場合に切り替えや切り離しを行います。ただし、データベースプロセスのダウンを検知した場合は、本パラメータの設定に関わらず即時に切り替えや切り離しを行います。 本パラメータの設定は、ストリーミングレプリケーションの異常監視にも使用されます。 0～2147483647の間の数値で指定してください。 デフォルトのリトライ回数は、heartbeat_retryの設定値です。
db_instance_timeout_action	none、message、failoverのいずれか	インスタンスの無応答監視に対する振る舞いを指定します。 none: 無応答監視を行わない message: 無応答監視を行い、異常を検知した場合はメッセージを通知する failover: 無応答監視を行い、異常を検知した場合は自動縮退を行う デフォルトは、failoverです。
disk_check_interval	ディスクの異常監視の間隔時間(ミリ秒数)	ディスク障害を監視するため、disk_check_intervalに指定した間隔で異常監視を行います。ファイルが作成できなかった場合は、プライマリサーバの切り替えやスタンバイサーバの切り離しが必要となる異常が発生したと見なします。 1～2147483647の間の数値で指定してください。ディスクアクセス時間より大きい値を設定してください。 デフォルトは、heartbeat_intervalの設定値です。
disk_check_retry	ディスクの異常監視のリトライ回数(回数)	プライマリサーバの切り替えやスタンバイサーバの切り離しが必要となる異常を検知した場合のリトライ回数を指定します。

パラメータ	設定値	説明
		指定回数＋1回以上連続して異常を検知した場合に切り替えや切り離しを行います。 0～2147483647の間の数値で指定してください。 デフォルトのリトライ回数は、heartbeat_retryの設定値です。
disk_check_timeout (注3)	異常監視のタイムアウト時間 (秒数)	ディスク異常が発生した時点の次のdisk_check_intervalの開始時刻からタイムアウトによる異常と判断するまで許容する時間です。 スタンバイサーバでこのタイムアウトによるディスク異常検知時にスタンバイサーバの切り離しを行うには、 shutdown_detached_synchronous_standbyにonを設定してください。 デフォルトは2147483です。 0～2147483までの間の数値で指定可能です。
disk_check_max_threads (注3)	異常監視に利用するスレッド数の上限	ディスク監視用スレッド数の上限です。 1～2147483647までの間の数値で指定してください。 デフォルトはJVMが利用可能なプロセッサ数です。 マシンが利用可能なスレッド以上の値を設定するとシステムエラーになる可能性があります。監視プロセスとは別に、mc_ctl statusコマンドを実行する場合に、mc_ctl statusごとに監視プロセスと同じ数量のスレッドが一時的に利用されます。 disk_check_max_threadsを設定する際に、マシンのスレッド上限、利用予定のテーブル空間数、および同時に実行する可能性があるmc_ctl statusコマンドの数を考慮し、設定してください。
tablespace_directory_error_action	messageまたはfailover	テーブル空間の格納ディレクトリに異常を検知した場合の振る舞いを指定します。 message: メッセージを通知する failover: 自動縮退を行う デフォルトは、failoverです。
arbiter_alive_interval	Mirroring Controller裁定プロセスとの接続監視の間隔時間(ミリ秒数)	指定された間隔でMirroring Controller裁定プロセスに対し、ハートビートを行います。 1～2147483647の間の数値で指定してください。 デフォルトは16000ミリ秒です。 本パラメータは、裁定サーバを利用しない運用では設定不要です。
arbiter_alive_timeout	Mirroring Controller裁定プロセスとの接続監視のタイムアウト時間(秒数)	ハートビートが指定された秒数以上のあいだ応答がない場合には、Mirroring Controller裁定プロセスとの接続が切断されたと判断し、メッセージを出力して、再接続を試みます。 1～2147483647の間の数値で指定してください。 デフォルトは20秒です。 本パラメータは、裁定サーバを利用しない運用では設定不要です。

パラメータ	設定値	説明
arbiter_alive_retry	Mirroring Controller裁定プロセスとの接続監視のリトライ回数(回数)	Mirroring Controller裁定プロセスとのハートビートの異常検知をした場合の、ハートビートのリトライ回数を指定します。指定された回数以上、ハートビートの応答がない場合には、Mirroring Controller裁定プロセスとの接続が切断されたと判断します。 0～2147483647の間の数値で指定してください。 デフォルトは0回です。 本パラメータは、裁定サーバを利用しない運用では設定不要です。
arbiter_connect_interval	Mirroring Controller裁定プロセスとの接続の試行間隔(ミリ秒数)	Mirroring Controllerプロセスの起動時に接続が失敗した場合、またはMirroring Controller裁定プロセスとの切断時に、指定された間隔で再接続を試行します。 1～2147483647の間の数値で指定してください。 デフォルトは16000ミリ秒です。 本パラメータは、裁定サーバを利用しない運用では設定不要です。
arbiter_connect_timeout	Mirroring Controller裁定プロセスとの接続のタイムアウト時間(秒数)	Mirroring Controllerプロセスの起動時、またはMirroring Controller裁定プロセスとの切断時の再接続時に、指定された秒数以上のあいだ接続ができない場合は、Mirroring Controller裁定プロセスとの接続に失敗したと判断し、再接続を試みます。 1～2147483647の間の数値で指定してください。 デフォルトは20秒です。 本パラメータは、裁定サーバを利用しない運用では設定不要です。
fencing_command	'フェンシングコマンドのファイルパス名' [設定例] fencing_command = '/mc/fencing_dir/execute_fencing.sh'	異常と判断したデータベースサーバをフェンシングするフェンシングコマンドのパス名を指定します。 ファイルパス名は絶対パスで指定し、半角シングルクォートで囲んでください。 1024バイト未満の長さで指定してください。 本パラメータは、heartbeat_error_actionにcommandを設定した場合は、指定必須です。
fencing_command_timeout	フェンシングコマンドのタイムアウト時間(秒数)	指定された秒数以上のあいだ、コマンドの応答がない場合は、フェンシングに失敗したと判断し、フェンシングコマンドの実行プロセスに対して、シグナル(SIGTERM)を送信します。 1～2147483647の間の数値で指定してください。 デフォルトは20秒です。
arbitration_timeout	Mirroring Controller裁定プロセスでの裁定処理のタイムアウト時間(秒数)	OS/サーバの生死監視時間である裁定定義ファイルのfencing_command_timeout以上の値を指定する必要があります。 指定された秒数以上のあいだ、応答がない場合には、プライマリサーバの切り替えやスタンバイサーバの切り離しは行われません。そのため、手動での縮退操作を行ってください。

