FUJITSU

Fujitsu
Enterprise Postgres 15 for Kubernetes

概説書

Linux

J2UL-OV15-03Z0(00)
2024年1月

まえがき

本書の目的

本書では、Fujitsu Enterprise Postgres for Kubernetesが提供するオペレーター機能の概要を説明します。

本書の読者

本書は、以下のような読者を対象に書かれています。

- オペレーター機能の導入を検討している方
- オペレーター機能を初めて使う方
- オペレーター機能とは何かを学習したい方
- オペレーター機能の機能概要を知りたい方

本書を読むためには、以下の知識が必要です。

- Linux
- Kubernetes
- コンテナ
- オペレーター

本書の構成

本書の構成と内容は以下のとおりです。

第1章 オペレーター機能の特長

オペレーター機能の特長について説明しています。

第2章 必要な操作

必要な操作について説明しています。

付録A オペレーター機能がサポートするOSS

オペレーター機能がサポートするOSSについて説明しています。

略称

本書では、以下の略称を使用しています。

正式名称	略称
Fujitsu Enterprise Postgres	FEP
Grafana, Alertmanager, Prometheus	GAP
Persistent Volume Claim	PVC

マニュアル名の略称

本書では、以下のマニュアルの名称を使用しています。

正式名称	略称
Fujitsu Enterprise Postgres for Kubernetes ユーザーズガイド	ユーザーズガイド

商標

- Linux(R)は、米国およびその他の国におけるLinus Torvaldsの登録商標です。
- Red HatおよびRed Hatをベースとしたすべての商標とロゴは、Red Hat, Inc.の米国およびその他の国における登録商標または商標です。

・ S/390は、International Business Machines Corporation (IBM) の米国およびその他の国における登録商標です。

そのほか、本マニュアルに記載されている会社名および製品名は、それぞれ各社の商標または登録商標です。

輸出管理規制について

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

出版年月および版数

2024年	1月	第3版
2023年	10月	第2版
2023年	4月	初版

著作権

目　次

第1章 オペレーター機能の特長 1
1.1 オペレーター機能とは? 1
1.2 オペレーターの機能 1
1.2.1 クラスタ配備 2
1.2.1.1 FEPクラスタの作成 2
1.2.1.2 FEP pgpool2コンテナの作成 3
1.2.2 高可用性機能 3
1.2.2.1 自動フェイルオーバー 3
1.2.2.2 自動リカバリ 3
1.2.2.3 手動切り替え 3
1.2.3 バックアップリカバリ 3
1.2.3.1 自動バックアップ 3
1.2.3.2 ポイントインタイムリカバリ 4
1.2.4 構成の変更 4
1.2.4.1 パラメータ変更 4
1.2.4.2 リソース変更 4
1.2.4.3 ディスク拡張 4
1.2.5 マイナーバージョンのアップグレード 5
1.2.5.1 マイナーバージョンのアップグレード 5
1.2.6 FEP機能 5
1.2.6.1 FEP機能のサポート範囲 5
1.2.7 監査ログ運用の自動化 5
1.2.8 モニタリングとアラート 6
1.2.8.1 モニタリング 6
1.2.8.2 アラートとイベント 6
1.2.9 レプリカの拡張 6
1.2.9.1 自動スケールアウト 7
1.2.9.2 手動スケールイン/アウト 7
1.2.10 ディザスタリカバリ 7
1.2.11 鍵管理システムを利用した透過的データ暗号化 7
1.3 オペレーターシステムの設定 8

第2章 必要な操作 10
2.1 導入 10
2.2 高可用性(自動フェイルオーバー/自動リカバリ) 10
2.3 構成の変更 10
2.4 アップグレード 11
2.4.1 マイナーバージョンのアップグレード 11
2.4.2 メジャーバージョンのアップグレード 11
2.5 クラスタごとに構成可能なボリューム 11
2.6 Pgpool-IIのデプロイとオペレーターからの FEPクラスタへの接続 11
2.7 バックアップ 12
2.7.1 オペレーターによるバックアップのスケジュール設定 12
2.7.2 オンデマンドバックアップ 12
2.8 オペレーターからのPITRおよび最新のバックアップリストアの実行 12
2.9 モニタリングとアラート 13
2.10 サーバログモニタリング 14
2.10.1 pgBadgerによるログモニタリング 14
2.10.2 PrometheusとElasticsearchによるログモニタリング 15
2.11 レプリカの拡張 15
2.11.1 自動スケールアウト機能を利用する場合 16
2.12 ディザスタリカバリ 17
2.13 FEPサーバコンテナイメージのカスタマイズ機能 17
2.14 クラウドシークレットストアとの統合 17

付録A オペレーター機能がサポートするOSS 19

第1章 オペレーター機能の特長

本章では、オペレーター機能の特長について説明します。

1.1 オペレーター機能とは?

