



EDB

Barman for PostgreSQL

リリース 2.12

EnterpriseDB

2021 年 06 月 28 日

目次

第 1 章	バーマン 1、多数の PostgreSQL サーバー	7
第 2 章	ストリーミングバックアップと rsync / SSH	9
第 3 章	標準アーカイビング、WAL ストリーミング…、またはその両方	11
第 4 章	バックアップの 2 つの典型的なシナリオ	13
4.1	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	13
4.2	シナリオ 2：rsync / SSH を介したバックアップ	14
第 5 章	バックアップの要件	17
第 6 章	リカバリの要件	19
第 7 章	RPM パッケージソフトを使用した RedHat / CentOS へのインストール	21
第 8 章	パッケージソフトを使用した Debian / Ubuntu へのインストール	23
第 9 章	ソースからのインストール	25
第 10 章	Barman 2.10 からのアップグレード	27
第 11 章	Barman 2.X からのアップグレード (2.8 より前)	29
第 12 章	Barman 1.X からのアップグレード	31
第 13 章	オプションの範囲	33
第 14 章	設定の例	35
第 15 章	準備手順	37
15.1	PostgreSQL 接続	37
15.2	PostgreSQL WAL のアーカイビングとレプリケーション	38
15.3	PostgreSQL ストリーミング接続	39
15.4	SSH 接続	40

15.4.1 postgres ユーザの SSH 設定	40
15.4.2 barman ユーザの SSH 設定	41
15.4.3 PostgreSQL から Barman へ	41
15.4.4 バーマンから PostgreSQL へ	41
第 16 章 サーバー構成ファイル	43
第 17 章 WAL ストリーミング	45
17.1 複製スロット	46
17.2 WAL ストリーミングを設定する方法	46
17.3 リカバリを伴う部分的な WAL ファイルの制限	47
第 18 章 archive_command を介した WAL アーカイビング	49
18.1 barman-wal-archive を介した WAL アーカイビング	49
18.2 rsync / SSH を介した WAL アーカイビング	50
第 19 章 WAL アーカイビング設定の検証	53
第 20 章 ストリーミングバックアップ	55
第 21 章 rsync / SSH を使用したバックアップ	57
第 22 章 Windows ベースのサーバーをセットアップする方法	59
第 23 章 cron	61
第 24 章 diagnose	63
第 25 章 list-server	65
第 26 章 archive-wal	67
第 27 章 backup	69
第 28 章 check	71
第 29 章 get-wal	73
第 30 章 list-backup	75
第 31 章 rebuild-xlogdb	77
第 32 章 receive-wal	79
32.1 receive-wal プロセス管理	79
32.2 複製スロット管理	80
第 33 章 replication-status	81

第 34 章	<code>show-server</code>	83
第 35 章	<code>status</code>	85
第 36 章	<code>switch-wal</code>	87
第 37 章	バックアップ ID のショートカット	89
第 38 章	<code>check-backup</code>	91
第 39 章	<code>delete</code>	93
第 40 章	<code>list-files</code>	95
第 41 章	<code>recover</code>	97
41.1	リモートリカバリ	98
41.2	表領域の再マッピング	98
41.3	ポイントインタイムリカバリ	99
第 42 章	<code>show-backup</code>	101
第 43 章	バックアップ機能	103
43.1	増分バックアップ	103
43.2	帯域幅の使用を制限する	104
43.3	ネットワーク圧縮	105
43.4	並行バックアップとスタンバイからのバックアップ	105
43.4.1	スタンバイからのバックアップに関する現在の制限	106
43.5	即時チェックポイント	107
43.6	ローカルバックアップ	107
第 44 章	アーカイブ機能### WAL 圧縮	109
44.1	同期 WAL ストリーミング	109
第 45 章	カタログ管理機能### 最小の冗長性の安全性	111
45.1	保持ポリシー	111
45.1.1	スコープ	112
45.1.2	仕組み	112
45.1.3	設定と構文	113
第 46 章	フックスクリプト	115
46.1	バックアップスクリプト	116
46.2	バックアップ削除スクリプト	117
46.3	WAL アーカイブスクリプト	117
46.4	WAL 削除スクリプト	118
46.5	回復スクリプト	119

第 47 章 カスタマイズ	121
47.1 ファイルディレクトリをロックする	121
47.2 バイナリ経路	121
第 48 章 クラスター管理システムとの統合	123
第 49 章 並列ジョブ	125
第 50 章 地理的冗長性	127
50.1 同期情報	127
50.2 設定	128
50.3 ノードの同期	128
50.4 手動同期	128
第 51 章 インストール	131
第 52 章 インストール	133
第 53 章 Barman インストールの診断	135
第 54 章 ヘルプのリクエスト	137
54.1 バグを提出する	137
第 55 章 サポートとスポンサーの機会	139
第 56 章 バーマンへの貢献	141
第 57 章 著者	143
第 58 章 リンク	145
第 59 章 ライセンスと貢献	147

% Barman Manual % 2ndQuadrant Limited % 2020 年 11 月 5 日 (2.12)

**** Barman **** (バックアップ リカバリー Manager) は、Python で書かれた 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] [11] は GNU GPL 3 の下で配布され、[PostgreSQL プロジェクト] [31] のプラチナスポンサーである [2ndQuadrant] [13] によって管理されています。

重要：改行>このマニュアルでは、理論的な災害リカバリの概念に精通し、物理的バックアップと災害リカバリの観点から 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、セクション「スタートする前に」を参照してください。

newpage

はじめに

完全な世界では、バックアップは必要ありません。ただし、特にビジネス環境では、「予期しない」が発生した場合に備えておくことが重要です。データベースシナリオでは、予期しないものは次のいずれかの形式をとることができます。

- 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、可能な限り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、もっと広い意味で事業継続性と呼びます。

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) ***] [rpo] : 「メジャーインシデントが原因で IT サービスからデータが失われる可能性がある最大対象期間」
- [**目標復旧時間 (RTO)**] [rto] : 「関連する容認できない結果を回避オーダーに、災害 (または混乱) の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事業継続性が途切れる」

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-それが望ましいことではありますが。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、* 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 *] [repmgr] などの* High Availability *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。

2ndQuadrant での経験に基づいて、Barman と 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。

バーマンの私たちの使命は、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 とバーマンの連続モニタリングに依存し、それは速やかに任意の異常を識別することができます

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 PostgreSQL で最も使用されているバックアップツールの 1 つである Barman の開発を開始しました。バーマンは、「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。

newpage

あなたがスタートする前に

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を理解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完全なリストを示します。

- PostgreSQL ドキュメント :
- [SQL ダンプ] [sqldump] [^ pgdump]
- [ファイルシステムレベルのバックアップ] [physicalbackup]
- [連続アーカイブとポイントインタイムリカバリ (PITR)] [pitr]
- [信頼性と先行書き込みログ] [wal]
- Book : [PostgreSQL 10 Administration Cookbook] [adminbook]
 - [^ pgdump] : 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 のようなツールの違いを知っておくことが重要です。

