



EDB

Barman Guide

リリース 2.19.0

EDB

2022 年 04 月 30 日

目次

第 1 章 Introduction	3
第 2 章 Before you start	5
第 3 章 Design and architecture	7
3.1 Barman をインストールする場所	7
3.2 1 つの Barman、多数の PostgreSQL サーバー	8
3.3 ストリーミングバックアップと rsync / SSH	9
3.4 標準のアーカイビング、WAL ストリーミング、またはその両方	9
3.5 バックアップの 2 つの典型的なシナリオ	10
3.5.1 シナリオ 1：ストリーミングプロトコルを介したバックアップ	10
3.5.2 シナリオ 2：rsync / SSH を介したバックアップ	11
第 4 章 System requirements	15
4.1 バックアップの要件	15
4.2 リカバリの要件	15
第 5 章 Installation	17
5.1 RPM パッケージソフトを使用した RedHat / CentOS へのインストール	17
5.2 パッケージソフトを使用した Debian / Ubuntu へのインストール	18
5.3 パッケージソフトを使用した SLES へのインストール	18
5.4 ソースからのインストール	19
5.5 PostgreSQL クライアントバイナリ	19
第 6 章 Upgrading Barman	21
6.1 Barman 2.10 からのアップグレード	21
6.2 Barman 2.X からのアップグレード (2.8 より前)	21
6.3 Barman 1.X からのアップグレード	21
第 7 章 Configuration	23
7.1 オプションの範囲	23
7.2 設定の例	24
第 8 章 Setup of a new server in Barman	27
8.1 サーバー構成ファイル	27
8.2 WAL ストリーミング	28
8.2.1 複製スロット	29
8.2.2 WAL ストリーミングを設定する方法	29
8.2.3 リカバリを伴う部分的な WAL ファイルの制限	30
8.3 archive_command を介した WAL アーカイビング	30
8.3.1 barman-wal-archive を介した WAL アーカイビング	31
8.3.2 rsync / SSH を介した WAL アーカイビング	31

8.4	WAL アーカイビング設定の検証	32
8.5	ストリーミングバックアップ	33
8.6	rsync / SSH を使用したバックアップ	33
8.7	Windows ベースのサーバーをセットアップする方法	34
第 9 章	準備手順	35
9.1	PostgreSQL 接続	35
9.2	PostgreSQL WAL のアーカイビングとレプリケーション	36
9.3	PostgreSQL ストリーミング接続	37
9.4	SSH 接続	38
9.5	postgres ユーザの SSH 設定	38
9.6	barman ユーザの SSH 設定	38
9.7	PostgreSQL から Barman へ	38
9.8	Barman から PostgreSQL へ	39
第 10 章	General commands	41
10.1	cron	41
10.2	diagnose	42
10.3	list-servers	42
第 11 章	Server commands	43
11.1	archive-wal	43
11.2	backup	43
11.3	check	44
11.4	generate-manifest	44
11.5	get-wal	44
11.6	list-backups	46
11.7	rebuild-xlogdb	46
11.8	receive-wal	46
11.8.1	receive-wal プロセス管理	46
11.8.2	複製スロット管理	47
11.9	replication-status	47
11.10	show-servers	48
11.11	status	48
11.12	switch-wal	48
11.13	verify	48
第 12 章	Backup commands	49
12.1	バックアップ ID のショートカット	49
12.2	check-backup	49
12.3	delete	50
12.4	keep	50
12.5	list-files	51
12.6	recover	51
12.6.1	リモートリカバリ	52
12.6.2	表領域の再マッピング	53
12.6.3	ポイントインタイムリカバリ	53
12.6.4	Barman サーバーから WAL を取得する	54

12.7	show-backup	55
第 13 章 Features in detail		57
13.1	バックアップ機能	57
13.1.1	増分バックアップ	57
13.1.2	帯域幅の使用を制限する	58
13.1.3	ネットワーク圧縮	59
13.1.4	並行バックアップとスタンバイからのバックアップ	59
13.1.5	スタンバイからのバックアップに関する現在の制限	60
13.1.6	即時チェックポイント	61
13.1.7	ローカルバックアップ	61
13.2	アーカイブ機能	62
13.2.1	WAL 圧縮	62
13.2.2	同期 WAL ストリーミング	62
13.3	カタログ管理機能	64
13.3.1	最小の冗長性の安全性	64
13.3.2	保持ポリシー	64
13.3.3	スコープ	65
13.3.4	仕組み	65
13.3.5	設定と構文	65
13.4	フックスクリプト	66
13.4.1	バックアップスクリプト	67
13.4.2	バックアップ削除スクリプト	68
13.4.3	WAL アーカイブスクリプト	69
13.4.4	WAL 削除スクリプト	69
13.4.5	回復スクリプト	70
13.5	カスタマイズ	70
13.5.1	ファイルディレクトリをロックする	70
13.5.2	バイナリ経路	71
13.6	クラスター管理システムとの統合	71
13.7	並列ジョブ	71
13.8	地理的冗長性	72
13.8.1	同期情報	72
13.8.2	設定	73
13.8.3	ノードの同期	73
13.8.4	手動同期	73
第 14 章 Barman client utilities (barman-cli)		75
14.1	インストール	75
第 15 章 Barman client utilities for the Cloud (barman-cli-cloud)		77
15.1	インストール	78
15.2	barman-cloud フックスクリプト	78
15.3	クラウドプロバイダーの選択	79
15.4	プロバイダーごとの特異性	79
15.4.1	Google Cloud Storage	79
15.4.2	セットアップ	79
15.4.3	使用法	79

第 16 章 Troubleshooting	81
16.1 Barman インストールの診断	81
16.2 ヘルプのリクエスト	81
16.2.1 バグを提出する	81
第 17 章 The Barman project	83
17.1 サポートとスポンサーの機会	83
17.2 Barman への貢献	83
17.3 著者	84
17.4 リンク	84
17.5 ライセンスと貢献	85
第 18 章 Feature matrix	87

Barman（バックアップ リカバリー Manager）は、Python で記述された PostgreSQL サーバーの災害リカバリ用のオープンソースの管理ツールです。組織は、ビジネスクリティカルな環境でマルチプルのサーバーのリモートバックアップを実行して、リスクを軽減し、リカバリフェーズ中の DBA をヘルプできます。

Barman は GNU GPL 3 の下で配布され、PostgreSQL project のプラチナスポンサーである EnterpriseDB によって管理されています。

重要: このマニュアルは、理論的な災害リカバリの概念に精通しており、物理的バックアップと災害リカバリに関する PostgreSQL の基礎を理解していることを前提としています。詳細については、以下の「スタートする前に」セクションを参照してください。

第1章 Introduction

完璧な世界では、バックアップの必要はありません。ただし、特にビジネス環境では、「予期しない」事態が発生した場合に備えておくことが重要です。データベースのシナリオでは、予期しないものは次のいずれかの形式を取ります。

- データ破損
- システム障害（ハードウェア障害を含む）
- ヒューマンエラー
- 自然災害

このような場合、ICT マネージャまたは DBA は、インシデントを修正し、できるだけ短時間でデータベースを回復する必要があります。通常、この分野を「災害リカバリ**」と呼び、より広くは「ビジネス継続性」と呼びます。

ビジネスの継続性の中で、Wikipedia で定義されている 2 つの基本的な指標に精通することが重要です。

- ****Recovery Point Objective (RPO)**** : 「メジャーインシデントのために IT サービスからデータが失われる可能性のある最大対象期間」
- ****Recovery Time Objective (RTO)**** : 「ビジネス継続性の中断に関連する容認できない結果を回避オーダーに、災害（または中断）後にビジネスプロセスを復元する必要がある目標期間とサービスレベル」

簡単に言うと、RPO は失われる可能性のあるデータの最大量を表し、RTO はサービスに許容できる最大のダウンタイムを表します。

当然、私たちは皆、**RPO=0**（* “ゼロデータロス”）および **RTO=0**（ゼロダウンタイム*、ユートピア）を望んでいます-それが私たちの祖母のレシピウェブサイトであっても。実際には、慎重なコスト分析フェーズにより、ビジネス継続性の要件を決定できます。

幸いなことに、**Barman** と **PostgreSQL** で構成されるオープンソーススタックを使用すると、同期的ストリーミングレプリケーションのおかげで **RPO = 0** を実現できます。RTO は、****repmgr**** のような高可用性ソリューションの焦点です。したがって、Barman と repmgr を統合することにより、RTO をほぼゼロに劇的に削減できます。

EnterpriseDB での経験に基づいて、Barman および repmgr を使用した PostgreSQL オープンソースクラスターは、適切に構成および監視されていれば、1 年間で 99.99 % 以上の稼働率を容易に達成できることを確認できます。

いずれにしても、実際のツールではなく、災害リカバリに関連する文化的側面をより重視することが重要です。人間のいない道具は役に立たない。

Barman の私たちの使命は、災害リカバリの文化をプロモートすることです。

- バックアップ手順に焦点を当てる

- リカバリ手順にさらに焦点を当てる
- PostgreSQL のクラッシュリカバリ、バックアップ、Point-In-Time-Recovery、およびチームメンバーのレプリケーションに関する強力な理論的および実践的な概念に関する教育とトレーニングに依存しています。
- バックアップのテスト（テスト済みのバックアップのみが有効であると見なすことができます）を手動または自動（バーマンのフックスクリプトで創造的に！）で促進します。
- devops チームのすべてのメンバーによるリカバリ手順の定期的な実践を促進します（はい、開発者も、システム管理者や DBA だけでなく）
- 3〜6 か月ごとにチームと定期的にスケジュールされた訓練と災害リカバリシミュレーションを要請する PostgreSQL と Barman の継続的なモニタリングに依存しており、異常を迅速に特定できます。

さらに、災害が発生したとき（はい、いつ）に備えて、自分とチームを準備するためにできる限りのことを行ってください。

- 金曜日の夕方になります。おそらく、オフィスを出ようとしています。
- 休暇中（世界中のクルーズの真ん中）になり、他の誰かがそれに対処しなければならないときです。
- それは確かにストレスになるだろう。
- 使用可能な最後のバックアップが有効であることを確信できなかったことを後悔します。
- 回復にどれくらいの時間がかかるかわからない限り、毎秒は永遠に思えます。

準備して、怖がらないでください。

2011 年、これらの目標を念頭に置いて、2ndQuadrant は Barman の開発を開始しました。これは、現在 PostgreSQL で最も使用されているバックアップツールの 1 つです。Barman は、「バックアップリカバリーマネージャー」の頭字語です。

現在、Barman は Linux および Unix オペレーティングシステムでのみ動作します。

第2章 Before you start

Barman の使用をスタートする前に、PostgreSQL と、基本バックアップ、WAL アーカイビングなどの物理的バックアップ、Point-In-Time-Recovery およびレプリケーションに関する概念に精通することが基本です。

以下に、読むことをお勧めするリソースの非網羅的なリストを示します。

PostgreSQL 文書：

- [SQL Dump¹](#)
- [File System Level Backup](#)
- [Continuous Archiving and Point-in-Time Recovery \(PITR\)](#)
- [Reliability and the Write-Ahead Log](#)

ブック：

- [PostgreSQL 10 Administration Cookbook](#)

これらのトピックに関する専門的なトレーニングは、これらの概念を学習する別の効果的な方法です。年中いつでも、EnterpriseDB などの PostgreSQL 企業が提供する多くのコースを世界中で利用できます。

¹ ロジカルバックアップと物理的バックアップの違い、つまり `pg_dump` と Barman などのツールの違いを理解することが重要です。

第3章 Design and architecture

3.1 Barman をインストールする場所

Barman の基盤の 1 つは、ネットワークを介してデータベースサーバからリモートで操作できることです。

理論的には、Barman サーバーを、PostgreSQL サーバーから数千マイル離れた世界の別のパートのデータセンターに配置することができます。現実的には、Barman サーバーを PostgreSQL サーバーから遠ざけたくないので、バックアップとリカバリの両方の時間を制御できます。

Barman をセットアップするための「すべてのサイズに対応する」方法はありませんが、特に従うことをお勧めするいくつかの推奨事項があります。特に：

- Barman を専用サーバーにインストールする PostgreSQL サーバーと同じストレージを共有しないでください Barman をモニタリングインフラストラクチャ¹と統合します
- 稼動環境に展開する前にすべてをテストする

災害リカバリアーキテクチャのモデリングをスタートする合理的な方法は、次のとおりです。

PostgreSQL および Barman に関して、次のようないくつかの可能なアーキテクチャをデザインします。1. 同じデータセンター 2. 同じ大都市圏の異なるデータセンター 3. 異なるデータセンター - 各仮説の長所と短所を詳しく説明する - 費用対効果分析により、システムの単一障害点（SPOF）を評価する - make を下し、初期ソリューションを実装する

とはいえ、非常に一般的な Barman のセットアップは、PostgreSQL サーバーと同じデータセンターにインストールすることです。この場合、単一障害点はデータセンターです。幸いなことに、このような SPOF のインパクトは、バックアップ層の数を増やすために Barman が提供する 2 つの機能のおかげで軽減できます。

1. 地理的冗長性（Barman 2.6 で導入）
2. フックスクリプト

地理的冗長性により、異なるデータセンター/可用性ゾーンにある Barman インスタンスに依存して、ソース Barman サーバーのコンテンツ全体を同期できます。さらに、地理的冗長性はグローバルレベルだけでなくサーバーレベルでも Barman で設定できるため、一部のサーバーがローカルの PostgreSQL サーバーに直接接続され、他のサブセットがサブセットをバックアップする Barman のハイブリッドインストールを作成できます異なる Barman インストールの（クロスサイトバックアップ）。下の図 Barman { プライマリ - インストール } は、2 つのアベイラビリティゾーン（1 つはヨーロッパにあり、もう 1 つは米国にある）を Barman でローカル PostgreSQL。パッシブ）rsync / SSH 経由の多層バックアップ。地理的冗長性の詳細については、特定のセクションをご覧ください。

¹ Nagios / Barman との統合は、Barb の最も重要な機能の 1 つであり真の命の恩人である `barman check --nagios` コマンドのおかげで簡単です。

Backup architecture for PostgreSQL

Example of geographical redundancy

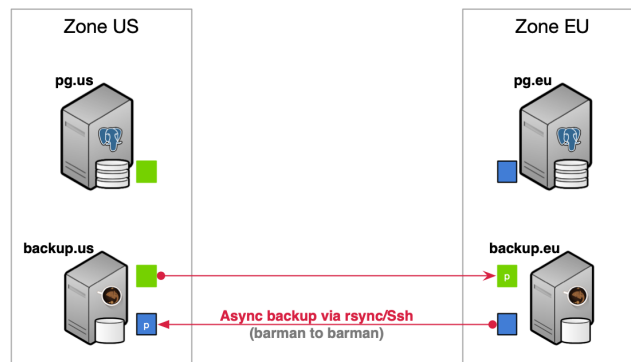


図 1: An example of architecture with geo-redundancy

代わりにフックスクリプトを使用することで、Barman のバックアップを、tar 経由のテープや Amazon クラウドの S3 バケットのような場所など、さまざまなメディアにエクスポートできます。

決定は永遠に続くものではないことを忘れないでください。この方法でスタートし、時間の経過とともに最適なソリューションに適応できます。ただし、スタートからシンプルに保つようにしてください。

3.2 1つの Barman、多数の PostgreSQL サーバー

Barman が最初に導入した別の関連機能は、マルチプルのサーバーのサポートです。Barman は、異なるバージョンであっても、マルチプルの PostgreSQL インスタンスからのバックアップデータを一元的に保存できます。[^recver]

[^]recver は、「リカバリの要件」セクションで詳しく説明されているように、リカバリに適用されます。

その結果、PostgreSQL サーバーが中央の Barman サーバーを中心にローテートする「スタースキーマ」を形成する複雑な災害リカバリアーキテクチャをモデルできます。

すべてのアーキテクチャは、独自の方法で意味をなします。実際の実験とテストに基づいて、あなたに共鳴するもの、最も重要なのはあなたが信頼するものを選択してください。

これ以降、簡単にするために、このガイドでは基本的なアーキテクチャを想定しています。

- 1 つの PostgreSQL インスタンス（ホスト名前 pg）
- Barman を使用した 1 つのバックアップサーバ（ホスト名前 Barman）

3.3 ストリーミングバックアップと rsync / SSH

従来、Barman は常に物理的バックアップ操作に `rsync` を利用して、SSH 経由でリモートで操作していました。バージョン 2.0 では、`pg_basebackup` を介した、バックアップ操作の PostgreSQL のストリーミングレプリケーションプロトコルのネイティブサポートが導入されています。

2

これら 2 つの方法のうちの 1 つを選ぶことはあなたがする必要がある make です。

一般的に、Barman 2.0 以降、ストリーミングレプリケーションを介したバックアップは、PostgreSQL 9.4 以降の推奨セットアップです。さらに、テーブルスペースを使用しない場合は、PostgreSQL 9.2 以降でストリーミングバックアップを使用できます。

重要: Barman は透過的に `pg_basebackup` を利用するため、増分バックアップ、並列バックアップ、重複排除、ネットワーク圧縮などの機能は現在利用できません。この場合、帯域幅制限には、`rsync` を介した従来のメソッドと比較して、いくつかの制限があります。

`rsync` / SSH を介した従来のバックアップは、8.3 以降の PostgreSQL のすべてのバージョンで使用でき、`pg_basebackup` の制限が発生するすべてのケースで推奨されます（例、増分バックアップと重複排除の恩恵を受けることができる非常にラージデータベース）。

ストリーミングバックアップを推奨する理由は、経験に基づいて、従来のバックアップよりもセットアップが簡単だからです。また、ストリーミングバックアップを使用すると、Windows³ で PostgreSQL サーバーをバックアップでき、Docker での作業が楽になります。