パラメータ	設定値	説明
		裁定サーバのarbitration.confにheartbeat_interval、heartbeat_timeout、heartbeat_retryの値を指定している場合は、裁定サーバの値を使用してarbitration_timeoutの時間を設計してください。 1～2147483647の間の数値で指定してください。 デフォルトは30秒です。 本パラメータは、裁定サーバを利用しない運用では設定不要です。
arbitration_command	'裁定コマンドのファイルパス名' [設定例] arbitration_command = '/mc/arbitration_dir/execute_arbitration_command.sh'	OS/サーバの生死監視において異常を検出した場合に実行する裁定コマンドのファイルパス名を指定します。ファイルパス名は絶対パスで指定し、半角シングルクォートで囲んでください。 1024バイト未満の長さで指定してください。 本パラメータは、heartbeat_error_actionにcommandを設定した場合は、指定必須です。
arbitration_command_timeout	裁定コマンドのタイムアウト時間(秒数)	指定された秒数以上のあいだ、裁定コマンドの応答がない場合は、裁定コマンドの実行に失敗したと判断し、裁定コマンドの実行プロセスに対して、シグナル(SIGTERM)を送信します。 1～2147483647の間の数値で指定してください。 デフォルトは30秒です。 本パラメータは、heartbeat_error_actionにcommandを設定した場合のみ、指定可能です。
shutdown_detached_synchronous_standby	onまたはoff	スタンバイサーバを切り離す際にスタンバイサーバのインスタンスを強制停止させるか否かを指定します。 on: インスタンスを停止する off: インスタンスを停止しない onを指定し、切り離し前コマンドを作成していた場合、切り離し前コマンドを実行した後にインスタンスの停止を行います。 デフォルトは、offです。
post_switch_command	'切り替え後コマンドのファイルパス名' [設定例] post_switch_command = '/mc/status_change/execute_post_switch.sh'	プライマリサーバのフェイルオーバ時に、新プライマリサーバが昇格した後にMirroring Controllerが呼び出すコマンドのパス名を指定します。 ファイルパス名は絶対パスで指定し、半角シングルクォートで囲んでください。 1024バイト未満の長さで指定してください。
post_attach_command	'組み込み後コマンドのファイルパス名' [設定例] post_attach_command = '/mc/status_change/execute_post_attach.sh'	スタンバイサーバがクラスタシステムに組み込まれた後にMirroring Controllerが呼び出すコマンドのパス名を指定します。 ファイルパス名は絶対パスで指定し、半角シングルクォートで囲んでください。 1024バイト未満の長さで指定してください。
pre_detach_command	'切り離し前コマンドのファイルパス名' [設定例]	スタンバイサーバがクラスタシステムから切り離される前にMirroring Controllerが呼び出すコマンドのパス名を指定します。

パラメータ	設定値	説明
	pre_detach_command = '/mc/status_change/execute_pre_detach.sh'	ファイルパス名は絶対パスで指定し、半角シングルクォートで囲んでください。 1024バイト未満の長さで指定してください。
status_change_command_timeout	状態遷移コマンドのタイムアウト時間(秒数)	切り替え後コマンド、組み込み後コマンド、および切り離し前コマンドのタイムアウト時間を指定します。指定された秒数以上のあいだ、コマンドの応答がない場合は、ステータス変更コマンドの実行プロセスに対して、シグナル(SIGTERM)を送信します。 タイムアウト時間は、1～2147483647の間の数値で指定します。 デフォルトは、20秒です。
enable_promote_on_os_and_admin_network_error (注4)	onまたはoff	onが指定された場合、管理用ネットワークに異常が発生し、さらにレプリケーションが切断された際に、プライマリサーバやデータベースが動作しているかを裁定サーバに確認するように依頼し、ダウンしていた場合はスタンバイサーバを昇格させます。 デフォルトは、offです。
check_synchronous_standby_names_validation	onまたはoff	postgresql.confのsynchronous_standby_namesパラメータがユーザーの誤操作により変更されたかを、Mirroring Controllerが運用中に定期的にチェックするか否かを指定します。 ただし、チェックを行うと、Mirroring Controllerがデータベースサーバのの CPUを冗長に使用し、SQL文を高頻度で実行することから、本パラメータを有効にすることは推奨できません。 デフォルトは、offです。
db_instance_ext_pq_conninfo	'データベースのインスタンスへ接続するlibpq接続SSLパラメータ'	Mirroring Controllerがデータベース接続時に追加するlibpqの接続パラメータをkey-valueの形式で指定します。指定可能な接続パラメータはSSLに関するパラメータです。パラメータはASCII文字で指定してください。 db_instance_ext_jdbc_conninfoにも、本パラメータに指定した接続パラメータを指定する必要があります。
db_instance_ext_jdbc_conninfo	'データベースのインスタンスへ接続するJDBC接続SSLパラメータ'	Mirroring Controllerがデータベース接続時に追加するJDBCの接続パラメータをURI形式で指定します。指定可能な接続パラメータはSSLに関するパラメータです。パラメータはASCII文字で指定してください。 db_instance_ext_pq_conninfoにも、本パラメータに指定した接続パラメータを指定する必要があります。