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。一年中いつでも、2ndQuadrant などの PostgreSQL 会社が提供する多くのコースが世界中で利用できます。

newpage

設計とアーキテクチャ

バーマンの基盤の 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、あなたはそう両方のことを、あなたはバーマンサーバが離れすぎて自分の PostgreSQL からにしたいくない、あなたのバーテンダーサーバは何千マイルも離れてあなたの PostgreSQL server.Realistically から、世界の別のパートにあるデータセンターに位置して持つことができますバックアップとリカバリーの時間は制御されます。

Barman をセットアップする_“万能サイズ”_の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：

- バーマン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と統合する [[^] nagios]
- 稼動環境に展開する前にすべてをテストする
 - [[^] nagios] : Nagios / Icinga との統合は、Barman 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。

災害リカバリー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 PostgreSQL。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
- あなたの意思決定を make と、最初のソリューションを実装

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_を使用すると、異なるデータセンター/アベイラビリティゾーンにある 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せん。さらにあります：グローバルレベルだけでなくサーバーレベルでも地理的冗長性を Barman で構成できる場合、一部のサーバーがローカル 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_hybrid installations_を作成できます以下の図 ref {georedundancy-design}は、それぞれがローカルの Barman インストールでバックアップされたプライマリ PostgreSQL サーバーを持つ 2 つのアベイラビリティゾーン（ヨーロッパとアメリカ）を示しています。、および rsync / SSH を介した多層バックアップのために、他の Barman サーバー（_passive_として定義）で中継されます。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。

代わりに_hook scripts_のおかげで、Barman のバックアップは、tar を介した_tape_などの異なるメディア、または Amazon クラウドの_S3 バケットのような場所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。

バーマン 1、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。バーマンは、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] : 同じ [PostgreSQL の PITR の要件] [requirements_recovery] がリカバリに適用されます。詳細については、セクション「リカバリの要件」を参照してください。

その結果、PostgreSQL サーバーが中央のバーマン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 backup)

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。[^fmatrix]

- [^fmatrix] : 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している「機能マトリクス」をチェックインします。

これらの 2 つの方法のいずれかを選択すると、あなたが make 必要があります決定です。

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 : `pg_basebackup` を利用しているので、現在利用可能でない増分バックアップ、並列バックアップ、重複排除、およびネットワーク圧縮などの機能<。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、バックアップをストリーミングと、バックアップに Windows の [^ウィンドウ] 上の PostgreSQL をことができますし、ドッカーで作業する際の人生が容易になります。

- [^windows] : Windows での 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、セクション「Windows ベースのサーバーのセットアップ方法」を参照してください。

標準アーカイビング、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 を介して、現在は `barman-cli` パッケージの `barman-wal-archive` を介して) を介して標準の 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。シンプルに保つために、これは通常 16MB のデータ変更ごとに発生します。

Barman 1.6.0 では、`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「ゼロに近い」値に下げます。

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) クラスタで 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、従来のアーカイビングの使用を余儀なくされます。他では、あなたがそれを行うにはない強い理由があるの両方を使用するか、単に WAL ストリーミング Unless するかどうかを選択することができ、我々は最大限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。

バックアップの 2 つの典型的なシナリオ

生活を楽しみ make オーダーに、バーマンの特定の PostgreSQL サーバーの最も一般的な 2 つの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した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忘れないでください。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。

4.1 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 以降を使用していて、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。

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、モニタリングのための 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。

この代替アプローチには以下が必要です。

- PostgreSQL 上の `postgres` ユーザはバーマンサーバ上 `barman` ユーザとして接続することができ、追加の SSH 接続
- PostgreSQL で `archive_command` はバーマンに WAL ファイルを出荷するように構成すること

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。

4.2 シナリオ 2 : `rsync` / SSH を介したバックアップ

SSH を介した `rsync` の `_traditional_` 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

Backup architecture for PostgreSQL

Scenario 2 - Traditional Barman setup

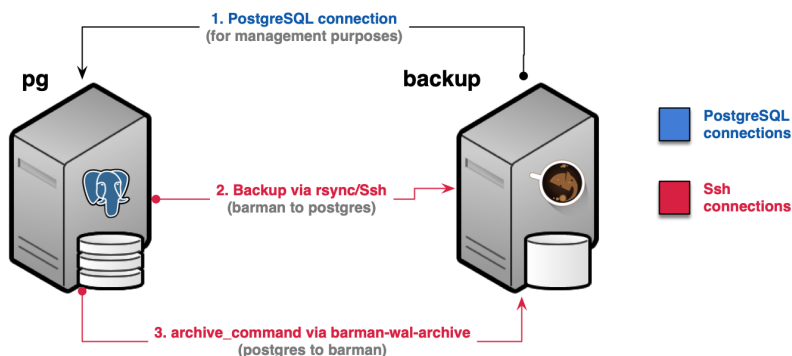


図 1 Scenario 2 - Backup via rsync/SSH

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. バーマンサーバ上 `barman` ユーザは PostgreSQL のサーバ 3 に `postgres` ユーザとして接続することができ `rsync` によって使用されるべきベースバックアップ操作の SSH 接続。WAL の SSH 接続

の PostgreSQL で `archive_command` で使用されるアーカイビングかつ、PostgreSQL 上の `postgres` ユーザはバーマンサーバ上 `barman` ユーザとして接続することができ

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？ ->

newpage

システム要件

- Linux / Unix Python >= 3.4 Python モジュール :
- `argcomplete`
- `argh` >= 0.21.2
- `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) リポジトリ] [epel] をインストールする必要があります。

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

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。

リカバリの要件

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

リモートリカバリは、間違いなく Barman で PostgreSQLserver を復元する最も一般的な方法です。

どちらの場合も、同じ [PostgreSQL のログ SHIPPING とポイントインタイムリカバリの要件が適用されます] [requirements_recovery] :

- 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、我々はあなたが同じオペレーティングシステム、同じ PostgreSQL のバージョン、同じディスクレイアウトなどを使用することを示唆しています。

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。回復操作を実践し、それらに精通することにより、予期しない何か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、指定された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。

newpage

インストール

重要 : >バーマンをインストールすることをお勧めの方法は、あなたの 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。

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。事前に追加パッケージ EnterpriseLinux (EPEL) リポジトリと [PostgreSQL グローバル開発グループ RPM リポジトリ] [yumpgdg] を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、2ndQuadrantvia Yum が [2ndQuadrant Public RPM リポジトリ] [2ndqrpmrepo] を通じて、そのウェブサイトにある指示に従って配布します。

次に、root として次のように入力します。

```
yum install barman
```

歴史的な理由により、2ndQuadrant は [Sourceforge.net] [3] 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ています。

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。

それはあなたがバーマンの最新バージョンを持っていると思います。IF、しかし、これらのリポジトリが利用可能な最新バージョンが含まれていない可能性があり、推奨されるメソッドは、これらのリポジトリの両方をインストールすることでの Debian と Ubuntu の公式リポジトリに直接提供されています：