3.4 標準のアーカイビング、WAL ストリーミング、またはその両方

PostgreSQL の Point-In-Time-Recovery では、xlog または WAL ファイルとも呼ばれるトランザクションログをベースバックアップと一緒に保存する必要があります。

従来、Barman は PostgreSQL の `archive_command`（通常は `rsync` / SSH 経由、現在は `barman-cli` パッケージの `barman-wal-archive` 経由）による標準の WAL ファイル SHIPPING をサポートしています。このメソッドでは、PostgreSQL が新しい WAL ファイルに切り替えたときにのみ WAL ファイルがアーカイブされます。シンプルにするために、これは通常 16MB のデータ変更ごとに発生します。

Barman 1.6.0 は、`pg_receivewal`（PostgreSQL 10 より前の `pg_receivexlog` と呼ばれる）を介したトランザクションログアーカイビングの追加メソッドとして、PostgreSQL サーバー 9.2 以降の WAL ファイルのストリーミングを導入しています。WAL ストリーミングは、データロスリスクを軽減し、RPO をほぼゼロの値まで引き下げます。

Barman 2.0 では、PostgreSQL サーバー 9.4 以降でのレプリケーションスロットのサポートが導入されているため、WAL ストリーミングのみの構成が可能です。さらに、PostgreSQL 9.5（またはそれ以降）クラス

² PostgreSQL バージョンが Barman でのストリーミングレプリケーションバックアップをサポートする「機能マトリクス」をチェックインします。

³ Windows 上の PostgreSQL サーバーのバックアップは可能ですが、継続的インテグレーションシステムのパートではないため、まだ実験段階です。詳細については、「Windows ベースのサーバーのセットアップ方法」セクションを参照してください。

ターで Barman を同期的 WAL レシーバとして追加し、**データロスゼロ**（RPO = 0）を達成できるようになりました。

場合によっては選択の余地がなく、従来のアーカイブングの使用を余儀なくされます。その他では、両方を使用するか、WAL ストリーミングのみを使用するかを選択できます。しない理由がない限り、最大の信頼性と堅牢性のために両方のチャンネルを使用することをお勧めします。

3.5 バックアップの2つの典型的なシナリオ

生活を楽に make オーダーに、Barman の特定の PostgreSQL サーバーの 2 つの最も典型的なシナリオを以下にまとめます。

これは、Barman でバックアップすることを決定するすべてのサーバーに対して make 必要がある決定であることを覚えておいてください。これは、同じインストール内で異種セットアップを使用できることを意味します。

前述したように、PostgreSQL サーバー（pg）と Barman サーバー（backup）についてのみ心配します。ただし、実際には、アーキテクチャに repmgr、pgBouncer、Nagios / Icinga などの他のテクノロジーが含まれている可能性があります。

3.5.1 シナリオ 1：ストリーミングプロトコルを介したバックアップ

PostgreSQL 9.4 以降を使用しており、データベースが一般的なユースケースシナリオに該当する場合、ストリーミングバックアップインストールを決定することになります。以下の図 `ref {scenario1- デザイン}` を参照してください。

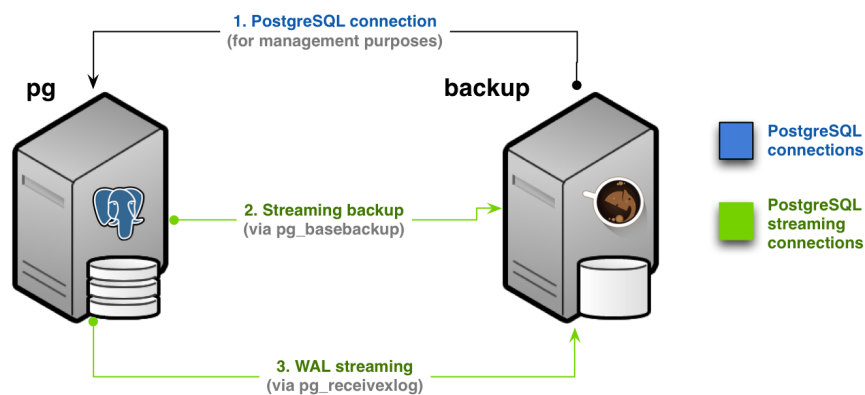


図 2: Streaming-only backup (Scenario 1)

このシナリオでは、以下を構成する必要があります。

1. 管理、調整、モニタリングを目的とした PostgreSQL への標準接続 2. `pg_basebackup`（ベースバックアップ操作）と `pg_receivewal`（WAL ストリーミング用）の両方で使用されるストリーミングレプリケーション接続

Barman の用語では、このセットアップは「ストリーミング専用」セットアップとして知られています。バックアップおよびアーカイビング操作に SSH 接続を必要としないためです。これは、Docker 環境に特に適しており、非常に実用的です。

ただし、前述のように、標準のアーカイビングを構成し、より堅牢なアーキテクチャを実装することもできます-以下の図 `ref {scenario1b-design}` を参照してください。

Backup architecture for PostgreSQL

Scenario 1b - Streaming backup with fallback WAL archiving

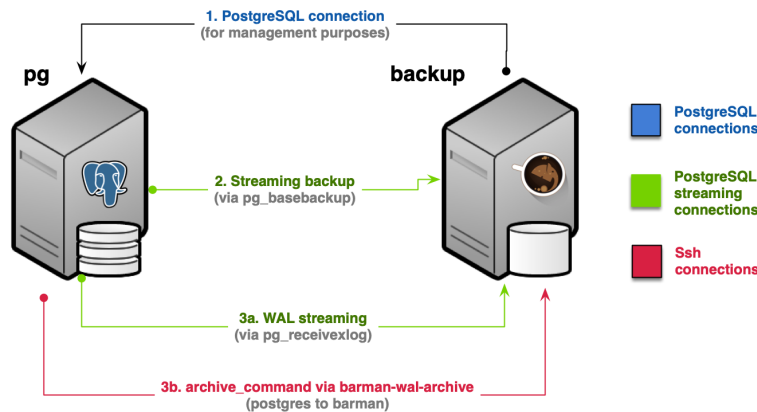


図 3: Streaming backup with WAL archiving (Scenario 1b)

この代替アプローチには以下が必要です。

- PostgreSQL サーバー上の `postgres` ユーザが Barman サーバー上の PostgreSQL ユーザとして接続できるようにする追加の SSH 接続
- PostgreSQL の PostgreSQL は、WAL ファイルを Barman に出荷するように構成されます

このアーキテクチャは、テーブルスペースを使用しない PostgreSQL 9.2 / 9.3 ユーザーも利用できます。

3.5.2 シナリオ 2: `rsync` / SSH を介したバックアップ

SSH を介した `rsync` の従来のセットアップは、次の場合にのみ使用可能なオプションです。

PostgreSQL サーバーバージョン 8.3、8.4、9.0 または 9.1 - テーブルスペースを使用している PostgreSQL サーバーバージョン 9.2 または 9.3 - 増分バックアップ、並列バックアップ、重複排除 - バックアップ中のネットワーク圧縮 - テーブルスペースを含む、帯域幅使用のより詳細な制御

このシナリオでは、以下を構成する必要があります。

1. 管理、調整、およびモニタリングのための PostgreSQL への標準接続
2. Barman サーバー上の `barman` ユーザが PostgreSQL サーバー上の `postgres` ユーザとして接続できるようにする `rsync` が使用するベースバックアップ操作の SSH 接続
3. PostgreSQL の `archive_command` が使用する WAL アーカイビング用の SSH 接続。これにより、PostgreSQL サーバーの `postgres` ユーザが Barman サーバーの `barman` ユーザとして接続できます。

Backup architecture for PostgreSQL

Scenario 2 - Traditional Barman setup

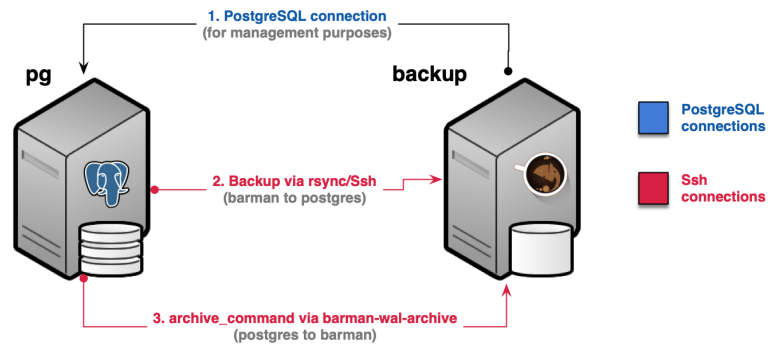


図 4: Scenario 2 - Backup via rsync/SSH

PostgreSQL 9.2 以降では、WAL ストリーミングに使用されるストリーミングレプリケーション接続を追加して、RPO を大幅に削減できます。このより堅牢な実装は、図 `ref {scenario2b-design}` に示されています。

< ! - TODO リカバリ用のアーキテクチャに関するセクションを追加しますか？ ->

Backup architecture for PostgreSQL

Scenario 2b - Traditional Barman setup with WAL streaming

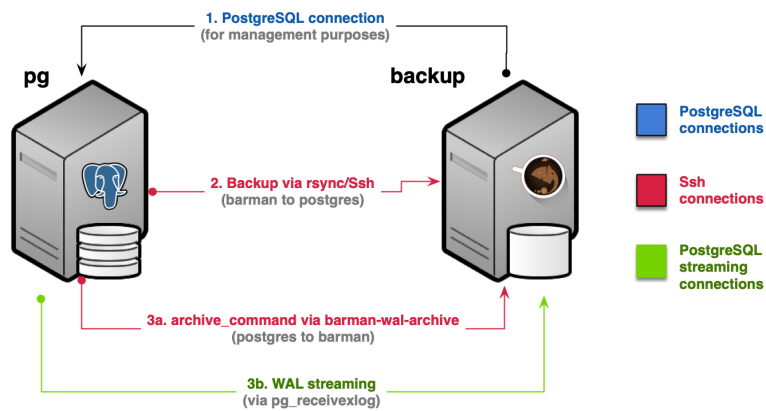


図 5: Backup via rsync/SSH with WAL streaming (Scenario 2b)

第4章 System requirements

- Linux / Unix Python >= 3.4 Python モジュール：
- argcomplete
- psycopg2 >= 2.4.2
- python-dateutil
- セットアップツール PostgreSQL >= 8.3
- rsync >= 3.0.4 (PostgreSQL >= 9.2 のオプション)

重要: RedHat Enterprise Linux、CentOS、および Scientific Linux のユーザーは、[Extra Packages Enterprise Linux \(EPEL\) repository](#) をインストールする必要があります。

注釈: Python 2.6 および 2.7 のサポートは非推奨であり、将来のリリースでは廃止されます。PostgreSQL <9.4 のサポートは非推奨であり、将来のリリースでは廃止されます。

4.1 バックアップの要件

Barman サーバーの最も重要な要件は、使用可能なディスクスペースです。クラスターのサイズ、1日に生成される WAL ファイルの数、バックアップの頻度、および保持ポリシーに基づいて、必要なディスクスペースを計画することをお勧めします。

公式にサポートしているファイルシステムは XFS と Ext4 ですが、ZFS や NFS を含むさまざまなファイルシステムに Barman を展開するユーザーを認識しています。

4.2 リカバリの要件

Barman では、ローカル（Barman が存在する場所）またはリモート（別のサーバー）で PostgreSQL インスタンスを回復できます。

リモートリカバリは、Barman を使用して PostgreSQL サーバーを復元する最も一般的な方法です。

いずれにしても、同じ [requirements for PostgreSQL's Log shipping and Point-In-Time-Recovery](#) apply :

- 同一のハードウェアアーキテクチャ PostgreSQL の同一のメジャーバージョン

一般に、保守が容易であるため、オリジナルのサーバーと可能な限り同一ではないにしても、可能な限り類似したリカバリ環境を作成することを強くお勧めします^{**}。例、同じオペレーティングシステム、同じ PostgreSQL バージョン、同じディスクレイアウトなどを使用することをお勧めします。

さらに、各 PostgreSQL サーバー専用のリカバリ環境は、オンデマンドでも、チームの災害リカバリ文化を育むことができます。リカバリ操作を実行し、それらに精通することにより、予期しないことが発生した場合に備えることができます。

当社の経験に基づいて、指定されリカバリ環境は実際の障害状況でのストレスのインパクトを軽減し、したがってリカバリ操作の効果を高めます。

最後に、例 NTP を使用して、サーバー間で時刻を同期することが重要です。

第5章 Installation

重要: Barman をインストールする推奨方法は、GNU/Linux ディストリビューションで利用可能なパッケージソフトを使用することです。

5.1 RPM パッケージソフトを使用した RedHat / CentOS へのインストール

Barman は、RPM パッケージソフトを使用して RHEL7 および RHEL6 Linux システムにインストールできます。Extra Packages Enterprise Linux (EPEL) リポジトリと PostgreSQL Global Development Group RPM repository を事前にインストールする必要があります。

Barman の公式 RPM パッケージソフトは、EnterpriseDB が Yum を介して [public RPM repository](#) 経由で配布します。そのウェブサイトに記載されている指示に従ってください。

次に、`root` として次のように入力します。

```
yum install barman
```

注: > Barman 関連のパッケージソフトは、PGDG リポジトリを介して更新されないようにすることをお勧めします。これは、以下の行を追加することで実行できます>内の Barman サーバーに含まれる PGDG リポジトリ定義> `/etc/yum.repos.d/pgdg-*.repo` ファイル: `edb_notranlate_2`> これを行うことで、Barman ソフトウェアのパッケージ管理にのみ EnterpriseDB リポジトリを使用します。

注釈: PGDG リポジトリを介した Barman 関連パッケージソフトの更新を除外することをお勧めします。これは、`/etc/yum.repos.d/pgdg-*.repo` ファイル内の Barman サーバーに含まれる PGDG リポジトリ定義に次の行を追加することで実行できます。 `edb_notranlate_2` これを行うことで、Barman ソフトウェアのパッケージ管理のみを EnterpriseDB リポジトリに依存します。

歴史的な理由により、EnterpriseDB は [Sourceforge.net](#) を通じて Barman のパッケージディストリビューションを維持し続けます。

5.2 パッケージソフトを使用した Debian / Ubuntu へのインストール

Barman は、パッケージソフトを使用して Debian および Ubuntu Linux システムにインストールできます。

Debian および Ubuntu の公式リポジトリで直接利用できますが、これらのリポジトリには利用可能な最新バージョンが含まれていない場合があります。Barman の最新バージョンが必要な場合、これらの両方のリポジトリをインストールするメソッドをお勧めします。

- [Public APT repository](#)、直接管理 Barman 開発者
- [PostgreSQL Community APT repository](#) の指示に従って、[PostgreSQL Community APT repository](#)

注釈: PostgreSQL Community APT リポジトリプロジェクトに Barman 開発者が直接関与したおかげで、Barman の最新バージョンに常にアクセスできます。

Barman のインストールは簡単です。root ユーザとして単純に入力します：

```
apt-get install barman
```

5.3 パッケージソフトを使用した SLES へのインストール

[PGDG SLES repositories](#) で利用可能なパッケージソフトを使用して、

Barman を SLES システムにインストールできます。 [PGDG site](#) で利用可能な指示に従って、必要なリポジトリをインストールします。

サポートされている SLES バージョンは、SLES 12 SP5 および SLES 15 SP3 です。

- SLES 12 のみ*：OpenSUSE python バックポートも有効にする必要がありますリポジトリは次のとおりです（この手順は SLES 15 では不要です）。

```
zypper addrepo https://download.opensuse.org/repositories/  
→devel:languages:python:backports/SLE_12_SP5/devel:languages:python:backports.repo  
zypper addrepo https://download.opensuse.org/repositories/devel:/languages:/python:/  
→backports/SLE_12_SP4 devel_languages_python_backports_sp4  
zypper refresh
```

必要なリポジトリがインストールされたら、Barman を root ユーザとしてインストールできます。

```
zypper install barman
```

5.4 ソースからのインストール

警告: ソースからの Barman の手動インストールは、GNU / Linux のエキスパートユーザーのみが実行する必要があります。この方法で Barman をインストールするには、依存関係管理、*barman* ユーザの作成、*barman.conf* ファイルの構成、*barman cron* コマンドの cron セットアップ、ログ管理などのシステム管理アクティビティが必要です。

backup サーバーで *barman* というシステムユーザを作成します。*barman* ユーザとして、ソースをダウンロードして解凍します。

システム全体のインストールの場合、次を入力します。

```
barman@backup$ ./setup.py build
# run this command with root privileges or through sudo
barman@backup# ./setup.py install
```

ローカルインストールの場合、次のように入力します。

```
barman@backup$ ./setup.py install --user
```

barman アプリケーションは、ユーザディレクトリ *make sure that your* `'PATH'` environment variable is set properly <<https://docs.python.org/3/install/index.html#alternate-installation-the-user-scheme>>`__ にインストールされます)。

Barman is also available on the Python Package Index (PyPI)。 `pip` からインストールできます。

5.5 PostgreSQL クライアントバイナリ

以下の Barman 機能は、PostgreSQL クライアントバイナリに依存しています。

これらのバイナリは PostgreSQL クライアントパッケージソフトとともにインストールされ、次の場所にあります。

- RedHat / CentOS の場合： `/usr/pgsql-${PG_MAJOR_VERSION}/bin`
- Debian / Ubuntu の場合： `/usr/lib/postgresql/${PG_MAJOR_VERSION}/bin`

次のいずれかを保証する必要があります。

1. Barman ユーザは、そのパスに適切な `PG_MAJOR_VERSION` の `bin` ディレクトリを持っています。または、2. 各サーバーの Barman 構成で `path_prefix` オプションが設定され、適切な `PG_MAJOR_VERSION` の `bin` ディレクトリを指します。

第6章 Upgrading Barman

Barman はトランクベースの開発パラダイムに従っています。そのため、最新の安定（stable）バージョンは 1 つだけです。コミットのたびに、Barman は、サポートされている各 PostgreSQL バージョンおよびサポートされている各 Linux ディストリビューションに対して数千の自動テストを実施します。