注1) postgresql.confファイルのsyslog_identパラメータの指定と同じプログラム名とすることで、Mirroring Controllerの出力内容と透過的に参照できるため、ログの参照が容易になります。

注2) OSのTCP接続のタイムアウト時間はカーネルパラメータtcp_syn_retriesにより決まります。remote_call_timeoutパラメータはOSのTCP接続のタイムアウト時間よりも短い値を設定しなければならないため、必要に応じていずれかのパラメータを変更してください。

注3) PH23295を含む緊急修正を適用することで利用できます。

注4) PH23520を含む緊急修正を適用することで利用できます。

OS/サーバの生死監視においてハートビート異常を検出した場合の動作を設定するheartbeat_error_actionパラメータに設定した値により、一部指定可能なパラメータが異なります。

表A.5 heartbeat_error_actionパラメータの設定値によるパラメータの指定可否

パラメータ	設定値			
	arbitration	command	message	fallback
arbiter_alive_interval	○	×	×	×
arbiter_alive_timeout	○	×	×	×
arbiter_alive_retry	○	×	×	×
arbiter_connect_interval	○	×	×	×
arbiter_connect_timeout	○	×	×	×
arbitration_timeout	○	×	×	×
arbitration_command	×	◎	×	×
arbitration_command_timeout	×	○	×	×
fencing_command	○	◎	○	×
fencing_command_timeout	○	○	○	×
shutdown_detached_synchronous_standby	○	○	×	×

◎:指定必須

○:指定可能

×:指定不可

A.4.2 裁定定義ファイル

Mirroring Controller裁定プロセスでの裁定および制御に関する情報をarbitration.confファイルに定義します。

表A.6 arbitration.confファイル

パラメータ	設定値	説明
port	Mirroring Controller裁定プロセスのポート番号	0 ～ 65535の範囲を超えて指定できません。他のソフトウェアと重ならないように設定してください。また、他のプログラムによって一時的に割り当てられる可能性のあるエフェメラルポートを指定しないでください。 データベースサーバのnetwork.confに指定する裁定サーバのポート番号は、本パラメータに指定したポート番号と同じ値を指定してください。
my_address	'データベースサーバのMirroring Controllerプロセスからの接続を受け付けるIPアドレス、またはホスト名' [設定例] my_address = ‘192.0.3.120’	データベースサーバのnetwork.confに指定する裁定サーバのIPアドレス、またはホスト名は、本パラメータに指定したIPアドレス、またはホスト名と同じ値を指定してください。 また、IPアドレスはIPv4、IPv6の指定が可能です。 半角シングルクォートで囲んで指定してください。
core_file_path	'コアファイルを出力するディレクトリ'	コアファイルを出力するディレクトリを半角シングルクォートで囲んで指定します。ASCII文字で指定してください。 本パラメータを省略すると、Mirroring Controller裁定プロセスの管理ディレクトリを指定したものとみなされます。

パラメータ	設定値	説明
syslog_facility	LOCAL0、LOCAL1、LOCAL2、LOCAL3、LOCAL4、LOCAL5、LOCAL6、LOCAL7のいずれかを指定します。	syslogへのログ取得が有効である場合、このパラメータの値がsyslogの"facility"に使用されます。 デフォルトは、LOCAL0です。
syslog_ident	'プログラム名'	システムログ内のMirroring Controller裁定プロセスのメッセージを特定するために使用するプログラム名を半角シングルクォートで囲んで指定します。半角空白以外のASCII文字で指定してください。 デフォルトは、'MirroringControllerArbiter'です。
fencing_command	'フェンシングコマンドのファイルパス名' [設定例] fencing_command = '/arbiter/fencing_dir/execute_fencing.sh'	異常と判断したデータベースサーバをフェンシングするフェンシングコマンドのファイルパス名を指定します。 ファイルパス名は絶対パスで指定し、半角シングルクォートで囲んでください。 1024バイト未満の長さで指定してください。
fencing_command_timeout	フェンシングコマンドのタイムアウト時間(秒数)	指定された秒数以上のあいだ、コマンドの応答がない場合は、フェンシングに失敗したと判断し、フェンシングコマンドの実行プロセスに対して、シグナル(SIGTERM)を送信します。 1～2147483647の間の数値で指定してください。 デフォルトは20秒です。
heartbeat_interval(注)	OS/サーバの生死監視の間隔時間(ミリ秒数)	指定された間隔でデータベースサーバの生死監視を行って、裁定を行います。 1～2147483647の間の数値で指定してください。 デフォルトは、データベースサーバのサーバ識別子.confに指定した値を使用します。 管理用ネットワークとの回線負荷の違いや縮退時間の短縮などを考慮して最適化を行う場合に指定します。
heartbeat_timeout	OS/サーバの生死監視のタイムアウト時間(秒数)	指定された秒数以上のあいだ、応答がない場合に、プライマリサーバ、またはスタンバイサーバのフェンシングが必要となる異常が発生したと見なします。 1～2147483647の間の数値で指定してください。 デフォルトは、データベースサーバのサーバ識別子.confに指定した値を使用します。 管理用ネットワークとの回線負荷の違いや縮退時間の短縮などを考慮して最適化を行う場合に指定します。
heartbeat_retry	OS/サーバの生死監視のリトライ回数(回数)	プライマリサーバ、またはスタンバイサーバのフェンシングが必要となる異常を検知した場合のリトライ回数を指定します。 指定回数＋1回以上連続して異常を検知した場合にフェンシングを行います。 0～2147483647の間の数値で指定してください。 デフォルトは、データベースサーバのサーバ識別子.confに指定した値を使用します。