- [2ndQuadrant Public APT リポジトリ] [2ndqdebrepo]、によって直接維持バーマン開発者
- [PostgreSQL コミュニティ APT リポジトリ] [aptpgdg]、[PostgreSQL Wiki の APT セクション] [aptpgdgwiki] の指示に従う

注：> 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ているため、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
```


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、依存関係管理などのシステム管理アクティビティ、> 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、ログ管理などが必要です。

backup サーバーに 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 アプリケーションがユーザディレクトリにインストールされます ([PATH 環境変数が適切に設定されていることを確認してください] [setup_user])。

[Barman は Python パッケージインデックス (PyPI)] [pypi] でも利用可能で、pip からインストールできます。

バーマンのアップグレード

バーマンはトランクベースの開発パラダイムに従います。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で数千の自動テストを実行します。

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 をアップグレード処理するには、通常 yum update または apt update を使用したパッケージ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、構成を少し変更する必要がありました。

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、通常どおり最新アップグレードできます。

BARMAN 2.X からのアップグレード (2.8 より前)

バーマン 2.7 またはプライマリーサーバ上 rsync バックアップメソッドの olderusers のバージョンからアップグレード処理する前にどちらか `concurrent_backup` (forPostgreSQL 9.6 以上を推奨) または `exclusive_backup` (現在のデフォルト) に `backup_options` を `explicitlyset` する必要があり、そうでない場合はバーマンは、それが実行されるたびにワーニングを發します。

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がデフォルトで有効になっていました。

あなたが最新バージョンにバーマンのインストールをアップグレードする前に、あなたは、グローバルに以下の行を追加して make または任意の serverthat のためにそれを必要とします。

```
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すべてのサーバーで透過的に setarchiver = on を設定し、ワーニングます。

newpage

設定

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 に設定）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：歴史的な理由から、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単一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により、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。

引用符で囲む必要はありません。

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

オプションの範囲

すべての構成オプションには `_scope_` 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、`_general セクション_`で許可されます。これは、INI ファイルで `[barman] label` によって識別されます。

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

サーバーオプションは、`_server section_`でのみ指定できます。`_serversection_`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
```

バーマン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、[「man」ページのセクション 5] [man5] をページ。

```
man 5 barman
```

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
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、`distributbarman.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。

newpage

バーマンでの新しいサーバーのセットアップ

「設計とアーキテクチャ」_セクションで述べたように、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の ID とホスト>として 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、バックアップとリカバリの履歴全体で同じ 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、インスタンスのアップグレードを実行する場合（例 ``pg\_upgrade``を使用する場合、Barman で同じサーバー定義を再利用してはならず、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。

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いくつかの準備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 のアーカイビングおよびバックアップ戦略の観点から決定することが重要です。特に、あなたは> WAL を使用する方法だけでなく、バックアップ>メソッドをアーカイビングかを決める必要があります。

15.1 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 --replication -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 `createuser` コマンドを使用すると、その後、backup サーバー上`> ~barman/.pgpass` ファイルに追加することをお勧めしますパスワード、>のためにプロンプトされます。詳細については、[PostgreSQL ドキュメントの [“パスワードファイル” セクション] [pgpass] を参照してください。

この接続は、その活動をサーバーと調整オーダー、およびモニタリング目的で Barman に必要です。

PostgreSQL が提供するものの中から、お気に入り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、[PostgreSQL ドキュメントの [“クライアント認証” セクション] [pg_hba] 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、およびデータベース名前）を書き留めて、後で使用でき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：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。

15.2 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 を適切に設定して、Barmanserver 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 ドキュメントの以下のセクションをお読みください：

- [ロール属性] [ロール]
- [pg_hba.conf ファイル] [authpg_hba]
- [ストリーミングレプリケーションを使用したスタンバイサーバのセットアップ] [stream-prot]

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すべて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。

設定を更新するには、PostgreSQL サーバーを再起動します。

15.3 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予定の場合は、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 名前付けの特定のユーザを PostgreSQL で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、あなたが、その後 backup サーバー上> ~barman/.pgpass ファイルに追加することをお勧めされている>、パスワードをプロンプトされます。詳細については、[PostgreSQL ドキュメントの [“パスワードファイル” セクション] [pgpass] 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：>あなたはそれ以上に行く前に>ストリーミングレプリケーションを経由して接続することができます make 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 アーキテクチャに依存します。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の最大数 [^ replslot94] 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 ファイルを受信する場合に必要です。

```
max_replication_slots = 2
```

- [^ replslot94] : 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。

`max_replication_slots` および `max_wal_senders` に提案された値は例として考慮される必要があり、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。

15.4 SSH 接続

SSH は、リモートサーバを開き、サーバーとローカルシステム間でファイルをコピーできるようにするプロトコルとツールのセットです。Digital Ocean の記事 [“SSH Essentials”] [ssh_essentials] に、SSH の使用文書があります。

SSH キー交換は非常に一般的な方法であり、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ます。また、WAL アーカイビングとバックアップに `rsync` を使用する必要があります。

注意 : >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不要です。

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

15.4.1 postgres ユーザの SSH 設定

以前に行っていない限り、PostgreSQL ユーザ用の 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。

15.4.2 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。

15.4.3 PostgreSQL から Barman へ

archive_commandSetting 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ーへの SSH 接続が必要です。

成功し BackupServer に PostgreSQL から接続するには、PostgreSQL の公開キーは barman ユーザのためのバックアップサーバの theauthorized キーに設定する必要があります。

許可される公開キーは、.ssh/id_rsa.pub 名前付けのファイルの postgres userhome ディレクトリ内に保存され、その内容は、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 名前付けのファイルに含まれる必要があります。authorized_keys ファイルが存在しない場合は、600 aspermissions を使用して作成します。

SSH キーペア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、「archive_command セクションを介した WAL アーカイビング」で説明します。

15.4.4 バーマンから 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ーへの接続と同様に、postgres ユーザ PostgreSQL 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。

キーペア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
```

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_「Preliminary steps : PostgreSQL connection」_に応じて設定されます。

backup_method オプションの意味は、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する場合、サーバー構成ファイルに streaming_conninfo パラメーターも必要です。

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

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

WAL ストリーミング

Barman は、標準の `archive_command` 戦略に加えて、ユーザーが PostgreSQL サーバーから連続的な 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。

バーテンは依存している `[pg_receivewal]` [25]、ネイティブ `streamingreplication` プロトコルを利用する PostgreSQL の 9.2 から `beenavailable` 有し、連続 `pg_receivewal` が `pg_receivexlog` 名前付けした aPostgreSQL サーバー（マスタまたはスタンバイ）の PostgreSQL 10 .Prior からトランザクションログを受信するユーティリティ。

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 `pg_receivexlog` of バージョン 9.2 をインストールする必要があります。PostgreSQL 9.3 以降では、互換性があるため、`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。それ以外の場合、ユーザーは `pg_receivewal` / `pg_receivexlog` 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ポイントできます。