また、**すべてのバージョンは以前のものと互換性があります**。したがって、Barman のアップグレード処理には通常、`yum update` または `apt update` を使用したパッケージソフトのシンプル更新が必要です。

ただし、開発履歴には次の例外があり、構成に若干の変更が必要でした。

6.1 Barman 2.10 からのアップグレード

`barman-cloud-wal-archive` または `barman-cloud-backup` を使用している場合、バージョン 2.11 からすべてのクラウドユーティリティが新しい `barman-cli-cloud` パッケージに移動されていることに注意する必要があります。したがって、`barman-cli-cloud` パッケージが最新バージョンへのアップグレードのパートとして適切にインストールされていることを保証必要があります。上記のツールを使用していない場合は、通常どおり最新バージョンにアップグレードできます。

6.2 Barman 2.X からのアップグレード（2.8 より前）

Barman 2.7 のバージョンからアップグレード処理前に、プライマリーサーバの `rsync` バックアップメソッドの古いユーザーは、ワーニングを明示的に `concurrent_backup`（PostgreSQL 9.6 以降に推奨）または Barman（現在のデフォルト）に設定する必要があります。。

6.3 Barman 1.X からのアップグレード

Barman インストールが 1.X の場合、アーカイビング戦略を明示的に構成する必要があります。以前は、`archiver` によって制御されるファイルベースのアーカイバはデフォルトで有効になっていました。

Barman インストールを最新バージョンにアップグレード `make` 前に、次の行をグローバルに、またはそれを必要とするサーバーに追加してください。

```
archiver = on
```

さらに、いくつかのリリースでは、Barman はアーカイビング戦略を明示的に設定していないサーバーで `archiver = on` を透過的に設定し、ワーニングを發します。

第7章 Configuration

Barman には 2 種類の構成ファイルがあります。

- グローバル/一般構成
- サーバー構成

メイン構成ファイル（デフォルトでは `/etc/barman.conf` にデフォルト）には、メインディレクトリ、システムユーザ、ログファイルなどの一般的なオプションが含まれています。

サーバー構成ファイル（Barman によってバックアップされるサーバーごとに 1 つ）は `/etc/barman.d` ディレクトリにあり、`.conf` サフィックスが必要です。

重要: : 歴史的な理由により、グローバルオプションとサーバーオプションの両方を含む 1 つの構成ファイルを保持できます。ただし、メンテナンス上の理由から、このアプローチは非推奨です。

Barman の構成ファイルは、INI フォーマットに従います。

構成ファイルは、異なるタイプのパラメーターを受け入れます。

- 文字列
- 列挙型
- 整数
- ブール値、`on/true/1` も `off/false/0` も受け入れられます。

それらのどれも引用する必要はありません。

注: 一部の `enum` では `off` は許可されますが、`false` は許可されません。

7.1 オプションの範囲

すべての構成オプションにはスコープがあります。

- グローバル
- サーバー
- `global / server`: 一般的にグローバルレベルで設定できるサーバーオプション

グローバルオプションは、`[barman]` label によって INI ファイルで識別される一般セクションで許可されます。

```
[barman]
; ... global and global/server options go here
```

サーバーオプションは、サーバーセクションでのみ指定できます。サーバーセクションは、構成ファイルの行で大括弧〔および〕で識別されます。サーバーセクションは、Barman のそのサーバーの ID を表します。次の例では、pg 名前付けのサーバーのセクションを指定しています。

```
[pg]
; Configuration options for the
; server named pg go here
```

Barman でサーバー名として使用できない 2 つの予約語があります。

- **barman** : グローバルセクションの識別子
- **all** : Barman が管理するすべてのサーバーでいくつかのコマンドをシーケンスに実行できる便利なショートカット

Barman は **Convention over Configuration** デザインパラダイムを実装しており、柔軟性を失わずに構成に必要なオプションの数を削減しようとしています。したがって、一部のサーバーオプションはグローバルレベルで定義し、サーバーレベルでオーバーライドできます。これにより、ユーザーは一般的な動作を指定し、1 つ以上のサーバーに対してそれを調整できます。これらのオプションには、グローバル/サーバースコープがあります。

使用可能なすべての構成とそのスコープのリストについては、[section 5 of the 'man' page](#) を参照してください。

```
man 5 barman
```

7.2 設定の例

以下は、メイン構成ファイルの基本的な例です。

```
[barman]
barman_user = barman
configuration_files_directory = /etc/barman.d
barman_home = /var/lib/barman
log_file = /var/log/barman/barman.log
log_level = INFO
compression = gzip
```

一方、以下の例は、ストリーミングバックアップを使用するサーバー構成ファイル。

```
[streaming-pg]
description = "Example of PostgreSQL Database (Streaming-Only)"
conninfo = host=pg user=barman dbname=postgres
```

(次のページに続く)

(前のページからの続き)

```
streaming_conninfo = host=pg user=streaming_barman
backup_method = postgres
streaming_archiver = on
slot_name = barman
```

次のコードは、`rsync` / `SSH` を使用した従来のバックアップの基本的な例を示しています。

```
[ssh-pg]
description = "Example of PostgreSQL Database (via Ssh)"
ssh_command = ssh postgres@pg
conninfo = host=pg user=barman dbname=postgres
backup_method = rsync
parallel_jobs = 1
reuse_backup = link
archiver = on
```

詳細については、配布された `barman.conf` ファイル、および `ssh-server.conf-template` および `streaming-server.conf-template` テンプレートファイルを参照してください。

第8章 Setup of a new server in Barman

「設計とアーキテクチャ」セクションで説明したように、次の規則を使用します。

PostgreSQL がインストールされているサーバー ID およびホスト名としての pg Barman が配置されているホスト名前として backup - backup サーバーで Barman を実行しているユーザとしての barman（構成内のパラメータ barman_user） - pg サーバーで PostgreSQL を実行しているユーザとして postgres

重要: Barman のサーバーは、バックアップおよびリカバリ可能性の履歴全体で同じ PostgreSQL インスタンスを参照する必要があります（つまり、同じシステム識別子）。これは、例 `pg\_upgrade` を使用してインスタンスのアップグレードを実行する場合、Barman で同じサーバー定義を再使用してはならず、共通点がないため別のサーバー定義を使用する必要があることを意味します。

8.1 サーバー構成ファイル

/etc/barman.d ディレクトリに、次のコンテンツの pg.conf という新しいファイルを作成します。

```
[pg]
description = "Our main PostgreSQL server"
conninfo = host=pg user=barman dbname=postgres
backup_method = postgres
# backup_method = rsync
```

conninfo オプションは、セクション「準備手順：PostgreSQL 接続」_に応じて設定されます。

backup_method オプションの意味については、このガイドのバックアップセクションで説明します。

WAL アーカイビングにストリーミング接続を使用するか、サーバーのバックアップを作成する場合は、サーバー構成ファイルに streaming_conninfo パラメータも必要です。

```
streaming_conninfo = host=pg user=streaming_barman dbname=postgres
```

セクション「準備手順：PostgreSQL 接続」で説明されているように、この値を適宜選択する必要があります。

8.2 WAL ストリーミング

Barman は、ユーザーが標準の `archive_command` 戦略に加えて、PostgreSQL サーバーからの連続的な WAL ストリーミングを追加できるようにすることで、Recovery Point Objective (RPO) を削減できます。

Barman は `pg_receivewal` に依存しています。これは、ネイティブストリーミングレプリケーションプロトコルを活用し、PostgreSQL サーバー（マスタまたはスタンバイ）からトランザクションログを継続的に受信する PostgreSQL 9.2 から利用可能なユーティリティです。PostgreSQL 10 より前は、`pg_receivewal` の名前付けは `pg_receivevlog` でした。

重要: Barman では、`pg_receivewal` が同じサーバーにインストールされている必要があります。PostgreSQL 9.2 サーバーの場合、Barman と一緒にバージョン 9.2 の `pg_receivevlog` をインストールする必要があります。PostgreSQL 9.3 以降の場合、`pg_receivewal` の最新バージョンをインストールすることをお勧めします。これは、互換性があるためです。そうでない場合、ユーザーは `pg_receivewal` / `pg_receivevlog` のマルチプルのバージョンを Barman サーバーにインストールし、構成ファイルの `path_prefix` オプションを使用して、サーバーの特定のバージョンを適切に指すことができます。

トランザクションログのストリーミングを有効にするには、オーダーのことが必要です。

1. 前述のようにストリーミング接続をセットアップします 2. `streaming_archiver` オプションを `on` に設定します

前述の要件が満たされている場合、`cron` コマンドは、`receive-wal` コマンドの実行を通じてログストリーミングを透過的に管理します。これは推奨シナリオです。

ただし、ユーザーは `receive-wal` コマンドを手動で実行できます。

```
barman receive-wal <server_name>
```

注釈: `receive-wal` コマンドはフォアグラウンドプロセスです。

トランザクションログは、`streaming_wals_directory` 構成オプションで指定されたディレクトリに直接ストリーミングされ、`archive-wal` コマンドによってアーカイブされます。

`streaming_archiver_name` パラメータで特に指定されていない限り、PostgreSQL 9.3 以降の場合のみ、Barman は WAL ストリーマープロセスの `application_name` を `barman_receive_wal` に設定し、PostgreSQL サーバーの `pg_stat_replication` システムビューでその状態をモニタできるようにします。

8.2.1 複製スロット

重要: レプリケーションスロットは、PostgreSQL 9.4 以降で使用可能です

レプリケーションスロットは、すべてのアーカイバが受信するまで PostgreSQL サーバーが WAL ファイルを削除しないように保証自動化された方法です。Barman はこのメカニズムを使用して、PostgreSQL からトランザクションログを受信します。

PostgreSQL manual でレプリケーションスロットに関する詳細情報を見つけることができます。

バックアップアーキテクチャをストリーミング接続のみに基づいて構築することもできます。このシナリオは、Docker ベースの PostgreSQL サーバーを構成したり、Windows 上で実行されて PostgreSQL サーバーと連携したりする場合にも役立ちます。

重要: 現時点では、Windows サポートはまだ継続的統合システムのパートではないため、まだ実験段階です。

8.2.2 WAL ストリーミングを設定する方法

まず、PostgreSQL サーバーは、トランザクションログファイルを Barman サーバーにストリームするように構成する必要があります。

Barman から PostgreSQL サーバーへのストリーミング接続を構成するには、既に説明したように、サーバー構成ファイルに次の行を含めて、`streaming_archiver` を有効にする必要があります。

```
streaming_archiver = on
```

レプリケーションスロットの使用を計画している場合（推奨）、ストリーミングベースのトランザクションログアーカイビングのセットアップに不可欠なもう 1 つのオプションは、`slot_name` オプションです。

```
slot_name = barman
```

このオプションは、Barman が使用するレプリケーションスロットの名前を定義します。レプリケーションスロットを使用する場合は必須です。

レプリケーションスロット名前を設定するとき、次のコマンドを使用して Barman のレプリケーションスロットを手動で作成できます。

```
barman@backup$ barman receive-wal --create-slot pg
Creating physical replication slot barman on server pg
Replication slot barman created
```

Barman 2.10 以降では、Barman を設定してレプリケーションスロットを自動的に作成することができます：

```
create_slot = auto
```

8.2.3 リカバリを伴う部分的な WAL ファイルの制限

`pg_receivewal` の標準的な動作は、WAL セグメント名前の後に、サフィックス `.partial` が付いたファイルにトランザクション情報を書き込むことです。

Barman は、部分ファイルがサーバーの `streaming_wals_directory` にあることを期待しています。完了すると、`pg_receivewal` は `.partial` サフィックスを削除して次の接尾辞を開き、ファイルを永続的なストレージと圧縮のために Barman の `archive-wal` コマンドに配信します。

マスタ PostgreSQL サーバーの突然の回復不能な障害の場合、Barman にストリーミングされた `.partial` ファイルには、標準アーカイバーが (PostgreSQL の `archive_command` を介して) Barman に配信できなかった非常に重要な情報が含まれています。

Barman 2.10 以降、`get-wal` コマンドは、`--partial/-P` オプションを介して現在の `.partial` WAL ファイルのコンテンツを結果することができます。これは、完全または特定の時点までのリカバリの場合に特に役立ちます。そのため、`get-wal` を有効にして `--standby-mode` を使用せずに `recover` コマンドを実行した場合、Barman は `-P` オプションを `barman-wal-restore` に自動的に追加します (`restore_command` リカバリオプションでリモート `get-wal` コマンドにリレーします)。

`get-wal` は、WAL ファイルが既に Barman に出荷されているがまだアーカイブされていない場合、`incoming` ディレクトリも検索します。

8.3 `archive_command` を介した WAL アーカイビング

`archive_command` は、WAL ファイルをアーカイブする従来のメソッドです。

この PostgreSQL 設定パラメータの値は、WAL ファイルを Barman 着信ディレクトリにコピーするために PostgreSQL サーバーによって実行されるシェルコマンドである必要があります。

これは 2 つの方法で行うことができ、両方とも SSH 接続が必要です。

- `barman-wal-archive` ユーティリティ経由 (Barman 2.6 以降)
- `rsync` / SSH 経由 (Barman 2.6 より前の一般的なアプローチ)

詳細については、以下のセクションを参照してください。

重要: PostgreSQL 9.5 は、スタンバイから `archive_command` を使用した WAL ファイルアーカイブのサポートを導入しました。Barman がこの機能をサポートする方法の詳細については、「同時バックアップとスタンバイからのバックアップ」セクションをお読みください。

8.3.1 barman-wal-archive を介した WAL アーカイビング

Barman 2.6 以降、`archive_command` を介して WAL ファイルを Barman に安全かつ確実にアーカイブするための推奨方法は、EnterpriseDB パブリックリポジトリを介して配布され、GNU GPL 3 ライセンスで利用可能な `barman-cli` パッケージに含まれる `barman-wal-archive` コマンドを使用することです。`barman-cli` は、Barman クラスターのパートである各 PostgreSQL サーバーにインストールする必要があります。

`rsync` / `SSH` の代わりに `barman-wal-archive` を使用すると、Barman サーバー上の出荷された WAL ファイルのデータ破損のリスクが軽減されます。`rsync` / `SSH` を `archive_command` WAL ファイルとして使用する場合、ファイルのコンテンツが宛先のディスクにフラッシュされ、`fsync` されることを保証するメカニズムはありません。

このため、Barman の `put-wal` コマンド (2.6 で導入) とネイティブに通信する `barman-wal-archive` ユーティリティを開発しました。このコマンドは、ファイルを受信し、そのコンテンツを `fsync` して、そのサーバーの適切な `incoming` ディレクトリに配置します。したがって、`archive_command` で使用される唯一のパラメータはサーバーの ID であるため、`barman-wal-archive` は Barman の間違った場所/ディレクトリに WAL ファイルをコピーするリスクを減らします。

`barman-wal-archive` コマンドの詳細については、PostgreSQL サーバーで「`man barman-wal-archive`」と入力してください。

次のコマンドを使用して、`barman-wal-archive` が Barman サーバーに接続できること、および必要な PostgreSQL サーバーが受信 WAL ファイルを受け入れるように Barman で構成されていることを確認できます。

```
barman-wal-archive --test backup pg DUMMY
```

`backup` は Barman がインストールされているホスト、`pg` は Barman で設定されている PostgreSQL サーバーの名前、`DUMMY` はプレースホルダです (`barman-wal-archive` には WAL ファイル名前の引数が必要ですが、無視されます)。

`pg` データベースで PostgreSQL インスタンスの `postgresql.conf` ファイルを編集し、アーカイブモードをアクティブにし、`barman-wal-archive` を使用するように `archive_command` を設定します。

```
archive_mode = on
wal_level = replica
archive_command = barman-wal-archive backup pg %p
```

次に、PostgreSQL サーバーをリスタートします。

8.3.2 rsync / SSH を介した WAL アーカイビング

`show-servers` Barman コマンドを使用して `incoming_wals_directory` 値を検索することで、着信 WAL ディレクトリを取得できます。

```
barman@backup$ barman show-servers pg |grep incoming_wals_directory
incoming_wals_directory: /var/lib/barman/pg/incoming
```

pg データベース上の PostgreSQL インスタンスの `postgresql.conf` ファイルを編集し、アーカイブモードをアクティブにします。

```
archive_mode = on
wal_level = replica
archive_command = rsync -a %p barman@backup:INCOMING_WALS_DIRECTORY/%f
```

上記の `barman show-servers pg` コマンドによって返された値で `INCOMING_WALS_DIRECTORY` プレースホルダを必ず変更してください。

PostgreSQL サーバーを再起動します。

場合によっては、`archive_command` プロセスにさらに厳密なチェックを追加することもできます。例、一部のユーザーは次のことを提案しています。

```
archive_command = test $(/bin/hostname --fqdn) = HOSTNAME \
    && rsync -a %p barman@backup:INCOMING_WALS_DIRECTORY/%f
```

`HOSTNAME` プレースホルダは、`hostname --fqdn` によって返される値に置き換える必要があります。このトリックは、サーバーのクローンが作成され、復旧された PostgreSQL インスタンスから WAL ファイルを受信しないようにする場合の安全策です。

8.4 WAL アーカイビング設定の検証

継続的アーカイビングが有効で適切に機能していることをテストオーダーには、PostgreSQL サーバーとバックアップサーバの両方を確認する必要があります。特に、WAL ファイルが宛先ディレクトリに正しく収集されていることを確認する必要があります。

この目的のため、および WAL アーカイビングプロセスの検証を容易にするために、`switch-wal` コマンドが開発されました：

```
barman@backup$ barman switch-wal --force --archive pg
```