パラメータ	設定値	説明
		管理用ネットワークとの回線負荷の違いや縮退時間の短縮などを考慮して最適化を行う場合に指定します。

注) 裁定サーバを使用した場合のOS/サーバの異常監視のパラメータのチューニングについては、"2.11.4 最適な異常監視による縮退運転を行うためのチューニング"を参照してください。

付録B　プライマリサーバとスタンバイサーバを同一サーバ内に構築する場合の補足事項

システムテストなどの目的のために、プライマリサーバとスタンバイサーバを擬似的に同一のサーバ内に構築することが可能です。ただし、性能および信頼性の観点からテスト以外の目的における使用は、推奨していません。したがって、この構成での運用は行わないでください。

なお、セットアップおよび運用操作は、プライマリサーバとスタンバイサーバを異なるサーバに構築する場合と同じ手順で行います。

以降では、プライマリサーバとスタンバイサーバを同一サーバ内に構築する場合の補足事項について、説明します。

注意

プライマリサーバとスタンバイサーバを同一サーバ内に構築する構成において、裁定サーバによる自動縮退を行う運用を設定しても、その効果を得ることはできません。

B.1　バックアップデータ格納先ディレクトリ

プライマリサーバとスタンバイサーバで、同じバックアップデータ格納先ディレクトリを使用しても問題ありません。

B.2　mc_ctlコマンドの実行方法

mc_ctlコマンドを実行するときには、操作対象のサーバを特定するために--local-serverオプションにサーバ識別子を指定してください。

以下は、network.confファイルに記載したサーバ“server1”のMirroring Controllerを起動する場合の例です。他のモードによるmc_ctlコマンドの操作も同様に--local-serverオプションを指定します。

network.confファイルに同一IPアドレスで異なるポート番号の2つのサーバ識別子を定義します。

例）

```
server1 192.0.2.100 27540
server2 192.0.2.100 27541
```

プライマリサーバおよびスタンバイサーバのどちらにおいても、ポート番号は他のソフトウェアと重ならないように設定してください。

自動切り替え/切り離しを有効化する場合

サーバ“server1”のMirroring Controllerを起動します。

例）

```
$ mc_ctl start -M /mcdir/inst1 --local-server server1
```

サーバ“server1”のMirroring Controllerを停止します。

例）

```
$ mc_ctl stop -M /mcdir/inst1 --local-server server1
```

自動切り替え/切り離しを有効化しない場合

サーバ“server1”のMirroring Controllerを起動します。

例）

```
$ mc_ctl start -M /mcdir/inst1 -F --local-server server1
```

サーバ“server1”のMirroring Controllerを停止します。

例）

```
$ mc_ctl stop -M /mcdir/inst1 --local-server server1
```

注意

systemd用のユニットファイルのExecStartおよびExecStopのmc_ctlオプションの指定には、--local-serverオプションを追加してください。

ExecStartおよびExecStopのmc_ctlオプションの指定方法は、“2.12 多重化したインスタンスとMirroring Controllerの自動起動・停止の設定”を参照してください。

付録C　ユーザーコマンド

ユーザーコマンドには、以下の3種類のコマンドがあります。

- フェンシングコマンド
- 裁定コマンド
- 状態遷移コマンド

各ユーザーコマンドについて、説明します。

C.1　フェンシングコマンド

形式

Mirroring ControllerプロセスまたはMirroring Controller裁定プロセスがフェンシングコマンドを呼び出す際の引数は以下です。

データベースサーバのフェンシングコマンドの場合

フェンシングコマンド名 フェンシングコマンドの実行契機 Mirroring Controllerが行う縮退操作 *サーバ識別子 サーバ識別子 センタ種別*

裁定サーバのフェンシングコマンドの場合

フェンシングコマンド名 フェンシングコマンドの実行契機 Mirroring Controllerが行う縮退操作 *サーバ識別子*

入力

データベースサーバのフェンシングコマンドの場合

フェンシングコマンドの実行契機

monitor: Mirroring Controllerプロセスの自動監視による異常検知

command: Mirroring Controllerのコマンド実行(mc_ctlコマンドのswitchモードまたはdetachモード)

Mirroring Controllerが行う縮退動作

switch: 切り替え

detach: 切り離し

サーバ識別子

フェンシングコマンドを呼び出したデータベースサーバのサーバ識別子

サーバ識別子

フェンシング対象のデータベースサーバのサーバ識別子

センタ種別

固定値（“primarycenter”が設定されます）

裁定サーバのフェンシングコマンドの場合

フェンシングコマンドの実行契機

monitor: Mirroring Controllerプロセスの自動監視による異常検知

command: Mirroring Controllerのコマンド実行(mc_ctlコマンドのswitchモードまたはdetachモード)