Fujitsu Enterprise Postgresは、運用を管理するためのオペレーター機能を提供します。

オペレーター機能には、複数のコンポーネントがあります。

オペレーター:FEPデータベースの導入、構成の更新、バックアップ、リカバリなど、FEPサーバコンテナのライフサイクルを管理します。

FEPサーバコンテナ:Postgresを実行するためのFEPサーバソフトウェアが含まれています。

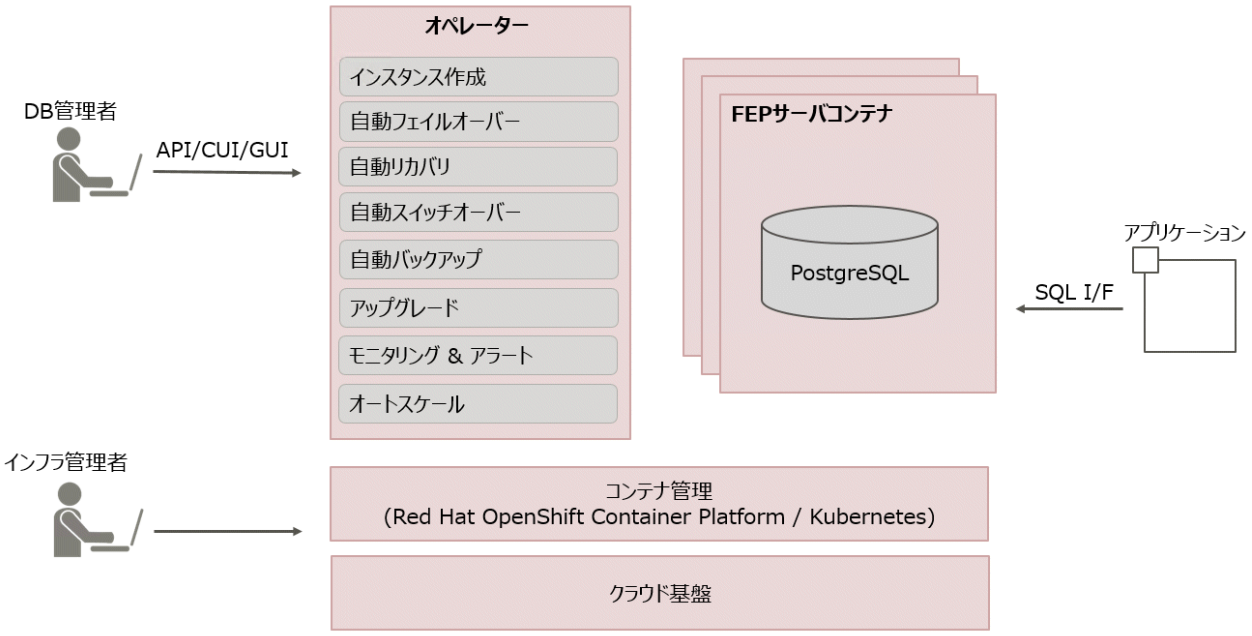
FEPバックアップコンテナ:スケジューリングされたバックアップを実行します。

FEPリストアコンテナ:バックアップデータからのデータ復旧を実行します。

FEP pgpool2コンテナ:Pgpool-IIを使用して負荷分散およびコネクションプーリングを行うためのFEPサーバソフトウェアが含まれています。

FEPエクスポータコンテナ:さまざまなヘルスメトリクスをPrometheusに収集して監視します。

オペレーターは数分で稼働し、運用を開始することができます。



オペレータは、事前に定義した構成で導入し、少ない手順で運用を開始します。デプロイ時およびデプロイ後に構成パラメータを調整することで、インスタンスを適したものに変更することができます。

FEPサーバコンテナは、Fujitsu Enterprise Postgresサーバコンポーネントを組み込むことを意味しています。実行中のFEPサーバコンテナは、原則として、Fujitsu Enterprise Postgresサーバインスタンスと同等であるとみなされます。

1.2 オペレーターの機能

本製品は、お客様のコンテナ管理基盤上のデータベースの構築と運用を自動化するオペレーターサービスを提供します。オペレーターの特徴を以下に示します。

- クラスタ配備
  - FEPクラスタの作成
  - FEP pgpool2コンテナの作成

- 高可用性機能
  - 自動フェイルオーバー
  - 自動リカバリ
  - 手動切り替え
- バックアップ・リカバリ
  - 自動バックアップ
  - ポイントインタイムリカバリ
- 構成の変更
  - パラメータ変更
  - リソース変更
  - ディスク拡張
- マイナーバージョンのアップグレード
- FEP機能
- 監査ログ運用の自動化
- モニタリングとアラート
- レプリカの拡張
- ディザスタリカバリ
- 鍵管理システムを利用した透過的データ暗号化