上記のコマンドは、PostgreSQL に WAL ファイルをスイッチさせ、Barman でアーカイビングプロセスをトリガーさせます。Barman は 1 つのファイルが 30 秒以内に到着するのを待ちます（タイムアウトは `--archive-timeout` オプションで変更できます）。WAL ファイルが受信されない場合、エラーが返されます。

`barman check` コマンドを使用して、WAL アーカイビングが正しく構成されているかどうかを確認できます。

8.5 ストリーミングバックアップ

Barman は、PostgreSQL 9.1 から利用可能なユーティリティである `pg_basebackup` に依存して、ストリーミング接続を使用して PostgreSQL サーバーをバックアップできます。

重要: Barman では、`pg_basebackup` が同じサーバーにインストールされている必要があります。PostgreSQL 9.2 サーバーの場合、Barman と一緒にバージョン 9.2 の `pg_basebackup` をインストールする必要があります。PostgreSQL 9.3 以上の場合、`pg_basebackup` の最新バージョンをインストールすることをお勧めします。これは後方互換性があるためです。構成ファイルの `path_prefix` オプションを使用して、Barman サーバーに `pg_basebackup` のマルチプルのバージョンをインストールし、サーバーの特定のバージョンを適切に指すこともできます。

ストリーミング接続でサーバーを正常にバックアップするには、バックアップメソッドとして `postgres` を使用する必要があります。

```
backup_method = postgres
```

重要: WAL が `archiver` または `streaming_archiver` を介して Barman に正しくアーカイブされていない場合、バックアップをスタートできません。

サーバー構成が有効かどうかを確認するには、`barman check` コマンドを使用できます。

```
barman@backup$ barman check pg
```

バックアップをスタートするには、`barman backup` コマンドを使用できます。

```
barman@backup$ barman backup pg
```

`postgres` バックアップメソッドを使用する場合、テーブルスペースのサポートには.. **IMPORTANT::** `pg_basebackup` 9.4 以上が必要です。

8.6 rsync / SSH を使用したバックアップ

`rsync` を介したバックアップは、2.0 より前の唯一の利用可能なメソッドであり、現在、増分バックアップ機能をサポートする唯一のバックアップメソッドです。詳細については、「機能の詳細」セクションをご覧ください。

`rsync` を使用してバックアップを取るには、これらのパラメーターを Barman サーバー構成ファイル内に配置する必要があります。

```
backup_method = rsync
ssh_command = ssh postgres@pg
```

`backup_method` オプションは `rsync` バックアップメソッドをアクティブにし、Barman サーバーから PostgreSQL サーバーへの SSH 接続を正しく作成するには `ssh_command` オプションが必要です。

重要: WAL が *archiver* または *streaming_archiver* を介して Barman に正しくアーカイブされていない場合、バックアップをスタートできません。

サーバー構成が有効かどうかを確認するには、`barman check` コマンドを使用できます。

```
barman@backup$ barman check pg
```

バックアップをとるには、`barman backup` コマンドを使用します。

```
barman@backup$ barman backup pg
```

8.7 Windows ベースのサーバーをセットアップする方法

WAL アーカイビングとバックアップの両方のストリーミング接続を使用して、Windows で実行されている PostgreSQL サーバーをバックアップできます。

重要: この機能は、継続的インテグレーションシステムのパートではないため、まだ実験段階です。

ストリーミング接続のセットアップについては、前述のすべての手順に従ってください。

警告: : 現時点では、Windows から Linux への `pg_basebackup` の相互運用性はまだ実験段階です。Windows サーバーからバックアップを取得する際に問題が発生し、PostgreSQL ロケールが英語でない場合、問題を回避するには、PostgreSQL に英語でメッセージを送信するよう指示します。これを行うには、`postgresql.conf` ファイルに次のパラメータを追加します。`edb_notranslate_0` これは問題を修正することが報告されています。

通常どおりサーバーをバックアップできます。

Windows サーバーではリモートリカバリはサポートされていないため、Barman サーバーでクラスターをローカルで回復し、Windows サーバー上のすべてのファイルをコピーするか、PostgreSQL サーバーと Barman サーバー間で共有されるフォルダーを使用する必要があります。

さらに、PostgreSQL の実行を選択したシステムユーザが、復元されたデータにアクセスするために必要なパーミッションを持っていることを確認してください。基本的に、PostgreSQL データディレクトリを完全に制御する必要があります。

第9章 準備手順

このセクションには、Barman で PostgreSQL サーバーをセットアップする前に実行する必要がある予備的な手順が含まれています。

重要: 先に進む前に、「設計とアーキテクチャ」セクションで概説されているように、WAL アーカイビングおよびバックアップ戦略に関して決定を下すことが重要です。特に、使用する WAL アーカイビング方法とバックアップメソッドを決定する必要があります。

9.1 PostgreSQL 接続

backup サーバーが pg 上の PostgreSQL サーバーにスーパーユーザとして接続できること、または PostgreSQL 10 以降から、データベースに接続するユーザに正しい特権セットが付与されていることを確認する必要があります。

次のように、barman 名前付けの特定のスーパーユーザを PostgreSQL で作成できます。

```
postgres@pg$ createuser -s -P barman
```

または、次のように必要な特権セットを持つ通常のユーザを作成します。

```
postgres@pg$ createuser -P barman
```

```
GRANT EXECUTE ON FUNCTION pg_start_backup(text, boolean, boolean) TO barman;
GRANT EXECUTE ON FUNCTION pg_stop_backup() TO barman;
GRANT EXECUTE ON FUNCTION pg_stop_backup(boolean, boolean) TO barman;
GRANT EXECUTE ON FUNCTION pg_switch_wal() TO barman;
GRANT EXECUTE ON FUNCTION pg_create_restore_point(text) TO barman;

GRANT pg_read_all_settings TO barman;
GRANT pg_read_all_stats TO barman;
```

実際のスーパーユーザがいないと、barman switch-wal コマンドの--force オプションが機能しないことに注意してください。

重要: 上記の createuser コマンドはパスワードのプロンプトを求めます。パスワードは、backup サーバーの ~barman/pgpass ファイルに追加することをお勧めします。詳細については、[The Password File](#) を参照

してください。

この接続は、モニタリングを行うためだけでなく、サーバーとのアクティビティを調整オーダーに Barman に必要です。

PostgreSQL が提供するものの中から好みのクライアント認証メソッドを選択できます。詳細については、[Client Authentication](#) を参照してください。

続行する前に、必ず次のコマンドをテストしてください。

```
barman@backup$ psql -c SELECT version() -U barman -h pg postgres
```

上記の情報（ユーザ名前、ホスト名前、データベース名前）を書き留めて、後で使えるようにしておいてください。この例のように、サーバー構成の `conninfo` オプションで必要になります。

```
[pg]
; ...
conninfo = host=pg user=barman dbname=postgres
```

注釈: Barman は、PostgreSQL サーバー 9.0 以降の *application_name* 接続オプションを尊重します。

9.2 PostgreSQL WAL のアーカイビングとレプリケーション

先に進む前に、`pg` で PostgreSQL を適切に構成して、Barman サーバーからのストリーミングレプリケーション接続を受け入れる必要があります。PostgreSQL 文書の次のセクションをお読みください：

- [Role attributes](#)
- [The pg_hba.conf file](#)
- [Setting up standby servers using streaming replication](#)

非常に重要な設定パラメータの 1 つは、`wal_level` パラメータです。このパラメータは、バックアップを一貫させるために必要な保証の有用な情報がトランザクションログファイルに含まれるように構成する必要があります。

```
wal_level = replica
```

PostgreSQL 9.4 以降では、ロジカルデコーディングが必要な場合に、`wal_level` を `logical` に設定することもできます。

9.6 より古い PostgreSQL バージョンの場合、`wal_level` を `hot_standby` に設定する必要があります。

構成を更新するには、PostgreSQL サーバーを再起動します。

9.3 PostgreSQL ストリーミング接続

WAL ストリーミングまたはストリーミングバックアップを使用する場合は、ストリーミング接続をセットアップする必要があります。次のように、`streaming_barman` 名前付けの PostgreSQL で特定のユーザを作成することをお勧めします。

```
postgres@pg$ createuser -P --replication streaming_barman
```

重要: 上記のコマンドはパスワードのプロンプトを求めます。パスワードは、*backup* サーバーの `~barman/.pgpass` ファイルに追加することをお勧めします。詳細については、[The Password File](#) を参照してください。

次のコマンドを使用して、ストリーミング接続が機能することを手動で確認できます。

```
barman@backup$ psql -U streaming_barman -h pg \
-c "IDENTIFY_SYSTEM" \
replication=1
```

重要: 先に進む前に、ストリーミングレプリケーションを介して接続できることを確認してください。

また、PostgreSQL 構成ファイルで `max_wal_senders` パラメータを構成する必要があります。WAL 送信者の数は、実装した PostgreSQL アーキテクチャによって異なります。この例では、2 に設定しています。

```
max_wal_senders = 2
```

このオプションは、サーバーが管理できる同時ストリーミング接続の最大数を表します。

もう 1 つの重要なパラメータは `max_replication_slots` です。これは、サーバーが管理できるレプリケーションスロット¹の最大数を表します。このパラメータは、ストリーミング接続を使用して WAL ファイルをストリーミング接続経由で受信することを計画している場合に必要です。

```
max_replication_slots = 2
```

`max_replication_slots` および `max_wal_senders` に提案されている値は例として考慮し、実際のセットアップで使用する値は、アーキテクチャを慎重に評価した後に選択する必要があります。ガイドラインと説明については、PostgreSQL の文書を参照してください。

¹ レプリケーションスロットは PostgreSQL 9.4 で導入されました。詳細については、「WAL ストリーミング/レプリケーションスロット」セクションを参照してください。

9.4 SSH 接続

SSH は、リモートサーバへのリモートシェルを開き、サーバーとローカルシステム間でファイルをコピーするためのプロトコルおよびツールセットです。Digital Ocean の記事 [SSH Essentials](https://www.digitalocean.com/community/tutorials/ssh-essentials-working-with-ssh-servers-clients-and-keys) に、SSH の使用文書があります。

SSH キー交換は、異なるマシン上のユーザー間で安全なパスワードなしの接続を実装するために使用される非常に一般的な方法であり、WAL アーカイビングおよびバックアップに **rsync** を使用する必要があります。

注釈: ストリーミング接続をトランザクションログのアーカイブと PostgreSQL サーバーのバックアップのみに使用する場合、このプロシージャは必要ありません。

[ssh_essentials]: <https://www.digitalocean.com/community/tutorials/ssh-essentials-working-with-ssh-servers-clients-and-keys>

9.5 postgres ユーザの SSH 設定

以前に実行したことがない限り、PostgreSQL ユーザの SSH キーを作成する必要があります。pg ホストに postgres としてログインし、次を入力します。

```
postgres@pg$ ssh-keygen -t rsa
```

パスワードを提供せずにホストから接続するにはこのキーを使用する必要があるため、キーの作成中にパスフレーズを入力しないでください。

9.6 barman ユーザの SSH 設定

前の段落と同様に、Barman ユーザの SSH キーを作成する必要があります。backup ホストに barman としてログインし、次を入力します。

```
barman@backup$ ssh-keygen -t rsa
```

同じ理由で、パスフレーズを入力しないでください。

9.7 PostgreSQL から Barman へ

archive_command 設定を使用して WAL ファイルを正しくアーカイブするには、PostgreSQL サーバーからバックアップサーバへの SSH 接続が必要です。

PostgreSQL サーバーからバックアップサーバに正常に接続するには、PostgreSQL 公開キーを、barman ユーザのバックアップサーバの認証キーに設定する必要があります。

承認される公開キーは、`postgres` ユーザホームディレクトリ内の `.ssh/id_rsa.pub` 名前付けファイルに保存され、そのコンテンツは、バックアップサーバの `barman` ユーザのホームディレクトリ内の `.ssh/authorized_keys` 名前付けファイルに含まれている必要があります。`authorized_keys` ファイルが存在しない場合は、`600` を権限として使用して作成します。

SSH キーの交換が正常に完了した場合、次のコマンドは出力なしで成功します。

```
postgres@pg$ ssh barman@backup -C true
```

`archive_command` 設定パラメータの値については、「`archivecommand` セクションを介した WAL アーカイビング」で説明します。

9.8 Barman から PostgreSQL へ

バックアップサーバと PostgreSQL サーバー間の SSH 接続は、`rsync` を介した従来のバックアップに使用されます。PostgreSQL サーバーからバックアップサーバへの接続と同様に、`postgres` ユーザに対して PostgreSQL サーバーのバックアップサーバの公開キーを承認する必要があります。

`barman` サーバーの `.ssh/id_rsa.pub` ファイルのコンテンツは、PostgreSQL サーバーの `.ssh/authorized_keys` 名前付けのファイルに配置する必要があります。そのファイルの許可は `600` でなければなりません。

キーの交換が正常に完了した場合、次のコマンドは出力なしで成功します。

```
barman@backup$ ssh postgres@pg -C true
```


第10章 General commands

Barman には多くのコマンドがあり、説明のために、範囲ごとに整理できます。

一般的なコマンドの範囲は、多くの PostgreSQL サーバーをバックアップできる Barman サーバー全体です。代わりに、**サーバーコマンド** は、指定されたサーバーでのみ動作します。**バックアップコマンド**は、特定のサーバーから取得したバックアップに対して機能します。

次のリストには、一般的なコマンドが含まれています。

10.1 cron

barman には、長時間実行されるデーモンやサービスファイルは含まれていません (`systemctl start`、`service start` などには何もありません)。代わりに、**barman** のバックグラウンド「定常状態」バックアップ操作を実行するための **barman cron** サブコマンドが提供されています。

cron コマンドを使用して、WAL ファイルとバックアップの両方でメンテナンス操作を実行できます。

```
barman cron
```

注釈: このコマンドは、**cron** スクリプトで実行する必要があります。 *barman cron* を毎分実行するようにスケジュールすることをお勧めします。rpm または debian パッケージを使用して Barman をインストールした場合、毎分実行される **cron** エントリーが作成されます。

barman cron はサーバーベースで WAL アーカイビング操作を同時に実行します。これにより、次の機能を持つサーバーに保存ポリシーも適用されます。

- **retention_policy** が空ではなく有効ではありません。
- **retention_policy_mode** を **auto** に設定。

cron コマンドは、**receive-wal** コマンドを透過的に実行することにより、要求したサーバーに対して WAL ストリーミングが開始されるようにします。

オーダーコマンドによって開始された操作を停止するには、**cron** エントリーをコメントアウトで実行します。

```
barman receive-wal --stop SERVER_NAME
```

barman list-servers をチェックして、すべてのサーバーを **make** ことをお勧めします。

10.2 diagnose

`diagnose` コマンドは、診断およびサポートの目的に役立つ JSON レポートを作成します。このレポートには、構成されたすべてのサーバーの情報が含まれます。

重要: 診断が JSON で記述されていて、そのフォーマットが機械可読であると考えられている場合でも、その構造はインタフェースのパートとは見なされません。フォーマットは、異なる Barman バージョン間で変更できます。

10.3 list-servers

以下を使用して、バックアップシステム用に設定されたアクティブなサーバーのリストを表示できます。

```
barman list-servers
```

`--minimal` オプションを使用すると、機械で読み取り可能な出力を取得できます。

```
barman list-servers --minimal
```

第11章 Server commands

前のセクションで述べたように、サーバーコマンドは PostgreSQL サーバーまたは Barman のエリアで直接動作し、ステータスの確認、メンテナンス操作の実行、バックアップの取得、WAL アーカイブの管理に役立ちます。

11.1 archive-wal

`archive-wal` コマンドは、特定のサーバーの WAL ファイルでメンテナンス操作を実行します。この操作には、ストリーミング接続または `archive_command` またはその両方から受信した WAL ファイルの処理が含まれます。

重要: `archive-wal` コマンドは、直接起動できる場合でも、`cron` 一般コマンドから開始されるように設計されています。

11.2 backup

`backup` コマンドは、特定のサーバーのフルバックアップ (*base backup*) を取得します。新しいバックアップの対応する設定パラメータを上書きできるいくつかのオプションがあります。詳細については、マニュアルページを参照してください。

以下を使用して、特定のサーバーのフルバックアップを実行できます。

```
barman backup <server_name>
```

ちなみに: `barman backup all` を使用して、構成済みのすべてのサーバーを順次バックアップできます。

Barman 2.10 では、`backup` コマンドに `-w/--wait` オプションが導入されています。設定すると、Barman は一時的にバックアップの状態を `WAITING_FOR_WALS` に保存し、必要なすべての WAL ファイルがアーカイブされるのを待ってから状態を `DONE` に設定し、バックアップ後のフックスクリプトを続行します。バックアップオプションが指定されている場合、Barman は指定された秒数が経過すると WAL ファイルの待機を停止し、状態は `label` のままになります。

11.3 check

check コマンドを使用して、特定のサーバーへの接続と構成の一貫性を確認できます。

```
barman check <server_name>
```

ちなみに: `barman check all` を使用して、構成済みのすべてのサーバーを確認できます。

重要: `check` コマンドは、おそらく Barman が実装する最も重要な機能です。アラートおよびモニタリングインフラストラクチャと統合することをお勧めします。 `--nagios` オプションを使用すると、Nagios / Icinga のプラグインを簡単に作成できます。

11.4 generate-manifest

このコマンドは、バックアップがリモートで作成され、`pg_basebackup` が関与せず、`backup_manifest` ファイルがバックアップに存在しない場合に役立ちます。barman サーバーのバックアップを使用して、`backup_id` から `backup_manifest` ファイルを生成します。ファイルが既に存在する場合、生成コマンドはアボートされます。

コマンド例: `bash barman generate-manifest <server_name> <backup_id> backup_id [バックアップ id shortcuts] { # backup-id-shortcuts }` のいずれかを使用できます。

このコマンドは、次のように `post_backup` フックスクリプトとしても使用できます。 `bash post_backup_script=barman generate-manifest ${BARMAN_SERVER} ${BARMAN_BACKUP_ID}`