Mirroring Controllerが行う縮退動作

switch: 切り替え

detach: 切り離し

サーバ識別子

フェンシング対象のデータベースサーバのサーバ識別子

出力

復帰値

0: Mirroring Controllerによる縮退処理を継続します。

0以外: Mirroring Controllerによる縮退処理を中断します。

説明

入力のサーバ識別子を元にフェンシング対象のデータベースサーバを特定し、不安定なデータベースサーバをクラスタシステムから隔離する処理を実装します。

注釈

- フェンシングコマンドは、Mirroring Controller、またはMirroring Controller裁定プロセスを起動したOSユーザーで実行されます。そのため、特定のOSユーザーで実行する場合は、フェンシングコマンド内で実行ユーザーを意図するOSユーザーに変更してください。
- Mirroring ControllerまたはMirroring Controller裁定プロセスを起動したOSユーザーに対して、フェンシングコマンドに実行権限を付与する必要があります。実行権限がない場合は、縮退処理を中断します。
- セキュリティの観点から、意図しないOSユーザーにフェンシングコマンドを書き換えられて不当な動作が行われないように、必要に応じてアクセス権の設定を行ってください。
- フェンシングコマンドが0以外で復帰した場合、Mirroring Controllerは縮退処理を中断するため、利用者はサーバの状態を確認し、必要に応じて手動で切り替えや切り離しを行う必要があります。
- フェンシング対象のサーバが既にフェンシングが行われている状態の場合、フェンシング処理が異常終了しないように、事前に状態を確認してから、フェンシング処理を実行するようにしてください。
- フェンシングコマンドの実行中にタイムアウトが発生した場合、Mirroring Controllerはフェンシングコマンドを停止させ、エラーメッセージを出力して、縮退処理を中断します。

参考

フェンシングコマンドの実装方法の一つとして、OSやサーバを停止する方法があります。例えば、データベースサーバの電源を停止する場合は、標準ハードウェア規格であるIPMI(Intelligent Platform Management Interface)に対応したHW制御ボードが装備されている環境で、本ボードを制御するユーティリティを利用する方法があります。

IPMItoolを利用した、データベースサーバの電源を停止するフェンシングコマンドのサンプルスクリプトを、以下に提供しています。

サンプルのシェルスクリプトのパス

```
/インストールディレクトリ/share/mcarb_execute_fencing.sh.sample
```

C.2 裁定コマンド

形式

Mirroring Controllerプロセスが裁定コマンドを呼び出す際の引数は以下です。

```
裁定コマンド名 サーバ識別子 サーバ識別子 センタ種別
```

入力

サーバ識別子

裁定コマンドを呼び出したデータベースサーバのサーバ識別子

サーバ識別子

裁定対象のデータベースサーバのサーバ識別子

センタ種別

固定値（“primarycenter”が設定されます）

出力

復帰値

0: 裁定対象のデータベースサーバを異常と判断し、Mirroring Controllerによる縮退処理を継続します。

0以外:裁定対象のデータベースサーバを正常と判断し、Mirroring Controllerによる縮退処理を中断します。

説明

入力のサーバ識別子を元に裁定対象のデータベースサーバを特定し、サーバの状態を確認するための処理を実装します。

注釈

- 裁定コマンドは、Mirroring Controllerを起動したOSユーザーで実行されます。
- Mirroring Controllerを起動したOSユーザーに対して、裁定コマンドの実行権限を付与する必要があります。実行権限がない場合は、裁定コマンドは呼び出されず、縮退処理を中断します。
- 裁定コマンドの実行中にタイムアウトが発生した場合、Mirroring Controllerは裁定コマンドを停止させ、エラーメッセージを出力して、縮退処理を中断します。

C.3 状態遷移コマンド

状態遷移コマンドには、以下の3種類のユーザーコマンドが存在します。Mirroring Controllerによるデータベースサーバの状態遷移に伴う任意の実装を行うことができます。

- 切り替え後コマンド
- 切り離し前コマンド
- 組み込み後コマンド

C.3.1 切り替え後コマンド

形式

Mirroring Controllerプロセスが切り替え後コマンドを呼び出す際の引数は以下です。

```
切り替え後コマンド名 サーバ識別子 センタ種別
```

入力

サーバ識別子

切り替えが行われたデータベースサーバ（新プライマリサーバまたは新プライマリ候補サーバ）のサーバ識別子

センタ種別

固定値（“primarycenter”が設定されます）

出力

復帰値

なし

注釈

- 切り替え後コマンドは、Mirroring Controllerを起動したOSユーザーで実行されます。

- Mirroring Controllerを起動したOSユーザーに対して、切り替え後コマンドの実行権限を付与する必要があります。実行権限がない場合は、切り替え後コマンドは呼び出されません。
- 切り替え後コマンドの実行中にタイムアウトが発生した場合、Mirroring Controllerは切り替え後コマンドを停止させ、エラーメッセージを出力しますが、処理を継続します。