トランザクションログのストリーミングを有効にするオーダーには、以下を行う必要があります。

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ログ。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： > `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、また 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ステータスを監視できるようにします。

17.1 複製スロット

重要： PostgreSQL 9.4 以降のレプリケーションスロットが利用可能

レプリケーションスロットは、PostgreSQL サーバーが `allarchivers` によって受信されるまで 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。

レプリケーションスロットの詳細については、[PostgreSQL マニュアル] [replication-slots] をご覧ください。

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作成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 上で実行されている 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 サポートはまだ実験的であり、継続的な統合システムのパートではありません。

17.2 WAL ストリーミングを設定する方法

最初に、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 PostgreSQL サーバーを構成する必要があります。

Barman から PostgreSQLserver へのストリーミング接続を構成するには、サーバー構成ファイルの次の行を含めて、すでに述べたように `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必要なもう 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>を設定する場合は、手動でこのコマンドを使用してバーマンため areplication スロットを作成することができます。

```
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、以下を設定することにより、Barman がレプリケーションスロットを自動的に作成するように設定できます。

```
create_slot = auto
```

17.3 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 .partial サフィックスを持つファイルにトランザクション情報を書き込むことです。

バーマンは、サーバーの streaming_wals_directory にある部分ファイルを想定しています。完了すると、pg_receivewal は .partial 接尾辞を削除し、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という非常に重要な情報が含まれています。

Barman 2.10 の時点で、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。したがって、get-wal を有効にして--standby-mode を使用せずに recover コマンドを実行する場合、Barman は -P オプションを restore_command リカバリオプションで自動的に barman-wal-restore に追加します (その後、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。

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 の受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なければなりません。

これは 2 つの方法で実行できます。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、WAL ファイルのサポートを導入しました>スタンバイから `archive_command` を使用したアーカイビング。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。

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

Barman 2.6 以降、`archive_command` 経由で WAL ファイルを安全かつ確実に Barman にアーカイブするための**推奨方法**は、2ndQuadrant パブリックリポジトリ経由で配布され、GNU GPL 3 ライセンスの下で利用可能な `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。

代わりに `rsync` の `barman-wal-archive` を使用して / SSH は `nomechanism` があり、`archive_command` WAL ファイルとして `rsync` を / SSH を使用してバーマン server。When に出荷された WAL ファイルの `riskof` データの破損を低減し、そのファイルのコンテンツがに `flushedand` に `fsync-ED` であることを保証します宛先のディスク。

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、そのサーバーの適切な `incoming` ディレクトリにコンテンツを `fsync` 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 Barman で受信 WAL ファイルを受け入れるように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 で設定された PostgreSQL サーバーの名前、`DUMMY` はプレースホルダ (`barman-wal-archive` には WAL ファイル名前の引数が必要ですが、これは無視されます)。

`pg` ファイル上の 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て、`archive_command` を `USEbarman-wal-archive` に設定します。

```
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。

18.2 rsync / SSH を介した WAL アーカイビング

`show-server` 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-server pg |grep incoming_wals_directory
*      incoming_wals_directory: /var/lib/barman/pg/incoming
```

`pg` ファイル上の 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-server 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_commandprocess` にさらに厳密なチェックを追加することもできます。例、一部のユーザは、次のいずれかを示唆しています：

```
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 `_trick_` は、サーバーが複製され、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。

WAL アーカイビング設定の検証

継続的アーカイビングが上、適切に動作していることをテストするオーダーには、PostgreSQL とバックアップサーバの両方をチェック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 ファイルのスイッチを強制し、Barman でアーカイビングプロセスをトリガーします。バーマンは `onfile` 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。

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 とともにインストールされるバージョン 9.2 の `pg_basebackup` が必要です。PostgreSQL 9.3 以降では、互換性があるため、`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。あなたは、Barman>サーバーに `pg_basebackup` のマルチプル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ポイントすることもできます。構成ファイルで `path_prefix` オプションを使用します。

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 アーカイビングが現在設定されていない場合、バックアップをスタートできないことに注意してください。

サーバー構成が有効かどうかを確認するには、`barmancheck` 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
```

重要：`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 `pg_basebackup` 9.4 以上が必要です。

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 より前の唯一の使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」_セクションを参照してください。

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ために ssh_command オプションが必要です。

重要：WAL アーカイビングが現在構成されていない場合、バックアップをスタートできないことに注意してください。

サーバー構成が有効かどうかを確認するには、barmancheck 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
```


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：この機能はまだ継続的な統合システムのパートではないため、まだ実験的です。

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。

通常どおりサーバーをバックアップします。

Windows サーバーではリモートリカバリがサポートされていないため、クラスターをローカルで Barman サーバーで復元し、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。

newpage

一般的なコマンド

バーマンには多くのコマンドがあり、説明のために、スコープ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。

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 などには何也没有什么ありません)。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 barmancron 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:>このコマンドは、_cron script_で実行する必要があります。推奨事項は、barman cron を毎分実行するようにスケジュールすることです。> 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 エントリが作成されます。

barman cron は、サーバーごとに WAL アーカイビング操作を同時に実行します。これにより、以下の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。

cron コマンドによって開始された動作を停止させるオーダーに、outthe の cron エントリをコメントアウトして実行します。

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

barman list-server をチェックして、すべてのサーバーを取得する make にしてください。

第

TWENTYFOUR

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、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、異なるバージョン間で変更できます。

TWENTYFIVE

LIST-SERVER

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-server
```

--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-server --minimal
```

newpage

サーバーコマンド

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 のそ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、バックアップの作成、WAL アーカイブの管理に役立ちます。

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、直接呼び出せる場合でも、cron 一般コマンドから開始するように設計されています。

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構成パラメータを上書きできるいくつかのオプション。詳細については、マニュアルページ。

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>
```

ヒント: > 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。

TWENTYEIGHT

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>
```

ヒント：> 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。アラートとモニタリングインフラストラクチャに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。

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] [-j|-x] <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` の値を `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。arecovery 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 ファイルを取得するように `restore_command` を設定することにより、リカバリ構成を準備します。

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 asbarman ユーザで呼び出す必要があり、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 がそれらのいずれか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ること、および必要な PostgreSQL サーバーが WAL ファイルを送信するように Barman で構成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 (`barman-wal-restore` には WAL ファイル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)。コマンド、PostgreSQL サーバーで `man barman-wal-restore` と入力します。

LIST-BACKUP

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-backup <server_name>
```

ヒント：all をサーバー名前として使用して、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。

REBUILD-XLOGDB

いつでも、特定のサーバー（または `all` ショートカットを使用してすべてのサーバー）の WAL アーカイブのコンテンツを再生成できます。WALarchive 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>
```


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。

32.1 `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：> `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に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` 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が高い場合、`receive-wal` プロセスが遅れて同期し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、誤った仮定を行わないために、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することを要求しています。