11.5 get-wal

Barman では、ユーザーは `get-wal` コマンドを使用して WAL アーカイブから xlog ファイルを要求できます。

```
barman get-wal [-o OUTPUT_DIRECTORY] () <server_name> <wal_id>
```

要求された WAL ファイルがサーバーアーカイブで見つかった場合、特に指定がない限り、非圧縮コンテンツは STDOUT に返されます。

`get-wal` コマンドでは、次のオプションを使用できます。

- `-o` により、ユーザーは Barman のインストール先ディレクトリを指定できます。要求された WAL ファイルをデポジットします
- `-j` は、`bzip2` アルゴリズムを使用して出力を圧縮します
- `-x` は、`gzip` アルゴリズムを使用して出力を圧縮します
- `-p SIZE` は、アーカイブから WAL ファイルまでのピークを開始します。要求されたファイル

リカバリオペレーション中に `get-wal` を使用して、Barman サーバーをサーバーの WAL ハブに変換することができます。これは、`get-wal` 値を `recovery_options` グローバル/サーバー構成オプションに追加することで自動的に実現できます。

```
recovery_options = get-wal
```

`recovery_options` は、コンマ区切り値のリストを受け入れるグローバル/サーバオプションです。リカバリオペレーション中にキーワード `get-wal` が存在する場合、Barman は、`barman get-wal` を使用して必要な WAL ファイルをフェッチするように `restore_command` を設定することにより、リカバリ構成を準備します。同様に、実行時に `recover` コマンドに `--get-wal` オプションを使用できます。

`get-wal` が `recovery_options` に設定されているが、リカバリオペレーション中に不要な場合、`--no-get-wal` オプションを `recover` コマンドで使用して、`get-wal` リカバリオプションを無効にすることができます。

これは、ローカルリカバリの `restore_command` の例です。

```
restore_command = sudo -u barman barman get-wal SERVER %f > %p
```

`get-wal` コマンドは常に `barman` ユーザとして起動する必要があり、カタログから WAL ファイルを読み取るための正しいパーミッションが必要であることにノートしてください。これが、例で `sudo -u barman` を使用している理由です。

リモートリカバリ用に `recovery_options` を `get-wal` に設定すると、代わりに `barman-wal-restore` スクリプトを使用して `restore_command` が生成されます。`barman-wal-restore` は、SSH 接続エラーを管理する、より回復力のあるシェルスクリプトです。

このスクリプトには、WAL ファイルの自動圧縮と解凍、PostgreSQL がそれらの 1 つを適用している間に次の WAL ファイルを取得できる `*peek*` 機能など、多くの便利なオプションがあります。これは、PostgreSQL と Barman 間の帯域幅の使用を最適化する優れた方法です。

`barman-wal-restore` は `barman-cli` パッケージで利用可能です。

これは、リモートリカバリの `restore_command` の例です。

```
restore_command = barman-wal-restore -U barman backup SERVER %f %p
```

SSH を使用して Barman サーバーと通信するため、`postgres` ユーザがバックアップサーバで `barman` としてログインするには SSH キー認証が必要です。

次のコマンドを使用して、`barman-wal-restore` が Barman サーバーに接続でき、必要な PostgreSQL サーバーが WAL ファイルを送信するように Barman で設定されていることを確認できます。

```
barman-wal-restore --test backup pg DUMMY DUMMY
```

`backup` は Barman がインストールされているホスト、`pg` は Barman で設定されている PostgreSQL サーバーの名前、`DUMMY` はプレースホルダです（`barman-wal-restore` には WAL ファイル名前と宛先ディレクトリの 2 つの引数が必要です。これらは無視されます）。`barman-wal-restore` コマンドの詳細については、PostgreSQL サーバーで「`man barman-wal-restore`」と入力してください。

11.6 list-backups

以下を使用して、特定のサーバーで使用可能なバックアップのカタログをリストできます。

```
barman list-backups <server_name>
```

ちなみに: サーバー名前として *all* を使用して、すべてのサーバーのバックアップの完全なリストを要求できます。

機械可読な出力を得るには、`--minimal` オプションを使用できます。

11.7 rebuild-xlogdb

いつでも、特定のサーバー（またはすべてのサーバー、`all` ショートカットを使用）の WAL アーカイブのコンテンツを再生成できます。WAL アーカイブは `xlog.db` ファイルに含まれており、Barman が管理するすべてのサーバーには独自のコピーがあります。

`xlog.db` ファイルは、`rebuild-xlogdb` コマンドで再構築できます。これにより、アーカイブされたすべての WAL ファイルがスキャンされ、アーカイブのメタデータが再生成されます。

例：

```
barman rebuild-xlogdb <server_name>
```

11.8 receive-wal

このコマンドは、ストリーミングプロトコルを使用して PostgreSQL ストリーミング接続から WAL ファイルを受信する `receive-wal` プロセスを管理します。

11.8.1 receive-wal プロセス管理

コマンドをオプションなしで実行すると、`receive-wal` プロセスが開始されます。このコマンドは、`pg_receivewal` PostgreSQL コマンドに基づいています。

```
barman receive-wal <server_name>
```

注釈: `receive-wal` コマンドはフォアグラウンドプロセスです。

`--stop` オプションを使用してコマンドを実行すると、現在実行中の `receive-wal` プロセスが停止します。

`receive-wal` プロセスは、ステータスファイルを使用して、トランザクションログの最後書き込まれたレコードを追跡します。ステータスファイルをクリーニングニーズがある場合は、`--reset` オプションを使用できます。

重要: レプリケーションスロットを使用していない場合、`wal_keep_segments`（または PostgreSQL バージョン 13.0 以降の `wal_keep_size`）の値に依存します。データベースのワークロードのピーク時に、`receive-wal` プロセスが遅れて同期が取れなくなる可能性があることに注意してください。予防措置として、Barman では現在、ユーザーが `--reset` オプションを使用してコマンドを手動で実行し、誤った仮定を行わないようにする必要があります。

11.8.2 複製スロット管理

`receive-wal` プロセスは、Barman の WAL アーカイビングプロシージャに必要なレプリケーションスロットを作成または削除するのにも役立ちます。

`--create-slot` オプションを使用すると、`slot_name` 構成オプションにちなんで名前付けたレプリケーションスロットが PostgreSQL サーバー上に作成されます。

`--drop-slot` を使用すると、以前のレプリケーションスロットが削除されます。

11.9 replication-status

`replication-status` コマンドは、Barman サーバーの `receive-wal` プロセス（設定されている場合）を含む、PostgreSQL サーバーに現在接続されているストリーミングクライアントのステータスを報告します。

次のようにコマンドを実行できます。

```
barman replication-status <server_name>
```

ちなみに: サーバー名前として `all` を使用して、すべてのサーバーのレプリカの完全なステータスレポートを要求できます。

機械可読な出力を得るには、`--minimal` オプションを使用できます。

11.10 show-servers

以下を使用して、特定のサーバーの構成パラメーターを表示できます。

```
barman show-servers <server_name>
```

ちなみに: では、サーバー名前として *all* を使用して完全な構成レポートを要求できます。

11.11 status

`status` コマンドは、サーバー名前として *all* を使用している場合、PostgreSQL サーバーまたはすべてのサーバーのライブ情報とステータスを表示します。

```
barman status <server_name>
```

11.12 switch-wal

このコマンドは、PostgreSQL サーバーを別のトランザクションログファイル（WAL）にスイッチ、現在のログファイルを閉じて受信し、アーカイブできるようにします。

```
barman switch-wal <server_name>
```

最後のトランザクションログファイルのスイッチ以降にトランザクションアクティビティがない場合、`--force` オプションを使用してスイッチを強制ニーズがあります。

`--archive` オプションは、Barman に `xlog` スイッチの後に WAL アーカイビングをトリガーするように要求します。デフォルトでは、30 秒のタイムアウトが強制されます（これは `--archive-timeout` で変更できます）。WAL ファイルが受信されない場合、エラーが返されます。

注釈: Barman 2.1 および 2.2 では、このコマンドは *switch-xlog* と呼ばれていました。PostgreSQL 10 以降との命名の一貫性のために名前が変更されました。

11.13 verify

`verify` コマンドは、バックアップの `backup_manifest` ファイルを使用して、それに対して `pg_verifybackup` を実行します。 `edb_notranlate_16` このコマンドは `pg_verifybackup <path_to_backup_manifest> -n` (PG = 13 で利用可能) を呼び出します `pg_verifybackup` バックアップサーバにインストールする必要があります。 `rsync` バックアップの場合、 `generate-manifest` コマンドで使用できます。

`backup_id` [バックアップ id shortcuts] { # backup-id-shortcuts } のいずれかを使用できます。

第12章 Backup commands

バックアップコマンドは、Barman のバックアップカタログに既に存在するバックアップに対して直接機能するコマンドです。

注釈: バックアップ ID は `barman list-backups <server_name>` で取得できることを忘れないでください

12.1 バックアップIDのショートカット

Barman では、特別なキーワードを使用して特定のバックアップを識別することができます。

- `last/latest` : カタログ内の最新のバックアップを識別します
- `first/oldest` : カタログ内の最も古いバックアップを識別します
- `last-failed` : カタログ内の最新の失敗したバックアップを識別します

これらのキーワードを Barman コマンドで使用すると、サーバーのバックアップの正確な ID を知らなくてもアクションを実行できます。例、次を発行できます。

```
barman delete <server_name> oldest
```

カタログで使用可能な最も古いバックアップを削除し、ディスクスペースを再利用します。

12.2 check-backup

バージョン 2.5 以降、`check-backup` コマンドを使用して、フルバックアップの一貫性に必要なすべての WAL ファイルが `barman` によって正しくアーカイブされていることを確認できます。

```
barman check-backup <server_name> <backup_id>
```

重要: このコマンドは、`cron` および `backup` オペレーションの終了時に自動的に呼び出されます。これは、通常の状態では、実行必要がないことを意味します。

バックアップのスタートから終了までの 1 つ以上の WAL ファイルがまだアーカイブされていない場合、`barman` はバックアップに `WAITING_FOR_WALS` の label を付けます。`cron` コマンドは、欠落している WAL ファイルがアーカイブされていることを引き続き確認し、バックアップに `DONE` の label を付けます。

バックアップの最後に最初に必要な WAL ファイルが見つからない場合、そのようなバックアップは **FAILED** としてマークされます。したがって、バックアップオペレーションを実行する前に、特にスタンバイサーバからバックアップする場合、WAL アーカイビングが（ストリーミングまたは `archive_command` を介して）適切に機能していることを確認することが重要です。

12.3 delete

特定のバックアップを削除するには：

```
barman delete <server_name> <backup_id>
```

`delete` コマンドは、**バックアップ ID のショートカット** を受け入れてバックアップを識別します。

12.4 keep

サーバーの保持ポリシーを超えて保持したいバックアップがある場合は、次の方法でアーカイブバックアップを `make` できます。

```
barman keep <server_name> <backup_id> [--target TARGET, --status, --release]
```

TARGET の可能な値は次のとおりです。

- **full**：バックアップはいつでも最新のポイントにリカバリするために使用できます時間。これを達成する保証に、Barman は、バックアップと後続のすべての WAL の一貫性。
- **standalone**：バックアップは、サーバーをそのバックアップが取られたときの状態。Barman は WAL のみを保持しますバックアップの一貫性を保証に必要です。

`--status` オプションが提供されている場合、Barman はバックアップのアーカイブステータスをレポートします。これは、アーカイブバックアップの場合は **full** または **standalone** のリカバリターゲットになり、アーカイブとしてフラグが設定されていないバックアップの場合は **nokeep** になります。

`--release` オプションが提供されている場合、Barman はこのバックアップから保持フラグをリリースします。これにより、アーカイブステータスが削除され、直接または保持ポリシーによって削除できる `make` になります。

バックアップがアーカイブバックアップとしてフラグ付けされると、Barman の動作は次のように変更されます。

- `barman delete` を使用して ID でそのバックアップを削除しようとする失敗します。
- 保持ポリシーは、そのバックアップを **OBSOLETE** およびしたがって、`barman cron` はそのバックアップを削除しません。
- そのバックアップに必要な WAL は永久に保持されます。もし指定されリカバリターゲットは **full** であり、その後のすべての WAL も* 保持されます。

これは、`barman keep <server_name> <backup_id> --release` で `keep` フラグを削除することで元に戻すことができます。

警告: *standalone* アーカイブバックアップがサーバーの保持ポリシーで不要になると、*barman cron* はそのバックアップと次の最新のバックアップの `begin_wal` 値の間の WAL を削除します。これは、ターゲットを *full* から *standalone* に変更することは安全ですが、最新の時点へのリカバリに必要な WAL がまだ維持される保証がないため、ターゲットを *standalone* から *full* に変更することは*安全*ではありません利用できます。

12.5 list-files

以下を使用して、特定のバックアップのファイル（ベースバックアップと必要な WAL ファイル）をリストできます。

```
barman list-files [--target TARGET_TYPE] <server_name> <backup_id>
```

`--target TARGET_TYPE` オプションを使用すると、特定のバックアップのリストのコンテンツを選択できます。

`TARGET_TYPE` の可能な値は次のとおりです。

- `data` : データファイルをリストします
- `standalone` : 必要な WAL を含むベースバックアップファイルをリストします ファイル
- `wal` : ベースバックアップの開始から次のスタート（またはログの終了まで）
- `full` : `data + wal` と同じ

`TARGET_TYPE` のデフォルト値は `standalone` です。

重要: *list-files* コマンドは外部ツールとの対話を容易にするため、Barman をアーカイビング手順に統合するのに非常に役立ちます。

12.6 recover

`backup` コマンドを使用してバックアップを実行した後、`recover` コマンドを使用してサーバー全体を回復します。

これは、次のようなコマンドを発行することで達成されます。

```
barman@backup$ barman recover <server_name> <backup_id> /path/to/recover/dir
```

重要: PostgreSQL インスタンスが実行されているターゲットデータディレクトリを使用して *recover* コマンドを発行しないでください。その場合は、リカバリを発行する前に必ず停止してください。これは、テーブルスペースディレクトリにも適用されます。

リカバリの実行が終了すると、選択したバックアップがローカルにリカバリされ、宛先パスには PostgreSQL インスタンスのスタートに使用できるデータディレクトリが含まれます。

重要: ユーザ *barman* としてこのコマンドを実行すると、データベーススーパーユーザになります。

バックアップの特定の ID は、`:ref:`list-backups`<list-backups>` コマンドを使用して取得できます。

重要: Barman は現在、PGDATA 内のシンボリックリンクを追跡しません (`pg_tblspc` 内のテーブルスペースを除く)。システム管理者は、シンボリックリンクを追跡し、オリジナルの場所に復元する必要がある場合に備えて、それらを災害リカバリ計画/手順に追加することをお勧めします。

リカバリコマンドには、コマンドの動作を変更するいくつかのオプションがあります。

12.6.1 リモートリカバリ

リカバリコマンドの呼び出しに `--remote-ssh-command <COMMAND>` オプションを追加します。これを行うと、提供されたコマンドを使用してリモートホストに接続し、Barman がリモートサーバでコピーを実行できるようになります。

注釈: リモートホストでリカバリを実行するには、*postgres* ユーザを使用することをお勧めします。

重要: PostgreSQL インスタンスが実行されているターゲットデータディレクトリを使用して *recover* コマンドを発行しないでください。その場合は、リカバリを発行する前に必ず停止してください。これは、テーブルスペースディレクトリにも適用されます。

リモートリカバリの既知の制限は次のとおりです。

- Barman は、システムの一時ディレクトリに少なくとも 4GB のフリースペースを必要とします
`:ref:`get-wal`<get-wal>` コマンドが指定されていない限り Barman 構成の `recovery_option` パラメータ。
- Barman とリモートホスト間の SSH 接続には、* 公開キー交換認証メソッド* リモートユーザはディレクトリ構造を作成できる必要があります* 宛先ディレクトリのバックアップの。* リモートサーバに十分なフリースペースが必要ですベースバックアップとリカバリに必要な WAL ファイルを含むために。

12.6.2 表領域の再マッピング

Barman は、`-` テーブルスペースオプションを指定した `recover` コマンドを使用して、1 つ以上のテーブルスペースを自動的に再マッピングできます。このオプションは、`NAME:DIRECTORY` フォーマットを使用して、引数として値のペアを受け入れます。

- `NAME` はテーブルスペースの識別子です
- `DIRECTORY` は、テーブルスペースの新しい宛先パスです

宛先ディレクトリが存在しない場合、Barman はそれを作成しようとします（必要な許可がある場合）。

12.6.3 ポイントインタイムリカバリ

Barman は PostgreSQL の Point-in-Time Recovery (PITR) をラップするため、タイムスタンプ、復元 label、またはトランザクション ID のいずれかでリカバリターゲットを指定できます。

重要: 特定のバックアップの最も早い PITR は、ベースバックアップ自分自身の終わりです。バックアップのスタートから終了までの任意の時点で回復する場合は、以前のバックアップを使用する必要があります。Barman 2.3 からは、`--target-immediate` オプション（PostgreSQL 9.4 以降でのみ使用可能）を使用して、一貫性に達したときにリカバリを終了できます。

リカバリターゲットは、次の相互に排他的なオプションのいずれかを使用して指定できます。

- `--target-time TARGET_TIME` : タイムスタンプを指定します
- `--target-xid TARGET_XID` : トランザクション ID を指定します
- `--target-lsn TARGET_LSN` : ログシーケンス番号 (LSN) を指定するには- PostgreSQL 10 以降が必要です
- `--target-name TARGET_NAME` : 名前付けリストアポイントを指定します `pg_create_restore_point` (名前) で以前に作成された ファンクション¹
- `--target-immediate` : 一貫性のある状態に達するとリカバリが終了します (つまり、ベースバックアッププロセスの終了です)²