C.3.2 切り離し前コマンド

形式

Mirroring Controllerプロセスが切り離し前コマンドを呼び出す際の引数は以下です。

切り離し前コマンド名 サーバ識別子 サーバの役割 サーバ識別子 センタ種別

入力

サーバ識別子

切り離し前コマンドを呼び出したデータベースサーバのサーバ識別子

サーバの役割

切り離し前コマンドを呼び出したデータベースサーバのサーバの役割

primary : プライマリ

standby : スタンバイ

サーバ識別子

クラスタシステムからの切り離しが行われるスタンバイサーバのサーバ識別子

センタ種別

固定値（“primarycenter”が設定されます）

出力

復帰値

なし

注釈

- 切り離し前コマンドは、Mirroring Controllerを起動したOSユーザーで実行されます。
- Mirroring Controllerを起動したOSユーザーに対して、切り離し前コマンドの実行権限を付与する必要があります。実行権限がない場合は、切り離し前コマンドは呼び出されませんが、Mirroring Controllerはエラーメッセージを出力して、処理を継続します。
- 切り離し前コマンドの実行中にタイムアウトが発生した場合、Mirroring Controllerは切り離し前コマンドを停止させ、エラーメッセージを出力しますが、処理を継続します。

C.3.3 組み込み後コマンド

形式

Mirroring Controllerプロセスが組み込み後コマンドを呼び出す際の引数は以下です。

組み込み後コマンド名 サーバ識別子 サーバの役割 サーバ識別子 センタ種別

入力

サーバ識別子

組み込み後コマンドを呼び出したデータベースサーバのサーバ識別子

サーバの役割

組み込み後コマンドを呼び出したデータベースサーバのサーバの役割

primary : プライマリ

standby : スタンバイ

サーバ識別子

クラスタシステムへの組み込みが行われたスタンバイサーバのサーバ識別子

センタ種別

固定値（“primarycenter”が設定されます）

出力

復帰値

なし

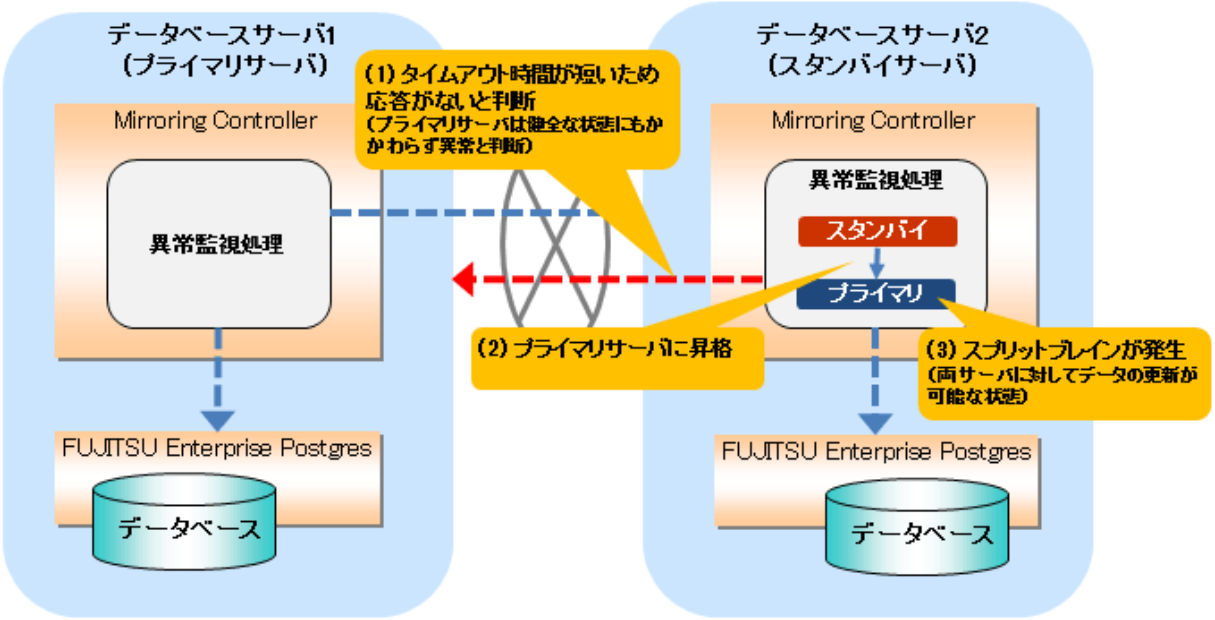
注釈

- 組み込み後コマンドは、Mirroring Controllerを起動したOSユーザーで実行されます。
- Mirroring Controllerを起動したOSユーザーに対して、組み込み後コマンドの実行権限を付与する必要があります。実行権限がない場合は、組み込み後コマンドは呼び出されません。
- 組み込み後コマンドの実行中にタイムアウトが発生した場合、Mirroring Controllerは組み込み後コマンドを停止させ、エラーメッセージを出力しますが、処理を継続します。