32.2 複製スロット管理

`receive-wal` プロセスは、Barman が WAL アーカイビングプロシージャに必要とする複製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。

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>
```

ヒント：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。

第

THIRTYFOUR

SHOW-SERVER

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-server <server_name>
```

**** TIP :** ******サーバー名前 all as>を使用して、完全な構成レポートを要求できます。

第

THIRTYFIVE

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>
```


SWITCH-WAL

このコマンドは、現在のログファイルは、閉じ受信し thenarchived ことができるように、別の TRANSACTIONLOG ファイル (WAL) への PostgreSQL スイッチになります。

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

最後の transactionlog 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かった場合、--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 : 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。

newpage

バックアップコマンド

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。

注意 : > barman list-backup <server_name> でバックアップ ID を取得できることを忘れないでください

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

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、次のコマンドを発行できます。

```
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。

THIRTYEIGHT

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 オペレーションの最後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 asWAITING_FOR_WALS という 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。そのため、バックアップ操作を実行する前に、特にスタンバイサーバからバックアップするときに、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。

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 はバックアップ toWAITING_FOR_WALS の状態を一時的に保存し、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処理します。

第

THIRTYNINE

DELETE

以下を使用して、指定されたバックアップを削除できます。

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

delete コマンドは、shortcut を受け入れてバックアップを識別します。

LIST-FILES

あなたが所与のバックアップ用のファイル（ベースバックアップおよび必要な WAL ファイル）をリストすることができます：

```
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、アーカイビング手順に Barman を統合するのに非常に役立ちます。

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停止することを忘れないでください。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はローカルにリカバリされ、宛先パスには PostgreSQL インスタンスをスタートするために使用できるデータディレクトリが含まれます。

重要：>このコマンドをユーザ barman として実行すると、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、list-backup コマンドを使用して取得できます。

重要：>現在、Barman は PGDATA>内のシンボリックリンクを追跡していません (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、それらを復元する必要がある場合に備えて災害リカバリ計画/手順に追加することをお勧めします>オリジナルの場所に。

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。

41.1 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。これにより、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：>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停止することを忘れないでください。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、システムの一部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 ``get-wal <#get-wal>`` __コマンドが指定されていない限りバーマン構成の `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。

41.2 表領域の再マッピング

Barman は `-tablespace` 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テーブルスペース。このオプションは、`NAME:DIRECTORY` 形式を使用して、引数として値のペアを受け入れフォーマット。

- `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）。

41.3 ポイントインタイムリカバリ

Barman は PostgreSQL の Point-in-Time Recovery (PITR) をラップし、タイムスタンプ、復元 label、またはトランザクション ID のいずれかとして、リカバリターゲットを指定できるようにします。

重要：>特定のバックアップの最も早い PITR は、ベースの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間の任意の時点でリカバリする場合は、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 (PostgreSQL 9.4 以降でのみ使用可能)。

相互に排他的な 4 つのオプションのいずれかを使用して、リカバリターゲットを指定できます。

- `--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 [`^ TARGET_NAME`]
- `--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) [`^ RECOVERY_TARGET_IMMEDIATE`]

重要：> `time`、`_XID`、LSN 経由のリカバリターゲットは、バックアップの終了後**に続く必要があります。バックアップのスタートから終了までの間の特定の時点でリカバリする場合は、カタログ内の以前のバックアップからリカバリする必要があります。

[`^ TARGET_NAME`] : PostgreSQL 9.1 以降でのみ利用可能

[`^ RECOVERY_TARGET_IMMEDIATE`] : PostgreSQL 9.4 以降でのみ利用可能

`--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。

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。`_`で述べたように節「をスタート前に、」あなたは、PostgreSQL の文書で詳細を見つけることができます。

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 がシャットダウンされます [`^ TARGET_SHUTDOWN`]

- `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状態、ユーザーがインスタンスを検査できるようにする [`^ TARGET_PAUSE`]
- `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 マスタとして昇格されます [`^ TARGET_PROMOTE`]

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。

[`^ TARGET_SHUTDOWN`] : PostgreSQL 9.5 以降でのみ利用可能

[`^ TARGET_PAUSE`] : PostgreSQL 9.1 以降でのみ利用可能

[`^ TARGET_PROMOTE`] : PostgreSQL 9.5 以降でのみ利用可能

上記の設定の詳細については、[リカバリターゲット設定に関する PostgreSQL 文書] [ターゲット] 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、指定された場合、`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。PostgreSQL の文書。

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、shortcut を受け入れてバックアップを識別します。

newpage

機能の詳細

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。

このリストは網羅的ではありません。多くのシナリオが Barman 設定で機能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。

バックアップ機能

43.1 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のデータ変更のみを保存するタイプの完全定期バックアップです。_WAL 連続アーカイブ_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：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使用できるようになります。

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 [ハードリンク] [8] 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される必要があります。

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 VLD contexts のと高い割合 of_read 専用歴史 tables_を含むこれらのデータベースの特に真です。

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/サーバー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) 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。

最後のノートとして、ユーザーが `barmanbackup` コマンドの `--reuse-backup` ランタイムによって `reuse_backupoption` の設定を上書きすることができます。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、次のようにワン `offincremental` バックアップを実行することができます：

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

43.2 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` および `tablespace_bandwidth_limit` オプションは、> postgres バックアップメソッドではサポートされていません

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 workload 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 = tbname:bwlimit[, tbname:bwlimit, ...]
```

このオプションは、`tablespace` 名前と帯域幅制限 (キロバイト/秒) で構成されるペアのコンマ区切り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、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。