重要: バックアップの終了後の時間、XID、LSN .. `IMPORTANT::` を介したリカバリターゲット。バックアップのスタートと終了の間の時点まで回復する場合は、カタログ内の以前のバックアップから回復する必要があります。

`--exclusive` オプションを使用して、リカバリターゲットの直前または直後に停止するかどうかを指定できます。

¹ PostgreSQL 9.1 以降でのみ利用可能

² PostgreSQL 9.4 以降でのみ利用可能

Barman では、`target-tli` オプションを使用して、リカバリのターゲットタイムラインを指定できます。タイムラインの概念はこのドキュメントの範囲を超えています; 「スタートする前に」セクションで説明したように、PostgreSQL の文書で詳細を確認できます。

Barman 2.4 は、`--target-action` オプションのサポートを導入し、次の値を受け入れます。

- `shutdown` : リカバリターゲットに到達すると、PostgreSQL はシャットダウンされます³
- `pause` : リカバリターゲットに到達すると、PostgreSQL は一時停止状態で開始されます ユーザーがインスタンス⁴を検査できる状態
- `promote` : リカバリターゲットに到達すると、PostgreSQL はリカバリを終了します マスタ⁵として昇格されます

重要: デフォルトでは、ターゲットアクションは定義されていません (後方互換性のため)。`--target-action` オプションでは、Point In Time Recovery ターゲットを指定する必要があります。

上記の設定の詳細については、[PostgreSQL documentation on recovery target settings](#) を参照してください。

また、Barman 2.4 は、`recover` コマンドに `--standby-mode` オプションを追加します。指定されている場合、`standby.signal` ファイル (PostgreSQL 12 から) を作成するか、生成されたリカバリ構成に `standby_mode = on` を追加することにより、リカバリされたインスタンスをスタンバイとして適切に構成します。スタンバイモードの詳細については、PostgreSQL の文書を参照してください。

12.6.4 Barman サーバーから WAL を取得する

`barman recover` コマンドは、オプションで、リカバリ中に Barman から WAL をフェッチするように PostgreSQL を設定できます。これは、`get-wal` section で説明されているように、`recovery_options` グローバル/サーバー構成オプションを `'get-wal'` に設定することで有効になります。`recovery_options` が設定されていないか空の場合、Barman は代わりに `barman recover` コマンドの実行中にリカバリに必要な WAL をコピーします。

`--get-wal` および `--no-get-wal` オプションを使用して、`recovery_options` で定義された動作をオーバーライドできます。`--get-wal` を `barman recover` と組み合わせて使用して、Barman サーバーからの WAL のフェッチを有効にします。あるいは、`--no-get-wal` を使用して無効にします。

³ PostgreSQL 9.5 以降でのみ利用可能

⁴ PostgreSQL 9.1 以降でのみ利用可能

⁵ PostgreSQL 9.5 以降でのみ利用可能

12.7 show-backup

以下を使用して、特定のサーバーの特定のバックアップに関するすべての利用可能な情報を取得できます。

```
barman show-backup <server_name> <backup_id>
```

show-backup コマンドは、 **バックアップ ID のショートカット** を受け入れてバックアップを識別します。

第13章 Features in detail

このセクションでは、いくつかの Barman 機能を紹介し、それらの適用性とそれらを使用するために必要な構成について説明します。

このリストは包括的なものではありません。多くのシナリオが Barman 構成で動作するように作成できるためです。それでも、一般的なパターンを議論することは有用です。

13.1 バックアップ機能

13.1.1 増分バックアップ

Barman は **ファイルレベルの増分バックアップ**を実装しています。増分バックアップは、特定の PostgreSQL サーバーのカatalogで利用可能な最新のフルバックアップからのデータ変更のみを保存する完全定期バックアップの一種です。WAL の継続的アーカイビングによって実装される差分バックアップと混同しないでください。

注釈: ブロックレベルの増分バックアップは、将来のバージョンで利用可能になる予定です。

重要: 現時点では、`reuse_backup` オプションは `postgres` バックアップメソッドでは使用できません。

Barman の増分バックアップの主な目標は次のとおりです。

- フルバックアッププロセスにかかる時間を短縮する
- いくつかの定期的なバックアップで占有されるディスクスペースを削減します（**データ重複排除**）

この機能は `rsync` と `hard links` に大きく依存しているため、基盤となるオペレーティングシステムとバックアップデータが存在するファイルシステムの両方でサポートする必要があります。

主な概念は、後続のベースバックアップが、前回のバックアップ以降に変更されていないファイルを共有し、ディスク使用量を適切に節約することです。これは、VLDB コンテキスト、および読み取り専用の履歴テーブルの割合が高いデータベースに特に当てはまります。

Barman は、`barman backup` コマンドを透過的に管理する `reuse_backup` と呼ばれるグローバル/サーバオプションを介して増分バックアップを実装します。次の3つの値を受け入れます。

- `off` : 標準フルバックアップ（デフォルト）

- **link** : サーバーの最後のバックアップを再利用する増分バックアップ変更されていないファイルのハードリンクを作成する（バックアップスペース用）および時間短縮）
- **copy** : サーバーの最後のバックアップを再利用する増分バックアップ変更されていないファイルのコピーを作成する（バックアップ時間のためだけに）削減）

最も一般的なシナリオは、次のように `reuse_backup` を `link` に設定することです。

```
reuse_backup = link
```

これをグローバルレベルで設定すると、すべてのサーバーの増分バックアップが自動的に有効になります。

最後に、ユーザーはノートコマンドの `--reuse-backup` ランタイムオプションを使用して、`reuse_backup` オプションの設定をオーバーライドできます。同様に、ランタイムオプションは、`off`、`link`、および `copy` の 3 つの値を受け入れます。例、次のように一時的な増分バックアップを実行できます。

```
barman backup --reuse-backup=link <server_name>
```

13.1.2 帯域幅の使用を制限する

1 秒あたりの最大キロバイト数を指定することにより、`bandwidth_limit` オプション（グローバル/サーバーごと）を使用して I/O 帯域幅の使用を制限することができます。デフォルトでは、制限なしを意味する 0 に設定されています。

重要: `bandwidth_limit` オプションは、Postgres 9.4 以降の *postgres* バックアップメソッドでサポートされていますが、`tablespace_bandwidth_limit` オプションは *rsync* を使用する場合にのみ使用できます

複数のテーブルスペースがあり、1 つ以上のテーブルスペースでバックアップ手順の I/O ワークロードを制限する場合、`tablespace_bandwidth_limit` オプション（グローバル/サーバーごと）を使用できます。

```
tablespace_bandwidth_limit = tname:bwlimit[, tname:bwlimit, ...]
```

このオプションは、テーブルスペース名前と帯域幅制限（キロバイト/秒）で構成されるコンマ区切りのペアのリストを受け入れます。

サーバーをバックアップするとき、Barman は上記のオプションで既存のテーブルスペースを見つけようとします。見つかった場合、指定された帯域幅制限が適用されます。そうでない場合は、そのサーバーのデフォルトの帯域幅制限が適用されます。

13.1.3 ネットワーク圧縮

圧縮を使用して、転送されるデータのサイズを減らすことができます。`network_compression` オプション（グローバル/サーバーごと）を使用して有効にできます。

重要: `network_compression` オプションは、`postgres` バックアップメソッドでは使用できません。

```
network_compression = true|false
```

このオプションを `true` に設定すると、ネットワーク転送中のデータ圧縮が有効になります（バックアップとリカバリの両方）。デフォルトでは、`false` に設定されています。

13.1.4 並行バックアップとスタンバイからのバックアップ

通常、バックアップ操作中に、Barman は `_exclusive backup_` に PostgreSQL ネイティブ関数 `pg_start_backup` および `pg_stop_backup` を使用します。これらの操作は、読み取り専用のスタンバイサーバでは許可されていません。

また、主に Barman 設定パラメータを使用して、9.2 以降のデータベースサーバから PostgreSQL のバックアップを **同時方式** で実行できます。¹

これにより、Barman の新しいアーキテクチャシナリオが導入されます。**スタンバイサーバからのバックアップ**、`rsync` を使用。

重要: !!!同時バックアップでは、PostgreSQL 9.2、9.3、9.4、および 9.5 のユーザーがクラスターのすべての PostgreSQL サーバに `pgespresso` オープンソース拡張機能をインストールする必要があります。詳細情報とソースコードについては、[pgespresso extension website](#) をご覧ください。Barman は、PostgreSQL 9.6 で導入された新しい API をサポートしています。これにより、このバージョンの PostgreSQL から同時バックアップを実行するための `pgespresso` 拡張機能の要件がなくなります。

デフォルトでは、下位互換性のため、`backup_options` は `exclusive_backup` に透過的に設定されています。PostgreSQL 9.6 以降のバージョンのユーザーは、`backup_options` を `concurrent_backup` に設定する必要があります。

重要: PostgreSQL 9.5 がコミュニティによって EOL として宣言されると、Barman はデフォルトで `backup_options` を `concurrent_backup` に設定します。`pgespresso` のサポートは終了します。

`backup_options` が `concurrent_backup` に設定されている場合、Barman はサーバの同時バックアップモードをアクティブにし、次の 2 つのシンプルルールに従います。

¹ 同時バックアップは、バージョン 9.2 以降、*streaming レプリケーション protocol*（例、`pg_basebackup` などのツールを使用）を介して PostgreSQL で使用可能な技術です。

重要: 同時バックアップの場合、現在、Barman はフルバックアップの閉じている WAL ファイルが実際に出荷されたかどうかを判断できません PostgreSQL 自分自身が WAL ファイルが正しくアーカイブされることを確認する排他的バックアップの反対です。その WAL ファイルが Barman によって受信およびアーカイブされるまで、フルバックアップは一貫性があると見なされないことに注意してください。Barman 2.5 は、`check-backup` コマンド（`cron` コマンドによって実行される通常のメンテナンスジョブのパート）によって管理される `WAITING_FOR_WALS` と呼ばれる新しい状態を導入します。Barman 2.10 以降では、`barman backup` コマンドで `--wait` オプションを使用できます。

13.1.5 スタンバイからのバックアップに関する現在の制限

現在、Barman では、バックアップデータ（ベースバックアップと WAL ファイル）が 1 つのサーバーからのみ取得されることを要求しています。したがって、スタンバイからバックアップする場合は、スタンバイサーバをポイントする必要があります。

- `conninfo`
- `streaming_conninfo`、`postgres` を `backup_method` として使用する場合、および/または WAL ストリーミングに依存する場合
- `rsync` を `backup_method` として使用する場合、`ssh_command`

重要: Barman 2.8 以降、スタンバイからのバックアップは PostgreSQL 9.4 以上でのみサポートされています（バージョン 9.4 および 9.5 には `pgespresso` が必要です）。9.2 および 9.3 のサポートは非推奨です。

推奨される最も簡単な方法は、PostgreSQL 9.4 を必要とするスタンバイから直接レプリケーションスロットを使用して WAL ストリーミングをセットアップすることです。これは次のことを意味します。

「WAL ストリーミング」で説明されているように、`'streaming_archiver = on'` を設定します「複製スロット」を含むセクション `archiver = on` を無効にする

または、PostgreSQL 9.5 から、`archive_command` を `archive_mode = always` とともに使用し、WAL ストリーミングを無効にすることによってのみ、スタンバイからアーカイブすることを決定できます。

注釈: 残念ながら、Barman が WAL 重複チェックと [an undocumented behaviours in all versions of PostgreSQL](#) を実行する方法により、現在、WAL アーカイビングとスタンバイからのストリーミングの両方を有効にすることはできません。

13.1.6 即時チェックポイント

バックアップを開始する前に、Barman は追加のワークロードを生成するチェックポイントを要求します。通常、そのチェックポイントは、PostgreSQL サーバーのワークロード制御の設定に従って調整されます。つまり、バックアップが遅延する可能性があります。

このデフォルトの動作は、`immediate_checkpoint` 構成グローバル/サーバオプション（デフォルトで `false` に設定）を使用して変更できます。

`immediate_checkpoint` が `true` に設定されている場合、PostgreSQL はワークロードを制限しようとせず、チェックポイントは最大速度で発生し、できるだけ早くバックアップを開始します。

次の 2 つのオプションのいずれかで `barman backup` を発行することにより、いつでも構成オプションの動作をオーバーライドできます。

- `--immediate-checkpoint`。即時チェックポイントを強制します。
- `--no-immediate-checkpoint`、チェックポイントの待機を強制します起こる。

13.1.7 ローカルバックアップ

DISCLAIMER Barman と PostgreSQL は同じサーバーに存在し、同じ単一障害点のパートであるため、この機能は稼働での使用にはお勧めしません。一部の EnterpriseDB のお客様は、特定の状況で、そして最も重要なこととして、会社が提供する 24 時間 365 日の稼働サービスで使用されるローカルバックアップのサポートを Barman に追加することを要求しています。現在、この機能を使用するには、ソースからのインストールが必要です。または、`postgres` ユーザの環境を、権限とロギングおよび `cron` 構成の観点からカスタマイズする必要があります。

特別な状況では、Barman は PostgreSQL インスタンスが存在する同じサーバーにインストールでき、バックアップデータは PGDATA と、該当する場合はテーブルスペースとは別のボリュームに保存されます。通常、これらのボリュームは、NFS などのファイルシステムを備えたネットワークストレージアプライアンスに存在します。

このアーキテクチャは、EnterpriseDB によって承認されていません。PostgreSQL のビジネス継続性を強化し、RPO と RTO の点でより良い結果を得るために、EnterpriseDB は、Barman のリモートインストールで、レプリケーションとモニタリングの目的で監視ウィットネスサーバのように動作できるシェアードナッシングアーキテクチャを推奨します。

ローカルバックアップの唯一の要件は、Barman が PostgreSQL サーバーと同じユーザ（通常は `postgres`）で実行されることです。コミュニティパッケージソフトがデフォルトで `barman` ユーザの下に Barman をインストールすることを考えると、このユースケースでは以下を含むマニュアルインストール手順が必要です。

- `cron` 構成
- `logrotate` を含むログ構成

Barman で特定のサーバーのローカルバックアップを使用オーダーには、`backup_method` を `local-rsync` に設定する必要があります。この機能は、代わりに SSH に依存し、リモートで動作する `rsync` と同等です。`local-rsync` では、ローカルで `rsync` コマンドを発行してファイルシステムのコピーが実行されます（このため、Barman は PostgreSQL と同じユーザで実行する必要があります）。

local-pg13 名前付けのサーバーのローカルバックアップの構成の抜粋は次のとおりです。

```
[local-pg13]
description = "Local PostgreSQL 13"
backup_method = local-rsync
...
```

13.2 アーカイブ機能

13.2.1 WAL 圧縮

設定ファイルで `compression` オプションが設定されている場合、`barman cron` コマンドは WAL ファイルを圧縮します。このオプションでは、次の 5 つの値を使用できます。

- `bzip2` : Bzip2 圧縮用 (`bzip2` ユーティリティが必要)
- `gzip` : Gzip 圧縮用 (`gzip` ユーティリティが必要)
- `pybzip2` : Bzip2 圧縮用 (Python の内部圧縮モジュールを使用)
- `pygzip` : Gzip 圧縮用 (Python の内部圧縮モジュールを使用)
- `pigz` : Pigz 圧縮の場合 (`pigz` ユーティリティが必要)
- `custom` : カスタム圧縮の場合、設定する必要があります以下のオプションも：
- `custom_compression_filter` : 圧縮フィルター
- `custom_decompression_filter` : 解凍フィルター
- `custom_compression_magic` : カスタム圧縮 wal ファイルを識別する 16 進文字列

注： `pybzip2` および `pygzip` を除くすべてのメソッドには、「`barman archive-wal`」は新しいプロセスをフォークします。

13.2.2 同期 WAL ストリーミング

重要: この機能は、PostgreSQL 9.5 以降でのみ利用可能です。

Barman は、同期的スタンバイサーバのようにトランザクション WAL ファイルを収集することにより、目標復旧ポイントをゼロに減らすこともできます。

このようなシナリオを構成するには、[PostgreSQL ストリーミング接続](#) を介して WAL をアーカイブするように Barman サーバーを構成する必要があり、`receive-wal` プロセスは PostgreSQL サーバーの同期的スタンバイとして認識される必要があります。

まず、`show-servers` コマンドで Barman `receive-wal` プロセスのアプリケーション名前を取得する必要があります。

```
barman@backup$ barman show-servers pg|grep streaming_archiver_name
streaming_archiver_name: barman_receive_wal
```

次に、アプリケーション名前を同期的スタンバイとして `postgresql.conf` ファイルに追加する必要があります。

```
synchronous_standby_names = barman_receive_wal
```

重要: これは、Barman が同期的スタンバイノードとして適格であることを示すための構成の例にすぎません。Barman のみを使用することは推奨していません。このトピックの詳細については、PostgreSQL 文書の [Synchronous Replication](#) を参照してください。

設定をリロードするには、PostgreSQL サーバーを再起動ニーズがあります。

サーバーが正しく構成されている場合、`replication-status` コマンドは `receive_wal` プロセスを同期的ストリーミングクライアントとして表示する必要があります。

```
[root@backup ~]# barman replication-status pg
Status of streaming clients for server pg:
Current xlog location on master: 0/90000098
Number of streaming clients: 1

1. #1 Sync WAL streamer
Application name: barman_receive_wal
Sync stage      : 3/3 Remote write
Communication    : TCP/IP
IP Address      : 139.59.135.32 / Port: 58262 / Host: -
User name       : streaming_barman
Current state    : streaming (sync)
Replication slot: barman
WAL sender PID  : 2501
Started at      : 2016-09-16 10:33:01.725883+00:00
Sent location   : 0/90000098 (diff: 0 B)
Write location  : 0/90000098 (diff: 0 B)
Flush location  : 0/90000098 (diff: 0 B)
```