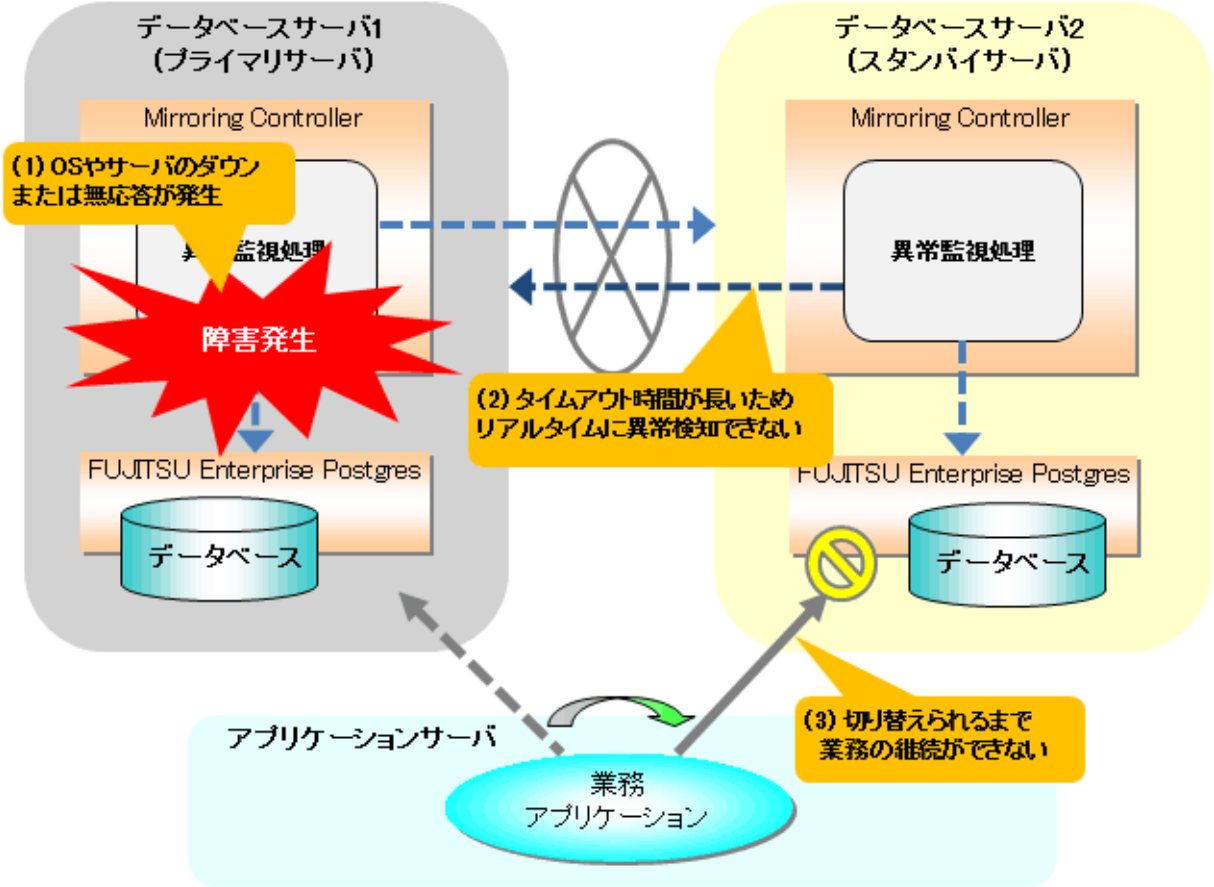
付録D　ハートビート異常で即時に自動縮退を行うを選択した場合の注意事項

OSやサーバの生死監視おいてハートビート異常で無条件に自動縮退を行う選択を行った場合、異常監視に関するチューニングが適切に行われていない場合には、以下のような事象が発生します。

● タイムアウト時間が短い場合



● タイムアウト時間が長い場合



OSやサーバのダウンおよび無応答による監視についての注意事項

OSやサーバのダウンおよび無応答は上記の通りタイムアウトによる監視を行っています。そのため、チューニングが正しく行われていないと、サーバが健全な状態にもかかわらず誤ってスプリットブレインを発生させる危険性があります。

スプリットブレインとは、一時的に両サーバがプライマリサーバとして動作してしまうために、両サーバでデータの更新業務が行われる現象をいいます。

スプリットブレインの検出方法

以下の条件の場合にスプリットブレインが発生していることが確認できます。

1. 両サーバでmc_ctlコマンドをstatusモードで実行した結果、両サーバの“host_role”が“primary”と出力される。かつ、
2. いずれかのサーバのシステムログに、以下のメッセージが出力されている。

```
プライマリサーバへの昇格処理が完了しました (MCA00062)
```

スプリットブレインからの復旧方法

以下の手順で行います。なお、新プライマリサーバは上記の検出方法の2)で確認されたサーバとなります。

1. 旧プライマリサーバおよび新プライマリサーバで動作しているすべてのアプリケーションを停止します。
2. データベースの調査と復旧を行います。
   新プライマリサーバに反映されていない更新結果を旧プライマリサーバのデータベースから調査して、必要に応じて新プライマリサーバに適用します。
3. 旧プライマリサーバのインスタンスおよびMirroring Controllerを停止します。
4. 手順1.で停止したアプリケーションを再開します。

5. 旧プライマリサーバを復旧します。
   “2.5 スタンバイサーバのセットアップ”を参照して、新プライマリサーバから旧プライマリサーバを新スタンバイサーバとして構築（セットアップ）します。

付録E　WebAdminで使用できない文字

WebAdminで使用できない文字を以下に示します。

|	&	;	$
%	@	'	"
¥'	¥"	<>	()
+(CR、ASCII 0x0d)	LF(ASCII 0x0a)	,	¥

付録F　障害調査情報の採取

環境構築や運用中に発生したトラブルの原因が判明しない場合、初期調査のための情報を採取します。

初期調査のための情報の採取方法について説明します。

pgx_fjqssinfコマンドを使用して、初期調査のための情報を採取してください。

参照

pgx_fjqssinfコマンドについては、“リファレンス”を参照してください。

索　引

[A]

arbitration.confファイル.....114

[C]

CPUアーキテクチャと製品に関する注意.....10

[I]