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

```
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。

43.4 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、Barman は_exclusivebackup_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 standbyserver では許可されていません。

また、Barman は 9.2 以上のデータベースサーバーから PostgreSQL のバックアップを同時方式で、主に backup_options 設定パラメータで実行できます。[^ ABOUT_CONCURRENT_BACKUP]

[^ ABOUT_CONCURRENT_BACKUP]：同時バックアップは、バージョン 9.2 以降の PostgreSQL で、streaming レプリケーション protocol（例、pg_basebackup などのツールを使用）を介して利用可能な技術です。

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

重要：同時バックアップ PostgreSQL、9.2、9.3、9.4、および 9.5 のユーザーは、クラスターのすべての PostgreSQL サーバーに pgespresso オープンソースエクステンション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 拡張ウェブサイト] [9] 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、backup_options は透過的に backup_optionsto concurrent_backup を設定する必要があり、下位互換性の PostgreSQL 9.6 の reasons.Users およびそれ以降のバージョンの設定 toexclusive_backup です。

重要：PostgreSQL 9.5 がコミュニティによって EOL と宣言されると、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 はサーバーの `_concurrent` バックアップ `mode_` をアクティブにし、次の 2 つの単純なルールに従います。

- `ssh_command` は宛先 Postgres サーバーを指している必要があります
- `conninfo` は、宛先 Postgres 上のデータベースを指している必要がありますデータベース PostgreSQL 9.2、9.3、9.4、および 9.5、`pgespresso` の使用 `CREATE EXTENSION` を介して正しくインストールする必要があります。9.6 またはさらに、同時バックアップは Postgres ネイティブを介して実行されます API (スタートから停止までアクティブな接続が必要です バックアップの)。

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ているかどうかを判断できません-排他的バックアップの反対> PostgreSQL 自分自身が WAL ファイルを確認する場合正しく>アーカイブされました。フルバックアップは、その WAL ファイルが受信されてアーカイブされるまで一貫性があるとは考えられないことに注意してください>バーマン。Barman 2.5 では、`WAITING_FOR_WALS` と呼ばれる新しい状態が導入されています。これは、`check-backup` コマンドによって管理されます (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)。> Barman 2.10 以降では、`--wait` オプションを `barman backup>` コマンドで使用できます。

43.4.1 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 台のサーバーからのみ必要です。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 から、スタンバイからのバックアップは PostgreSQL 9.4 以上 (バージョン 9.4 および 9.5 が必要> `pgespresso`) でのみサポートされます。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、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。これには、PostgreSQL 9.4 が必要です。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で、スタンバイからアーカイブすることを決定できます。

注：残念ながら、Barman が WAL の重複チェックと [an undocumented behaviours in all versions of PostgreSQL](#) を実行する方法のため、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。

43.5 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 サーバー上のワークロード制御の設定に従って調整されます。これは、バックアップが遅延する可能性があることを意味します。

このデフォルトの動作は、`immediate_checkpointconfiguration global` / 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を使用して `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。

43.6 ローカルバックアップ

免責事項: この機能は、同じサーバー上のバーマンと PostgreSQL 存在として>、稼働の使用を推奨し、故障の同じシングルポイント>のパートであるされていません> 2ndQuadrant の顧客の一部はのためのサポートを追加するために要求してきました>バーマンへのローカルバックアップは、企業によって>配信 24/7 稼働サービスの下で、最も重要なのは、特定の状況下で使用>とします。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、PostgreSQL インスタンスが存在する同じサーバーに Barman をインストールでき、バックアップデータは PGDATA および該当する場合はテーブルスペースとは別のボ

リユ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使用して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、RPO と RTO の観点 betterresults で、2ndQuadrant.For での PostgreSQL の強化、事業継続の経験を承認されていない、2ndQuadrant はまだレプリケーションおよびモニタリングのためにウィットネスサーバのように行動し capableof、バーマンのリモートインストールと theshared ナッシングアーキテクチャを推奨しています。

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 barmanuser の下に Barman をインストールすることを考えると、このユースケースに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 ファイルシステムでは、rsync コマンドをローカルに発行してコピーを実行します（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
...
```

アーカイブ機能### WAL 圧縮

`compression` オプションが設定ファイルで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、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

注意 : `pybzip2` および `pygzip` を除くすべてのメソッドは、新しいプロセスをフォークするために `barman> archive-wal` を必要とします。

44.1 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ー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、streaming connection 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-server コマンドで Barmanreceive-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-server 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 に追加する必要があります。

```
synchronous_standby_names = 'barman_receive_wal'
```

重要：これは設定の例であり、それを示すために> Barman は同期的スタンバイノードとして適格です。> Barman のみを使用することはお勧めしません。> [“同期レプリケーション”] [synch] _ PostgreSQL の文書>このトピックに関する詳細情報。

設定をリロードするには、PostgreSQL サーバーを再起動する必要があります。

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: 0/9000098
* 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   : 0/9000098 (diff: 0 B)
*   Write location  : 0/9000098 (diff: 0 B)
*   Flush location  : 0/9000098 (diff: 0 B)
```

カタログ管理機能###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` というグローバル/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値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、偶発的な `barman delete` 操作から保護されます。

重要：>保持ポリシーの設定が最小冗長要件と競合しないことを確認してください。このトピック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。

45.1 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。

バックアップ保持ポリシーは、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 ファイルを満たすために必要な定期的なバックアップを保持します。

バーマンユーザーは、`* backupredundancy *` (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

- ・：リカバリは、リカバリ・ウィンドウ中の任意の時間にポイントインタイムリカバリのために 1 つの DBA は、一定の期間を指定しているバーマンバックアップ保存ポリシーの種類、およびバックアップのバーマン性を保証保持および/または WAL をアーカイブに必要なファイルです。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時間を遡ります。例、保持ポリシー日間のリカバリ、および現在の時刻に設定されている場合は、金曜日の午前 9 時 30 分で、バーマンは 9:30 AM にポイントインタイムリカバリのバックを可能にするために必要なバックアップを保持します前金曜日。

45.1.1 スコープ

保持ポリシーは以下に対して定義できます。

- ・ **** PostgreSQL 定期ベースバックアップ**** : `retention_policy` を使用設定オプション
- ・ **アーカイブログ、Point-In-Time-Recovery の場合** : `wal_retention_policy` 構成オプション

重要 : >時間ディメンションでは、アーカイブログを定期的なバックアップの時間ウィンドウに含める必要があります。

ここには、2 つの典型的な使用例があります。完全または部分的な特定時点回復です。

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の保持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 週間のアーカイブログを保持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` に制約することにより、フルポイントインタイム>リカバリシナリオのみを実装しています。

45.1.2 仕組み

バーマン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ります。

45.1.3 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` はサーバーの最小値以上必要があります冗長レベル（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が逆に値番目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が前に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；逆順の値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。

フックスクリプト

Barman を使用すると、データベース管理者は次の 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

バーマン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 つのタイプのフックスクリプトの唯一の違いは、Barman は戻り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 2 回以上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き、Barman は、スクリプトが SUCCESS (標準リターンコード 0)、ABORT_CONTINUE (戻りコード 62)、または ABORT_STOP (リターンコード 63) のいずれかを返すまで、戻りコードをチェックして無期限に再試行します。バーマンは、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 を返すことで、ユーザーは Barman にメインオペレーションを中断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「pre」フックスクリプト (存在する場合) 3. リトライの「pre」フックスクリプトがで中止されなかった場合の実際のイベント (バックアップオペレーション、または WAL アーカイビング)。リトライの「投稿」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:>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ことを除けば、ABORT_STOP>は ABORT_CONTINUE のように動作します。

46.1 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` : `_hook script_実行_before_` ベースバックアップ、一度だけ、終了コードのチェックなし
- `pre_backup_retry_script` : `retry フック script_実行_before a` 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` : `retry フック script_実行_after a` 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` : `_hook script_実行後_after_` ベースバックアップ、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。`_retry_` 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（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` : バーマンのバージョン

46.2 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オーバーライドできます。

- `pre_delete_script` : `_hook script_` が起動された `_before_` 削除バックアップの 1 回のみ、終了コードのチェックなし
- `pre_delete_retry_script` : `retry フック script_実行_before` 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` : `retry フック script_実行_after` 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` : `_hook script_起動_after_` 削除バックアップの 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、`_retry_` スクリプト、バーテンダーチェックリターンコード（煮上側部分）の場合には任意の終了コード `.Only` を結果ことができます。

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、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）

46.3 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` : `hook script_実行_before` 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、一度だけ、終了コードを確認してください
- `pre_archive_retry_script` : `retry フック script_実行_before` a 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` : `retry フック script_実行_after` a 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` : `hook script_実行_after` WAL ファイルは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、_retry_スクリプト、バーテンダーチェックリターンコード（煮上側部分）の場合には任意の終了コード.Only を結果ことができます。

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

46.4 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、サーバーごとにオーバーライドできます。

- `pre_wal_delete_script` : hook script_が実行される_before WAL ファイルの削除
- `pre_wal_delete_retry_script` : retry フック script_実行_before 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` : retry フック script_実行_after 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` : hook script_が実行された_after 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、_retry_スクリプト、バーテンダーチェックリターンコード（煮上側部分）の場合には任意の終了コード.Only を結果ことができます。

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。

46.5 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 `globalconfiguration` 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` : `_hook script_` が起動された `_before_` リカバリバックアップの 1 回のみ、終了コードのチェックなし
- `pre_recovery_retry_script` : `retry フック script_実行_before` 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` : `retry フック script_実行_after` 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` : `_hook script_` を起動しました `_after_` リカバリバックアップの 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、`_retry_` スクリプト、バーテンダーチェックリターンコード（煮上側部分）の場合には任意の終了コード `.Only` を結果ことができます。

回復スクリプトは、`backupscrip`t と同じ環境変数にプラス、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)

カスタマイズ

47.1 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。

**** TIP : ****>ユーザーは、ランタイム変数データ専用のディレクトリ（例：/var/run/barman）など、揮発性パーティションのディレクトリを使用することをお勧めします。

47.2 バイナリ経路

バージョン 1.6.0 以降、Barman では、グローバル/サーバーオプション `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 はこれらのディレクトリ内を最初に検索し、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。

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 SHIPPING） および [repmgr] [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は、PostgreSQL は toarchive WAL は、バーマン server.Barman にファイルを直接 get-wal フレームワークのおかげで、この目的の WAL hub.For としても使用することができるように構成されている必要があり、あなたは単語の一部、barman-wal-restore スクリプトを使用することができますすべてのスタンバイサーバを含む barman-cli パッケージ。

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。

並列ジョブ

デフォルトでは、Barman はバックアップとリカバリの両方の操作中にファイルコピーに 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でき、これらに対応する構成ファイルに追加します。