13.3 カタログ管理機能

13.3.1 最小の冗長性の安全性

デフォルトで 0 に設定された `minimum_redundancy` と呼ばれるグローバル/サーバーごとの構成オプションを使用して、PostgreSQL サーバーの定期的なバックアップの最小数を定義できます。

この値を 0 より大きい任意の数に設定することにより、Barman はサーバーカタログに少なくともその数のバックアップがあることを常に確認します。

これにより、誤った `barman delete` 操作から保護されます。

重要: 保持ポリシー設定が最小冗長要件と競合しないようにしてください。このトピックに関するメッセージについては、定期的に Barman のログを確認してください。

13.3.2 保持ポリシー

Barman は、バックアップの **保持ポリシー** をサポートしています。

バックアップ保存ポリシーは、リカバリ手順のためにバックアップおよび関連するアーカイブログ (Write Ahead Log セグメント) を保持する必要がある期間を決定するユーザー定義のポリシーです。

ユーザーの要求に基づいて、Barman は現在の保持ポリシーを満たすために必要な定期的なバックアップと、それらのバックアップの完全なリカバリに必要なアーカイブされた WAL ファイルを保持します。

Barman ユーザーは、**バックアップの冗長性** (定期的なバックアップの数) または **リカバリ期間** (期間) の観点から保持ポリシーを定義できます。

冗長性に基づく保持ポリシー

：冗長性ベースの保持ポリシーでは、ユーザは保持する定期的なバックアップの数を決定します。冗長性ベースの保持ポリシーは、リカバリウィンドウを使用する保持ポリシーとは対照的です。

リカバリウィンドウに基づく保持ポリシー

：リカバリウィンドウは、Barman バックアップ保存ポリシーの一種です DBA は期間を指定し、Barman は、リカバリウィンドウ中の任意の時点へのポイントインタイムリカバリに必要なバックアップやアーカイブ WAL ファイルの保持を保証します。インターバルは常に現在の時刻で終了し、ユーザが指定した日数だけ過去に戻ります。例、保持ポリシーが 7 日間のリカバリウィンドウに設定されており、現在の時刻が金曜日の午前 9:30 である場合、Barman は、特定のポイントインタイムリカバリを午前 9:30 に戻すために必要なバックアップを保持します前金曜日。

13.3.3 スコープ

保持ポリシーは以下に対して定義できます。

重要: 時間ディメンションでは、定期的なバックアップの時間枠にアーカイブログを含める必要があります。

ここには、2つの典型的なユースケースがあります。完全または部分的なポイントインタイムリカバリです。

フルポイントインタイムリカバリシナリオ：

：ベースバックアップとアーカイブログは同じ保持ポリシーを共有するため、最初に利用可能なバックアップからいつでも回復できます。

部分的なポイントインタイムリカバリシナリオ：

：ベースバックアップ保存ポリシーはアーカイブログのポリシーよりも広いため、例、ユーザーは過去6か月間の完全な週単位のバックアップを保持できますが、過去4週間はアーカイブログを保存できます定期的な週4回のバックアップ)。

重要: 現在、Barman は `wal_retention_policy` オプションを `main` に制限することにより、****完全時点リカバリ****シナリオのみを実装しています。

13.3.4 仕組み

Barman の保持ポリシーは次のとおりです。

- **自動化**：barman cron により実施
- **マニュアル**：Barman は古いバックアップを報告するだけで、それらを削除する

重要: 現在、Barman はマニュアル強制を実装していません。この機能は将来のバージョンで利用可能になる予定です。

13.3.5 設定と構文

保持ポリシーは、次の構成オプションを使用して定義できます。

- `retention_policy`：ベースバックアップの保持用
- `wal_retention_policy`：アーカイブログの保持用
- `retention_policy_mode`：auto のみに設定できます（保持ポリシーは barman cron コマンドによって自動的に実施されます）

これらの構成オプションは、グローバルレベルとサーバーレベルの両方で定義でき、ユーザーはマルチサーバー環境で最大限の柔軟性を実現できます。

retention_policy の構文

retention_policy を介したベースバックアップ保存ポリシーの一般的な構文は次のとおりです。

```
retention_policy = {REDUNDANCY value | RECOVERY WINDOW OF value {DAYS | WEEKS | MONTHS}}
```

どこで：

- 構文は大文字と小文字を区別しません
- value は整数であり、> 0
- 冗長性保持ポリシー の場合：
 - value はサーバーの最小以上でなければなりません冗長レベル（その値が割り当てられていない場合、ワーニングが生成されます）
 - 最初の有効なバックアップは、逆の値のバックアップです順序付けられた時シリーズ
- リカバリウィンドウポリシー の場合：
 - 回復可能性のポイント：現在の時間-ウィンドウ
 - 最初の有効なバックアップは、最初に使用可能なバックアップです回復可能性のポイント; 逆順の時シリーズはサーバー以上でなければなりません最小冗長レベル（その値に割り当てられていない場合およびワーニングが生成されます）

デフォルトでは、retention_policy は空です（保持は強制されません）。

wal_retention_policy の構文

現在、wal_retention_policy に許可されている値は、アーカイブログの保持ポリシーをベースバックアップの保持ポリシーにマップする特別な値 main のみです。

13.4 フックスクリプト

データベース管理者は、Barman を使用して、次の 2 つのイベントでフックスクリプトを実行できます。

- バックアップの前後
- バックアップの削除の前後
- WAL ファイルがアーカイブされる前と後
- WAL ファイルが削除される前後

Barman が管理できるフックスクリプトには 2 つのタイプがあります。

- 標準フックスクリプト
- フックスクリプトのリトライ

これら 2 種類のフックスクリプトの唯一の違いは、Barman はリターンコードをチェックせずに標準のフックスクリプトを 1 回だけ実行するのに対して、リトライフックスクリプトはリターンコードに応じて複数回実行される場合があります。

具体的には、リトライフックスクリプトを実行すると、Barman は、スクリプトが SUCCESS（標準のリターンコード 0）、ABORT_CONTINUE（リターンコード 62）、または ABORT_STOP（リターンコード 63）を返すまで、リターンコードをチェックして無期限に再試行します。Barman は、他のリターンコードを、再試行する一時的な障害として扱います。ユーザーはより強力になります。フックスクリプトは、障害が一時的なものかどうかを指定することで、ワークフローを制御できます。また、「pre」フックスクリプトの場合、ABORT_STOP を返すことにより、ユーザーは Barman にメインオペレーションを失敗して中断するように要求できます。

フックスクリプトは、次のオーダーで実行されます。

1. 標準の 'pre' フックスクリプト（存在する場合） 2. リトライリトライの 'pre' フックスクリプト（存在する場合） 3. 実際のイベント（つまり、バックアップオペレーション、または WAL アーカイビング）。ABORT_STOP で中断 4. リトライの 'post' フックスクリプト（存在する場合） 5. 標準の 'post' フックスクリプト（存在する場合）

フックスクリプトによって生成された出力は、Barman のログファイルに書き込まれます。

注釈: 現在、ABORT_STOP は 'post' フックスクリプトのリトライによって無視されます。これらの場合、追加のワーニングをロギングする以外は、ABORT_STOP は ABORT_CONTINUE のように動作します。

13.4.1 バックアップスクリプト

これらのスクリプトは、次のグローバル構成オプションで構成できます（サーバーごとにオーバーライドできます）。

- **pre_backup_script** : ベースバックアップの前に実行されるフックスクリプト、一度だけ、終了コードのチェックなし
- **pre_backup_retry_script** : フックスクリプトのリトライベースバックアップ、成功またはアボートまで繰り返し
- **post_backup_retry_script** : 後に実行されるリトライフックスクリプトベースバックアップ、成功またはアボートまで繰り返し
- **post_backup_script** : ベースバックアップ後に実行されるフックスクリプト、一度だけ、終了コードのチェックなし

スクリプト定義はシェルに渡され、任意の終了コードを結果ことができます。リトライスクリプトの場合のみ、Barman はリターンコードをチェックします（hook script section を参照）。

シェル環境には次の変数が含まれます。

- `BARMAN_BACKUP_DIR` : バックアップ先ディレクトリ
- `BARMAN_BACKUP_ID` : バックアップの ID
- `BARMAN_CONFIGURATION` : Barman が使用する構成ファイル
- `BARMAN_ERROR` : エラーメッセージ (存在する場合) (`post` フェーズのみ)
- `BARMAN_PHASE` : スクリプトのフェーズ、`pre` または `post`
- `BARMAN_PREVIOUS_ID` : 以前のバックアップの ID (存在する場合)
- `BARMAN_RETRY` : リトライスクリプトの場合は 1、そうでない場合は 0
- `BARMAN_SERVER` : サーバーの名前
- `BARMAN_STATUS` : バックアップのステータス
- `BARMAN_VERSION` : Barman のバージョン

13.4.2 バックアップ削除スクリプト

バージョン 2.4 には、バックアップ前およびバックアップ後の削除スクリプトが導入されています。

以前のスクリプトと同様に、グローバル構成オプション内でバックアップ削除スクリプトを構成でき、サーバーごとにそれらを上書きできます。

- `pre_delete_script` : 削除前に起動されるフックスクリプトバックアップの 1 回のみ、終了コードのチェックなし
- `pre_delete_retry_script` : 以前に実行されたフックスクリプトのリトライバックアップの削除、成功またはアボートまで繰り返し
- `post_delete_retry_script` : 後に実行されるフックスクリプトのリトライバックアップの削除、成功またはアボートまで繰り返し
- `post_delete_script` : 削除後に起動されるフックスクリプトバックアップの 1 回のみ、終了コードのチェックなし

スクリプトはシェルを介して実行され、任意の終了コードを結果ことができます。リトライスクリプトの場合のみ、Barman はリターンコードをチェックします (上のセクションを参照)。

削除スクリプトは、バックアップスクリプトと同じ環境変数にプラスで、次を使用します。

- `BARMAN_NEXT_ID` : 次のバックアップの ID (存在する場合)

13.4.3 WAL アーカイブスクリプト

バックアップスクリプトと同様に、アーカイブスクリプトはグローバル構成オプションで構成できます（サーバーごとにオーバーライドできます）。

- **pre_archive_script** : WAL ファイルの前に実行されるフックスクリプトメンテナンス（通常は `barman cron`）によってアーカイブされ、一度だけ、終了コードを確認してください
- **pre_archive_retry_script** : の前に実行されるフックスクリプトのリトライ WAL ファイルはメンテナンス（通常は `barman cron`）によってアーカイブされます。成功するか中止されるまで繰り返し
- **post_archive_retry_script** : 後に実行されるフックスクリプトのリトライ WAL ファイルはメンテナンスによってアーカイブされます。成功または中止
- **post_archive_script** : WAL ファイルの実行後に実行されるフックスクリプトメンテナンスによってアーカイブされ、終了コードはチェックされずに 1 回だけ

スクリプトはシェルを介して実行され、任意の終了コードを結果ことができます。リトライスクリプトの場合のみ、Barman はリターンコードをチェックします（上のセクションを参照）。

アーカイブスクリプトは、バックアップスクリプトといくつかの環境変数を共有します。

- **BARMAN_CONFIGURATION** : Barman が使用する構成ファイル
- **BARMAN_ERROR** : エラーメッセージ（存在する場合）（`post` フェーズのみ）
- **BARMAN_PHASE** : スクリプトのフェーズ、`pre` または `post`
- **BARMAN_SERVER** : サーバーの名前

次の変数は、アーカイブスクリプトに固有です。

- **BARMAN_SEGMENT** : WAL ファイルの名前
- **BARMAN_FILE** : WAL ファイルのフルパス
- **BARMAN_SIZE** : WAL ファイルのサイズ
- **BARMAN_TIMESTAMP** : WAL ファイルのタイムスタンプ
- **BARMAN_COMPRESSION** : WAL ファイルに使用される圧縮のタイプ

13.4.4 WAL 削除スクリプト

バージョン 2.4 では、WAL 削除スクリプトの前後に導入されています。

他のフックスクリプトと同様に、`wal delete` スクリプトはグローバル構成オプションで構成でき、サーバーごとにそれらをオーバーライドできます。

- **pre_wal_delete_script** : 前に実行されたフックスクリプト WAL ファイルの削除
- **pre_wal_delete_retry_script** : 以前に実行されたフックスクリプトのリトライ WAL ファイルの削除、成功するまで繰り返し または中止されました

- `post_wal_delete_retry_script` : 後に実行されるフックスクリプトのリトライ WAL ファイルの削除、成功するまで繰り返し または中止されました
- `post_wal_delete_script` : 後に実行されるフックスクリプト WAL ファイルの削除

スクリプトはシェルを介して実行され、任意の終了コードを結果ことができます。リトライスクリプトの場合のみ、Barman はリターンコードをチェックします（上のセクションを参照）。

WAL 削除スクリプトは、WAL アーカイブスクリプトと同じ環境変数を使用します。

13.4.5 回復スクリプト

バージョン **2.4** では、回復前およびリカバリのスクリプトが導入されています。

以前のスクリプトとして、リカバリスクリプトはグローバル構成オプション内で構成でき、サーバーごとにそれらをオーバーライドできます。

- `pre_recovery_script` : リカバリの前に起動されるフックスクリプトバックアップの 1 回のみ、終了コードのチェックなし
- `pre_recovery_retry_script` : 以前に実行されたフックスクリプトのリトライバックアップのリカバリ、成功またはアボートまで繰り返し
- `post_recovery_retry_script` : 後に実行されるフックスクリプトのリトライバックアップのリカバリ、成功またはアボートまで繰り返し
- `post_recovery_script` : リカバリ後に起動されるフックスクリプトバックアップの 1 回のみ、終了コードのチェックなし

スクリプトはシェルを介して実行され、任意の終了コードを結果ことができます。リトライスクリプトの場合のみ、Barman はリターンコードをチェックします（上のセクションを参照）。

回復スクリプトは、バックアップスクリプトと同じ環境変数にプラスで、次を使用します。

- `BARMAN_DESTINATION_DIRECTORY` : 新しいインスタンスがあるディレクトリ 回復する
- `BARMAN_TABLESPACES` : テーブルスペース再配置マップ (JSON、存在する場合)
- `BARMAN_REMOTE_COMMAND` : 使用されたセキュアシェルコマンドリカバリによって (存在する場合)
- `BARMAN_RECOVER_OPTIONS` : 追加オプションのリカバリ (JSON、存在する場合)

13.5 カスタマイズ

13.5.1 ファイルディレクトリをロックする

Barman では、`barman_lock_directory` グローバルオプションを使用して、ロックファイルのディレクトリを指定できます。

ロックファイルは、グローバルレベルとサーバーレベルでの同時作業を調整するために使用されます（例、cron 操作、バックアップ操作、WAL アーカイブへのアクセスなど）。

デフォルトでは（下位互換性のため）、`barman_lock_directory` は `barman_home` に設定されます。

ちなみに: ユーザーは、ランタイム変数データ専用のディレクトリ（`/var/run/barman` など）などの volatile パーティション内のディレクトリを使用することをお勧めします。

13.5.2 バイナリ経路

バージョン 1.6.0 以降、Barman では、グローバル/サーバオプション `path_prefix` を使用して、Barman が実行可能ファイルを検索する 1 つ以上のディレクトリを指定できます。

`path_prefix` が提供される場合、コロンで区切られた 1 つ以上のディレクトリのリストを含む必要があります。Barman はまずこれらのディレクトリ内を検索し、次に `PATH` 環境変数で指定されたディレクトリ内を検索します。

デフォルトでは、`path_prefix` オプションは空です。

13.6 クラスター管理システムとの統合

Barman は、スタンバイサーバ（ストリーミングレプリケーションまたは従来のファイルベースのログシッピング）および (<https://www.repmgr.org/>) [repmgr] などの高可用性ツールとの統合用に設計されています。

アーキテクチャのビューから、WAL ファイルを直接 Barman サーバーにアーカイブするように PostgreSQL を構成する必要があります。Barman は、`get-wal` フレームワークのおかげで、WAL ハブとしても使用できます。この目的のために、すべてのスタンバイサーバで `barman-cli` パッケージのパートである `barman-wal-restore` スクリプトを使用できます。

`replication-status` コマンドを使用すると、管理対象サーバ、特にホットスタンバイサーバや WAL ストリーマーに接続されているストリーミングクライアントに関する情報を取得できます。

13.7 並列ジョブ

デフォルトでは、Barman はバックアップ操作とリカバリー操作の両方でファイルコピーに 1 つのワーカーのみを使用します。バージョン 2.2 以降では、ファイルコピーを実行するワーカーの数をカスタマイズします。この場合、コピーされるファイルは、すべての並列ワーカーに均等に分散されます。

グローバルスコープとサーバースコープで構成し、対応する構成ファイルにこれらを追加できます。

```
parallel_jobs = n
```

ここで、`n` は、ファイルコピー操作で使用する並列ワーカーの望ましい数です。デフォルト値は 1 です。

いずれの場合でも、ユーザーは、`backup` または `recover` コマンドを実行するときに、実行時にこの値をオーバーライドできます。例、次のように 4 つの並列ワーカーを使用できます。

```
barman backup --jobs 4 server1
```

または、代わりに：

```
barman backup --j 4 server1
```

この並列ジョブ機能は、`rsync` / `SSH` を介して構成されたサーバーでのみ使用できることにノートしてください。ストリーミングプロトコルで構成されたサーバーの場合、Barman は、現在一つのワーカーのみに制限されている `pg_basebackup` に依存します。

13.8 地理的冗長性

Barman で **カスケードバックアップアーキテクチャ** をセットアップすることができます。バックアップサーバーのソースは、PostgreSQL サーバーではなく Barman インストールです。

この機能により、ユーザーは地理的に分散した PostgreSQL バックアップのコピーを透過的に保持できます。

Barman の専門用語では、PostgreSQL サーバーではなく Barman インストールに接続されているバックアップサーバーが **パッシブノード** と定義されています。パッシブノードは、`primary_ssh_command` オプションを使用して構成され、グローバル（プライマリ Barman インストールの完全なレプリカ用）およびサーバーレベル（混在サーバー、直接サーバーとパッシブサーバーの両方）で使用できます。