ICMPの設定.....14

[M]

mc_ctlコマンドの実行方法.....117
Mirroring Controller管理ディレクトリのリカバリ.....75,80
Mirroring Controller裁定プロセスの起動.....19,54
Mirroring Controller裁定プロセスの起動と停止.....54
Mirroring Controller裁定プロセスの自動起動・停止の設定.51
Mirroring Controller裁定プロセスの停止.....54
Mirroring Controllerの資源.....8
Mirroring Controllerの停止.....75,79
Mirroring Controllerのメッセージ監視.....59

[N]

network.confファイル.....103

[P]

postgresql.confファイル.....98,100

[あ]

アプリケーションの作成.....34
アプリケーションの接続先サーバの設定.....34
異常原因の特定とスタンバイサーバの再構築.....75,79
異常原因の特定とリカバリ.....75,80
インスタンスの起動と停止.....54
インストール.....15
運用の変更.....66

[か]

管理用/ログ転送用ネットワークの冗長化.....10
切り替え後コマンド.....121
切り替え発生後の縮退運転における運用操作.....73
切り離し発生後の縮退運転における運用操作.....78
切り離し前コマンド.....122
組み込み後コマンド.....122

[さ]

裁定コマンド.....120
裁定サーバでセットアップと操作を行うユーザー.....14
裁定サーバでの状態確認.....57
裁定サーバでの接続状態の確認.....34
裁定サーバの資源.....9
裁定サーバの設定.....16
裁定サーバのセットアップ.....16
裁定サーバのプロセス.....9
裁定サーバの保守.....66
裁定サーバを利用して自動縮退を行うデータベース多重化運用への変更.....70
裁定定義ファイル.....114
最適な異常監視による縮退運転を行うためのチューニング..36
サーバ識別子.confファイル.....106
サーバ定義ファイル.....106
サーバの保守.....61
システムの時刻合わせ.....14
自動切り替えに失敗した場合の対処.....81
自動切り離しに失敗した場合の対処.....81
縮退運転になった場合の対処.....73
縮退運転中の異常時の対処.....80
状態遷移コマンド.....121
スタンバイサーバで設定するパラメータ.....100
スタンバイサーバでの参照系業務.....6
スタンバイサーバでの問い合わせを安定させるためのチューニング.....35
スタンバイサーバでの問い合わせを安定させるためのチューニング（プライマリサーバで頻繁に更新する場合）.....35
スタンバイサーバに転送するトランザクションログの暗号化...13
スタンバイサーバのMirroring Controllerの起動.....30
スタンバイサーバのインスタンスの作成・設定・登録.....29
スタンバイサーバの組込み.....77,80
スタンバイサーバの手動切り離し.....58
スタンバイサーバのセットアップ.....27
スタンバイサーバの停止を伴う変更.....67
スタンバイサーバのデータベース多重化運用のセットアップ 28
スタンバイサーバの認証.....12
ストリーミングレプリケーションの状態確認.....32
すべてのデータベースサーバまたはインスタンスが停止した場合の対処.....82
接続状態の確認.....33
セットアップ.....14
セットアップの準備作業.....16

[た]

多重化したインスタンスとMirroring Controllerの自動起動・停止の設定.....49
単体のサーバでの運用からデータベース多重化運用への変更.....67
チューニング.....35
データベースサーバでセットアップと操作を行うユーザー.....14
データベースサーバでの状態確認.....56
データベースサーバでの接続状態の確認.....33
データベースサーバのサーバ定義ファイル.....106
データベースサーバの準備作業.....16
データベースサーバのプロセス.....9
データベース多重化運用から単体のサーバでの運用への変更.....68
データベース多重化運用情報のバックアップ.....52
データベース多重化運用でのアンインストール.....71
データベース多重化運用とは.....1
データベース多重化運用による監視.....4
データベース多重化運用の異常時の対処.....73
データベース多重化運用の概要.....1
データベース多重化運用のシステム構成.....6
データベース多重化運用の状態確認.....56
データベース多重化運用のセットアップ.....14
データベース多重化運用の操作.....54

データベース多重化運用を安定させるためのチューニング. 35
データベース多重化におけるセキュリティ..... 11
データベースのバックアップ運用.....52
同期スタンバイサーバで参照系の業務を行う場合..... 6
動作確認..... 35

[な]

ネットワーク定義ファイル.....103

[は]

バックアップ運用..... 52
バックアップデータ格納先ディレクトリ..... 117
バックアップ用のディスクの準備.....16
パラメータ..... 98
パラメータの変更.....71
ハートビート異常検出時の操作..... 59
ハートビート異常検出時の動作の決定.....10
ハートビート異常で即時に自動縮退を行うを選択した場合の注意事項..... 124
フェンシングコマンド..... 119
プライマリサーバで行う主業務を優先する場合.....6
プライマリサーバで設定するパラメータ.....98
プライマリサーバとスタンバイサーバを同一サーバ内に構築する場合の補足事項..... 117
プライマリサーバのMirroring Controllerの起動.....27
プライマリサーバのインスタンスの作成・設定・登録..... 23
プライマリサーバの切り戻し.....77
プライマリサーバの手動切り替え..... 58
プライマリサーバのセットアップ..... 19
プライマリサーバのデータベース多重化運用のセットアップ. 19
保守停止..... 65

[や]

ユーザーコマンド..... 119
ユーザーの誤操作からのリカバリ.....86

[ら]

ローリングアップデート.....61