```
parallel_jobs = n
```

ここで、`n` は、ファイルの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 4 つの並列 workers を使用することができます。

```
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。`streamingprotocol` を介して構成されたサーバーの場合、Barman は現在 1 ワーカーのみに制限されている `pg_basebackup` に依存します。

地理的冗長性

バックアップサーバーのソースが PostgreSQL サーバーではなく Barman インストールである場合、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。

この機能により、ユーザーは透過的に PostgreSQL バックアップの `_geographically distributed_copies` を保持できます。

バーマン用語、PostgreSQL が定義されているよりも `installationrather` バーマンに接続されているバックアップサーバパッシブノード。A のパッシブノードが `primary_ssh_command` オプションで構成され、プライマリバーマンインストールの完全なレプリカのために（グローバルに `availableboth` で）およびサーバーレベル（混在シナリオの場合、`_direct_`サーバーと `_passive_`サーバーの両方を使用）。

50.1 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内）
- 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。

50.2 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、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するプライマリサーバーへの SSH 接続パラメーターを指定します。

50.3 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-backup、プライマリ node3 で利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。barman sync-wals、プライマリノードで利用可能なすべての WAL ファイルをローカルにコピーオーダー

50.4 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に接続し、バックアップがリモートマシンに存在する場合、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 1 つの同期プロセスのみが許可されます。

WAL ファイルは、次の方法でも同期できます。

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

newpage

Barman クライアントユーティリティ (barman-cli)

以前は別個のオープンソースプロジェクトであった `barman-cli` は、バージョン 2.8 から Barman のコアに統合されており、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。

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
```

クラウド用の Barman クライアントユーティリティ (`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統合をサポートし、WAL ファイルとバックアップを S3 互換オブジェクトストアに中継することにより、PostgreSQL データベースのディザスタリカバリ機能を強化するために拡張されました。

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。

- `barman-cloud-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 サーバーをバイパスして、WAL ファイルを S3 オブジェクトストアに直接出荷する。あるいは、WAL アーカイビングのフックスクリプトとして (`pre_archive_retry_script`) ；
- `barman-cloud-wal-restore` : `restore_command` として使用されるスクリプト S3 オブジェクトストアから WAL ファイルをフェッチし、Barman サーバーをバイパスし、PostgreSQL スタンバイに直接保存します。

- `barman-cloud-backup` : ローカルバックアップを取るために使用されるバックアップスクリプト PostgreSQL サーバー上で直接、それを S3 オブジェクトストアに出荷するために、Barman サーバーをバイパスします。
- `barman-cloud-list-backup` : のコンテンツをリストするために使用されるスクリプト S3 オブジェクトストアから `barman-cloud-backup` で取得した Barman バックアップ。
- `barman-cloud-restore` : バックアップを直接復元するために使用されるスクリプト S3 オブジェクトストアから `barman-cloud-backup` で取得。

クラウドユーティリティの資格情報の設定方法については、Boto 3 文書の [“資格情報” セクション] [`boto3creds`] を参照してください。

**** WARNING :** ******クラウドユーティリティは [`boto3` ライブラリ] を必要 [`boto3`] は、お使いのシステムにインストールされています。

インストール

Cloud 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、S3 互換オブジェクトストア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-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
```

newpage

#トラブルシューティング

BARMAN インストールの診断

以下を使用して、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や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。

ヘルプのリクエスト

バーマン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 : irc : //irc.freenode.net/barman の IRC チャンネル

あなたがバグを発見しイベントに、あなたは GitHub のを使用してチケットを開くことができます。 <https://github.com/2ndquadrant-it/barman/issues>

2ndQuadrant は、24 時間年中無休のサービスを含む Barman の専門的なサポートを提供します。

54.1 バグを提出する

バーマンは広範囲にテストされており、現在いくつかの実稼動環境で使用されています。しかし、いずれのソフトウェアとして、バーマンは、フリーのバグを ISNOT。

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。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提出する際には、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として、>>注意してください。

newpage

#バーマンプロジェクト

サポートとスポンサーの機会

Barman は、2ndQuadrant 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、2ndQuadrant にお問い合わせください。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。

- [バーマンのウェブサイト] [11]
- [サポートセクション] [12]
- [2ndQuadrant ウェブサイト] [13]
- [バーマン FAQ] [14]
- [2ndQuadrant ブログ : Barman] [15]

バーマンへの貢献

2ndQuadrant 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、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。クロスファンクショナルなコラボレーションを通じて厳格なテスト手順を実装する。すべてのバーマンのコミットは、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貢献です。