13.8.1 同期情報

`barman sync-info` コマンドは、同期の目的に役立つ Barman サーバーの現在のステータスに関する情報を収集するために使用されます。使用可能な構文は次のとおりです。

```
barman sync-info [--primary] <server_name> [<last_wal> [<last_position>]]
```

このコマンドは、次を含む JSON オブジェクトを返します。

- そのサーバーのステータスが **DONE** であるすべてのバックアップを含むマップ
- アーカイブされたすべての WAL ファイルのリスト
- サーバーの構成
- 最後の読み取り位置（xlog データベースファイル）
- 最後に読み取られた WAL ファイルの名前

JSON 応答には、**master** ノードと **passive** ノード間の同期に必要なすべての情報が含まれています。

`--primary` が指定されている場合、コマンドはローカルではなく、定義されたプライマリノードで実行されます。

13.8.2 設定

サーバーを **passive node** として構成するのは簡単なオペレーションです。サーバー構成に次のオプションを追加するだけです。

```
primary_ssh_command = ssh barman@primary_barman
```

このオプションは、プライマリサーバーへの SSH 接続パラメーターを指定し、パッシブサーバーのバックアップデータのソースを識別します。

-c/--config オプションで **barman** を呼び出しており、パッシブノードがプライマリノードで **barman** を呼び出すときに同じオプションを使用する場合は、次のオプションを追加します。

```
forward_config_path = true
```

13.8.3 ノードの同期

ノードが **passive** としてマークされると、Barman によって特別な方法で処理されます。

- 標準のメンテナンス操作から除外されます
- **barman backup** を含む、PostgreSQL への直接操作は禁止されています

パッシブサーバーとそのプライマリサーバー間の同期は、**barman cron** によって自動的に管理され、透過的に呼び出されます。

1. 同期情報を収集するオーダーの **barman sync-info --primary** 2. プライマリノードで使用可能なすべてのバックアップのローカルコピーを作成するオーダーの **barman sync-backup** 3. プライマリノードプライマリノードで使用可能なすべての WAL ファイルをローカルでコピーオーダーの **barman sync-wals**

13.8.4 手動同期

barman cron はパッシブ/プライマリノードの同期を自動的に管理しますが、次の方法でバックアップの同期を手動でトリガーすることができます。

```
barman sync-backup <server_name> <backup_id>
```

sync-backup barman を起動すると、**primary_ssh_command** を使用してマスターサーバーに接続し、リモートマシンにバックアップが存在する場合、rsync を使用してすべてのファイルのコピーを開始します。バックアップごとに許可される同期プロセスは 1 つだけです。

WAL ファイルは、次の方法でも同期できます。

```
barman sync-wals <server_name>
```


第14章 Barman client utilities (barman-cli)

以前は別個のオープンソースプロジェクト `barman-cli` は、バージョン 2.8 から Barman のコアにマージされ、RPM / Debian パッケージとして配布されています。 `barman-cli` には、PostgreSQL サーバーと一緒にインストールする推奨クライアントユーティリティのセットが含まれています。

- `barman-wal-archive` : `archive_command` として使用されるアーカイビングスクリプト「`barman-wal-archive` による WAL アーカイビング」セクションで説明されているように。
- `barman-wal-restore` : のパートとして使用される WAL 復元スクリプトスタンバイおよびリカバリサーバーの `restore_command` リカバリオプション、上記の「`get-wal`」セクションで説明されているとおり。

詳細については、特定のマニュアルページまたは `--help` オプションを参照してください。

14.1 インストール

Barman クライアントユーティリティは通常、PostgreSQL がインストールされている場所にインストールされます。プライマリまたはスタンバイであるすべての PostgreSQL サーバーに `barman-cli` パッケージをインストールすることをお勧めします。

リポジトリをインストールするには、メインの「インストール」セクションを参照してください。

RedHat / CentOS システムにパッケージをインストールするには、`root` と入力します：

```
yum install barman-cli
```

Debian / Ubuntu では、`root` ユーザタイプとして：

```
apt-get install barman-cli
```


第15章 Barman client utilities for the Cloud (barman-cli-cloud)

Barman クライアントユーティリティは、オブジェクトストレージの統合をサポートし、サポートされているクラウドプロバイダーに WAL ファイルとバックアップを中継することにより、PostgreSQL データベースの災害リカバリ機能を強化するために拡張しました。

サポートされているクラウドプロバイダーは次のとおりです。

- AWS S3 (または任意の S3 互換オブジェクトストア)
- Azure Blob Storage
- Google Cloud Storage (Rest API)

これらのユーティリティは `barman-cli-cloud` RPM / Debian パッケージで配布されており、PostgreSQL サーバーと一緒にインストールできます。

`aws-s3` クラウドプロバイダーの資格情報をセットアップする方法については、[Credentials](#) を参照してください。

`azure-blob-storage` クラウドプロバイダーの資格情報については、[Environment variables for authorization parameters](#) を参照してください。次の環境変数がサポートされています: `AZURE_STORAGE_CONNECTION_STRING`、`AZURE_STORAGE_KEY`、および `AZURE_STORAGE_SAS_TOKEN`。 `--credential` オプションを使用して、`azure-cli` または `managed-identity` の資格情報を指定して、Azure Active Directory を介して認証オーダーこともできます。

警告: クラウドユーティリティには、使用するクラウドプロバイダーの適切なライブラリが必要です- (`<a href="https://github.com/boto/boto3">https://github.com/boto/boto3</a>`) [boto3] または (`<a href="https://docs.microsoft.com/en-us/python/api/azure-storage-blob/?view=azure-python">https://docs.microsoft.com/en-us/ python / api / azure-storage-blob / ? ビュー= azure-python</a>`) [azure-storage-blob] および (オプション) (`<a href="https://docs.microsoft.com/en-us/python/api/azure-identity/?view=azure-python">https://docs.microsoft.com/en-us/python/api/azure-identity / ? ビュー= azure-python</a>`) [azure-identity]

15.1 インストール

クラウド用の Barman クライアントユーティリティは、Barman をバイパスして、クラウドプロバイダーに直接バックアップする PostgreSQL サーバーにインストールする必要があります。

`barman-cloud-backup` および/または `barman-cloud-wal-archive` をフックスクリプトとして使用する場合は、`barman-cli-cloud` パッケージを Barman サーバーにもインストールできます。

リポジトリをインストールするには、メインの「インストール」セクションを参照してください。

RedHat / CentOS システムにパッケージをインストールするには、`root` と入力します：

```
yum install barman-cli-cloud
```

Debian / Ubuntu では、`root` ユーザタイプとして：

```
apt-get install barman-cli-cloud
```

15.2 barman-cloud フックスクリプト

上記の説明に従って、`barman-cli-cloud` パッケージを Barman サーバーにインストールします。

PostgreSQL サーバーの Barman 構成に次を追加して、`barman-cloud-backup` をバックアップ後スクリプトとして構成します。

```
post_backup_retry_script = barman-cloud-backup [*OPTIONS*] *DESTINATION_URL* ${BARMAN_
→SERVER}
```

警告： フックスクリプトとして実行する場合、`barman-cloud-backup` ではバックアップのステータスが `DONE` であることが必要であり、バックアップに他のステータスがある場合は失敗します。このため、バックアップのステータスが `WAITING_FOR_WALS` である間はフックスクリプトが実行されないように、`-w/--wait` オプションを使用してバックアップを実行することをお勧めします。

PostgreSQL サーバーの Barman 設定に次を追加して、`barman-cloud-wal-archive` を WAL アーカイブ前スクリプトとして設定します。

```
pre_archive_retry_script = barman-cloud-wal-archive [*OPTIONS*] *DESTINATION_URL* $
→{BARMAN_SERVER}
```

15.3 クラウドプロバイダーの選択

--cloud-provider オプションを使用して、バックアップと WAL のクラウドプロバイダーを選択します。これは、次のいずれかに設定できます。

- `aws-s3` [デフォルト] : AWS S3 または S3 互換オブジェクトストア。
- `azure-blob-storage` : Azure Blob Storage サービス。
- `google-cloud-storage` : Google Cloud Storage サービス。

15.4 プロバイダーごとの特異性

15.4.1 Google Cloud Storage

15.4.2 セットアップ

`google_storage_client` 依存関係が必要です : `bash pip3 install google-cloud-storage`

資格情報を設定するには :

- [Create a service account](#) そして、サービスアカウントキーを作成します。

*バケットアクセス権の設定 :

バケットのサービスアカウントに [Storage Admin Role](#) を与えることをお勧めします。

- `barman_cloud` を使用する場合、バケットが存在しない場合は作成されます。デフォルトのオプションを使用して作成しますバケツ。バケットに特定のオプション（リージョン、ストレージクラス、ラベル）が必要な場合は、作成することをお勧めしますそして、必要なものすべてにマッチようにバケットを設定しニーズ。

`postgres` (`archive_command` または `restore_command`) から `barman` クラウドを実行している場合、`postgres` 環境ファイルで `GOOGLE_APPLICATION_CREDENTIALS` を設定することを忘れないでください。

15.4.3 使用法

一部の詳細は、すべてのバーマンクラウドコマンドに固有です。 * `Google Cloud Storage` `--cloud-provider=google-cloud-storage` を選択 * `SOURCE_URL` は `gs` と `https` フォーマットの両方をサポートします。例 : `edb_notranlate_8` 一部の詳細は、すべてのバーマンクラウドコマンドに固有です。 * `Google Cloud Storage` `--cloud-provider=google-cloud-storage` を選択 * `SOURCE_URL` は `gs` と `https` フォーマットの両方をサポートします。例 : `edb_notranlate_8`

第16章 Troubleshooting

16.1 Barman インストールの診断

次を使用して、構成されたすべてのサーバーのステータスに関する重要な情報を収集できます。

```
barman diagnose
```

`diagnose` コマンドの出力は `barman` サーバーの完全な snapshot であり、グローバル構成、SSH バージョン、Python バージョン、`rsync` バージョン、PostgreSQL クライアントバージョン、およびすべてのサーバーの現在の構成とステータスなどの有用な情報を提供します。

`diagnose` コマンドは、Barman インストールのステータスに関するグローバルビューを提供するため、問題のトラブルシューティングに非常に役立ちます。

16.2 ヘルプのリクエスト

Barman は広範囲にわたって文書化されていますが、カバーされていない多くのシナリオがあります。

Barman および Barman を使用した災害リカバリシナリオに関する質問については、コミュニティメーリングリストを使用して開発チームに連絡できます。

<https://groups.google.com/group/pgbarman>

または freenode の IRC チャンネル：`irc://irc.freenode.net/barman`

バグを発見したイベントは、Github を使用してチケットを開くことができます：[https://github.com/ EnterpriseDB/ barman / issues](https://github.com/EnterpriseDB/barman/issues)

EnterpriseDB は、24 時間年中無休のサービスを含む Barman の専門サポートを提供します。

16.2.1 バグを提出する

Barman は広範囲にテストされており、現在いくつかの稼動環境で使用されています。ただし、他のソフトウェアと同様、Barman にはバグがありませフリー。

バグを発見した場合は、次のプロシージャに従ってください。

- `barman diagnose` コマンドを実行します
- を添付して、Github 課題トラッカーを通じてバグをファイルします上記の診断コマンドによって取得された出力（`barman` 診断）

警告: 診断コマンドの出力を送信するときは、セキュリティのビューから潜在的に危険な情報を開示する可能性があるため注意してください。

第17章 The Barman project

17.1 サポートとスポンサーの機会

Barman は、EnterpriseDB によって作成および保守されているフリーソフトウェアです。Barman の使用に関するサポートが必要な場合、または新しい機能が必要な場合は、EnterpriseDB に連絡してください。オープンソースとして一般公開される Barman と PostgreSQL の新機能の開発を後援できます。

詳細については、次をご覧ください。

- [Barman website](#)
- [Support section](#)
- [EnterpriseDB website](#)
- [Barman FAQs](#)
- [2ndQuadrant blog: Barman](#)

17.2 Barman への貢献

EnterpriseDB には、ソフトウェアエンジニア、アーキテクト、データベース管理者、システム管理者、QA エンジニア、開発者、および管理者のチームがあり、バーマンのコードを改善するために時間と専門知識を捧げています。私たちは、ソフトウェア開発に無駄のないアジャイルな方法論を採用し、部門を超えたコラボレーションを通じて厳密なテスト手順を実装できる devops 文化を信じています。Barman のコミットはすべて、稼動パイプラインのさまざまな段階でのマルチプル個人の個人の貢献です。

これは Barman を開発するための好ましい方法ですが、外部開発者からのパッチを喜んで受け入れます：

- ユーザ文書（チュートリアルとマニュアルページ）が提供されています。
- ソースコードが適切に文書化され、関連するコメントが含まれています。
- 提供されたコードは単体テストでカバーされています。
- 無関係な機能が侵害または破損されていない。
- ソースコードは、現在のマスタブランチに基づいています。
- コミットおよびプルリクエストは、単一の機能（マルチ機能パッチのテストとレビューはハード）。
- ユーザインタフェースの変更については、EnterpriseDB で事前に説明します。

また、開発者の雇用主からの著作権の譲渡と、雇用のための所有権の主張の免責事項を提供することも求めています。

この目的には、Github のプルリクエストシステムを使用できます。

17.3 著者

アルファベット順に：

- Abhijit Menon-Sen ジェーンスリーフットマイケルウォレス

過去の貢献者（アルファベット順）：

- Anna Bellandi (QA /テスト)
- Britt Cole (文書レビューア)
- Carlo Ascani (開発者)
- Francesco Canovai (QA /テスト) *ガブリエーレバルトリニ (建築家)
- Gianni Ciolli (QA /テスト)
- Giulio Calacoci (開発者)
- Giuseppe Broccolo (開発者)
- Jonathan Battiato (QA /テスト)
- Leonardo Cecchi (開発者) マルコネンシアリーニ (プロジェクトリーダー) NiccolòFei (QA /テスト)
*ルーベンス・スーザ (QA /テスト)
- Stefano Bianucci (開発者)

17.4 リンク

- [check-barman](#) : ホルガーによって書かれた Barman 用の Nagios プラグイン
ハマン (MIT ライセンス)
- [puppet-barman](#) : Puppet 用の Barman モジュール (GPL)
- [Tutorial on "How To Back Up, Restore, and Migrate PostgreSQL Databases with Barman on CentOS 7"](#)、
Sadequl
Hussain 作 (DigitalOcean コミュニティで入手可能)
- [BarmanAPI](#) : Barman 向け RESTFul API、Mehmet
EminKarakaş (GPL) が作成

17.5 ライセンスと貢献

Barman は EnterpriseDB UK Limited のプロパティであり、そのコードは GNU General Public ライセンス 3 の下で配布されています。

© Copyright EnterpriseDB UK Limited 2011-2022

Barman は、欧州委員会の第 7 フレームワークプログラムによって資金提供された研究プロジェクトである [4CaaSt](#) を通じて部分的に資金提供されています。

Barman への貢献は歓迎され、AUTHORS ファイルにリストされます。EnterpriseDB UK Limited は、いかなる貢献でも、著作権の譲渡と、開発者の雇用者からの雇用のための所有権の主張の免責事項を提供することを要求しています。これにより、すべての Barman ディストリビューションがフリーコードのままであることを確認できます。関連する Copyright 譲渡フォームのコピーについては、[barman @ enterprisedb.com](mailto:barman@enterprisedb.com) までお問い合わせください。

付録

第18章 Feature matrix

以下に、バックアップとアーカイビング用の PostgreSQL バージョンと Barman 機能のマトリクスを示します。

Version	Backup with rsync/SSH	Backup with pg_basebackup	Standard WAL archiving	WAL Streaming	RPO=0
14	Yes	Yes	Yes	Yes	Yes
13	Yes	Yes	Yes	Yes	Yes
12	Yes	Yes	Yes	Yes	Yes
11	Yes	Yes	Yes	Yes	Yes
10	Yes	Yes	Yes	Yes	Yes
9.6	Yes	Yes	Yes	Yes	Yes
9.5	Yes	Yes	Yes	Yes	Yes ~(d)~
9.4	Yes	Yes	Yes	Yes	Yes ~(d)~
9.3	Yes	Yes ~(c)~	Yes	Yes ~(b)~	No
9.2	Yes	Yes ~(a)~ ~(c)~	Yes	Yes ~(a)~ ~(b)~	No
9.1	Yes	No	Yes	No	No
9.0	Yes	No	Yes	No	No
8.4	Yes	No	Yes	No	No
8.3	Yes	No	Yes	No	No

注：

a) pg_basebackup および pg_receivexlog 9.2 が必要 b) WAL ストリーミングのみはサポートされていない（標準アーカイビングが必要） c) サポートされていない表領域のバックアップ d) pg_receivexlog 9.5 を使用する場合、マイナーバージョン 9.5.5 以上が必要 [^commitsync]

^commitsync が必要です（バージョン 9.5.5 に含まれています）

Barman は、同じバージョン（またはそれ以降）の PostgreSQL サーバーの pg_basebackup および pg_receivewal / pg_receivexlog が Barman が存在する同じサーバーにインストールされる必要があります。唯一の例外は、PostgreSQL 9.2 ユーザーが pg_basebackup および pg_receivexlog のバージョン 9.2 を Barman と一緒にインストールする必要があることです。

ヒント： PostgreSQL の場合、pg_basebackup および pg_receivewal を介したストリーミングレプリケーションでバックアップおよび WAL アーカイビングを使用する場合、PostgreSQL クライアントの最後のメジャーで安定（stable）バージョン（11 など）を Barman サーバーにインストールすることをお勧めし PostgreSQL 9.3 以降のサーバー。

TIP: “RPO = 0” アーキテクチャの場合、少なくとも 1 つの同期的スタンバイサーバを使用することをお勧めします。