これが Barman の好ましい開発方法ですが、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、2ndQuadrant で事前に説明します。

また、寄付は、著作権の譲渡と、開発者の雇用主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ます。

この目的で Github のプルリクエストシステムを使用できます。

第

FIFTYSEVEN

著者

アルファベット順に：

- Gabriele Bartolini、gabriele.bartolini@2ndquadrant.it (建築家)
- Jonathan Battiato、jonathan.battiato@2ndquadrant.it (QA /テスト)
- Giulio Calacoci、giulio.calacoci@2ndquadrant.it (開発者)
- Francesco Canovai、francesco.canovai@2ndquadrant.it (QA /テスト)
- Leonardo Cecchi、leonardo.cecchi@2ndquadrant.it (開発者)
- Gianni Ciolli、gianni.ciolli@2ndquadrant.it (QA /テスト)
- Britt Cole、britt.cole@2ndquadrant.com (文書)
- Marco Nenciarini、marco.nenciarini@2ndquadrant.it (プロジェクトリーダー)
- ルーベンス・スーザ、rubens.souza@2ndquadrant.it (QA /テスト)

過去の貢献者：

- カルロアスカーニ
- ステファノ・ピアヌッチ
- ジュゼッペ・ブロッコロ

リンク

- [check-barman] [16] : Barman 用の Nagios プラグイン、Holger によって書かれました。ハマン (MIT ライセンス)
- [puppet-barman] [17] : Puppet 用バーマンモジュール (GPL)
- [CentOS 7 で Barman を使用して PostgreSQL データベースをバックアップ、復元、および移行する方法に関するチュートリアル] [26]、Sadequl Hussain 氏 (DigitalOcean コミュニティで入手可能)
- [BarmanAPI] [27] : バーマン向け RESTful API、Mehmet Emin Karakaş (GPL) によって作成

ライセンスと貢献

Barman は 2ndQuadrant 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 (C) 2011-2017 [2ndQuadrant Limited] [13]。

バーマンは、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] [18] 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。2ndQuadrant 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著作権に関する申し立て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、Barman ディストリビューションのすべて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 CopyrightAssignment Form のコピーについては、info @ 2ndQuadrant.com までお問い合わせください。

newpage

付録

機能マトリクス

以下に、バックアップおよび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。

V e r - s i o n	B a c k u p w i t h r s y n c / S S H	B a c k u p w i t h p g _ b a s e b a c k u p	S t a n d a r d W A L a r c h i v i n g	W A L S t r e a m i n g	R P O = 0
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
					:sub
					:(d)
9.4	Yes	Yes	Yes	Yes	Yes
					:sub
					:(d)
9.3	Yes	Yes ^(c)	Yes	Yes ^(b)	No
9.2	Yes	Yes :s ub:(a)(c)	Yes	Yes :su b:(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) 表領域のバックアップはサポートされていない 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] : コ ミ ッ ト [“pg_receivexlog を 修 正 - 同 期 的”] [49340627f9821e447f135455d942f7d5e96cae6d] 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と 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ーすることをお勧めします。

< ! -参照リンク->

- [rpo]: https://en.wikipedia.org/wiki/Recovery_point_objective [rto]: https://en.wikipedia.org/wiki/Recovery_time_objective [repmgr]: <http://www.2ndquadrant.com/en/projects/repmgr/>

: <http://www.repmgr.org/> [sqldump]: <https://www.postgresql.org/docs/current/静的/backup-dump.html> [physicalbackup]: <https://www.postgresql.org/docs/current/静的/backup-file.html> [pitr]: <https://www.postgresql.org/docs/current/静的/continuous-archiving.html> [adminbook]: <https://www.2ndquadrant.com/en/books/postgresql-10-administration-cookbook/> [wal]: <https://www.postgresql.org/docs/current/静的/wal.html> [49340627f9821e447f135455d942f7d5e96cae6d]: <https://git.postgresql.org/gitweb/?p=postgresql.git>; a = コミット; h = 49340627f9821e447f135455d942f7d5e96cae6d [requirements_recovery]: <https://www.postgresql.org/docs/current/静的/warm-standby.html#STANDBY-PLANNING> [yumpgdg]: <http://yum.postgresql.org/> [aptpgdg]: <http://apt.postgresql.org/wiki/> [wiki]: <https://wiki.postgresql.org/wiki/Apt> [epel]: <http://fedoraproject.org/wiki/EPEL> [man5]: <http://docs.pgbarman.org/barman.5.html> [setup_user]: <https://docs.python.org/3/install/インデックス.html> # alternate-installation-the-user-scheme [pypi]: <https://pypi.python.org/pypi/barman/> [pgpass]: <https://www.postgresql.org/docs/current/静的/libpq-pgpass.html> [pghba]: <http://www.postgresql.org/docs/current/静的/client-authentication.html> [authpghba]: <http://www.postgresql.org/docs/current/静的/auth-pg-hba-conf.html> [streamprot]: <http://www.postgresql.org/docs/current/静的/protocol-replication.html> [ロール]: <http://www.postgresql.org/docs/current/静的/role-attributes.html> [レプリケーションスロット]: <https://www.postgresql.org/docs/current/静的/warm-standby.html#STREAMING-REPLICATION-SLOTS> [synch]: <http://www.postgresql.org/docs/current/静的/warm-standby.html#SYNCHRONOUS-REPLICATION> [ターゲット]: <https://www.postgresql.org/docs/current/静的/recovery-target-settings.html> [2ndqrpmrepo]: <https://rpm.2ndquadrant.com/> [2ndqdebrepo]: <https://apt.2ndquadrant.com/> [boto3]: <https://github.com/boto/boto3> [boto3creds]: <https://boto3.amazonaws.com/v1/文書/api/latest/ガイド/configuration.html>

- [3]: <https://sourceforge.net/projects/pgbarman/files/> [8]: [http://en.wikipedia.org/wiki/Hard link](http://en.wikipedia.org/wiki/Hard_link) [9]: <https://github.com/2ndquadrant-it/pgespresso> [11]: <http://www.pgbarman.org/> [12]: <http://www.pgbarman.org/support/> [13]: <https://www.2ndquadrant.com/> [14]: <http://www.pgbarman.org/faq/> [15]: <http://blog.2ndquadrant.com/タグ/barman/> [16]: <https://github.com/hamann/check-barman> [17]: <https://github.com/2ndquadrant-it/puppet-barman> [18]: <http://4caast.morfeo-project.org/> [20]: <http://www.postgresql.org/docs/current/静的/functions-admin.html> [24]: <http://www.postgresql.org/docs/current/静的/warm-standby.html#STREAMING-REPLICATION> [25]: <http://www.postgresql.org/docs/current/静的/warm-standby.html#STREAMING-REPLICATION>

gresql.org/ docs / current / 静的/app-pgreceivewal.html [26]: https :
//goo.gl/218Ghl [27]: https : //github.com/emin100/barmanapi

- [31]: <http://www.postgresql.org/>