1.2.1 クラスタ配備

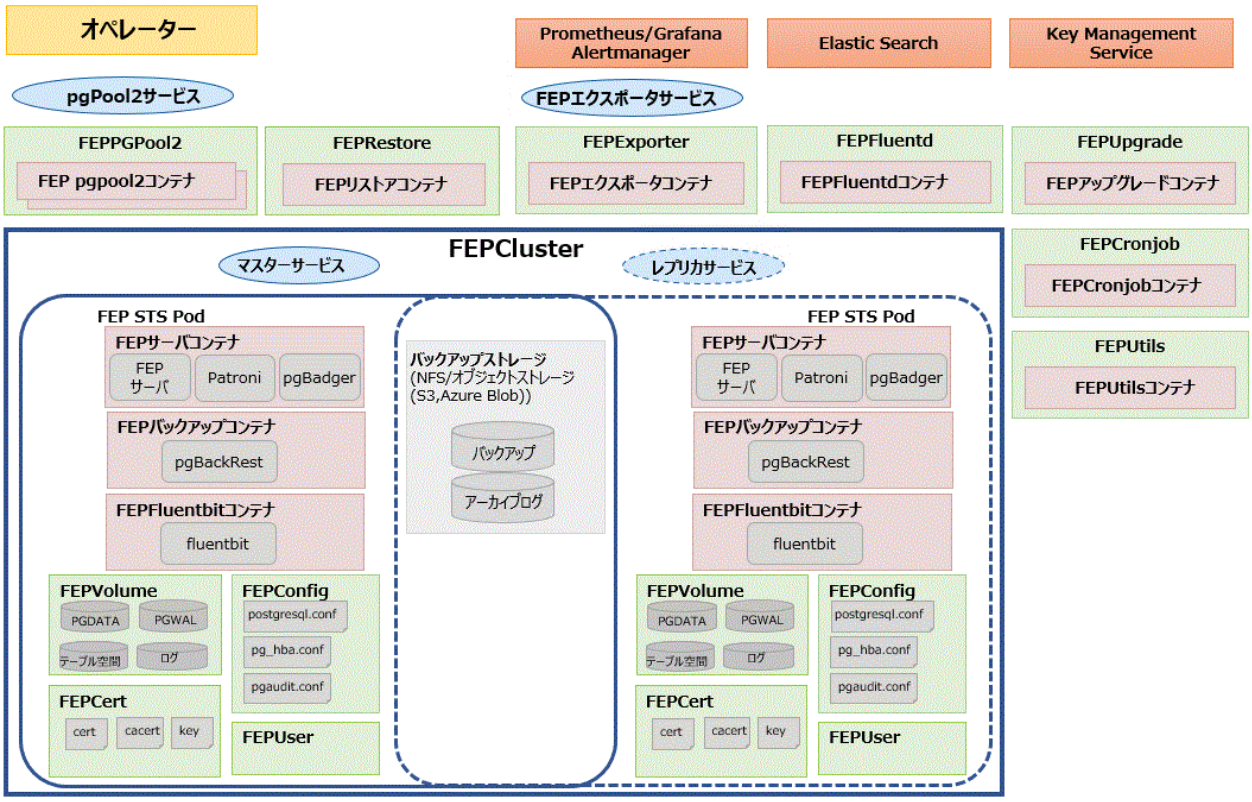
1.2.1.1 FEPクラスタの作成

ユーザーはオペレーターに指示することで、FEPがインストールされたコンテナやボリュームのプロビジョニング、ネットワークリソースを含めたシステムを構築できます。作成されたシステムをFEPクラスタと呼びます。作成されるFEPクラスタは、1台のマスターサーバと2台のレプリカサーバで構成されます。レプリカサーバは同期レプリケーションと非同期レプリケーションから選択できます。デフォルトは非同期レプリケーションです。

FEPクラスタは、以下のコンポーネントで構成されています。

- FEPサーバコンテナ
  - FEPサーバ
  - Patroni
- FEPバックアップコンテナ
- ボリューム用のFEPVolumeカスタムリソース
- データベース・ユーザー用のFEPUserカスタムリソース
- Postgres設定用のFEPConfigカスタムリソース
- TLS証明書、キーストアパスフレーズ用のFEPCertカスタムリソース

以下の図は、1つのマスターPodと1つのレプリカPodを持つFEPクラスタを示しています。



1.2.1.2 FEP pgpool2コンテナの作成

ユーザーは、FEP pgpool2コンテナによって、負荷分散とコネクションプーリングのためにPgpool-IIをデプロイすることができます。

複数のFEP pgpool2 Podをデプロイすることで、可用性を高めることができます。

1.2.2 高可用性機能

1.2.2.1 自動フェイルオーバー

マスターサーバのコンテナやPodで異常を検知すると、自動フェイルオーバーによりレプリカサーバをマスターサーバに昇格し、データベースの接続先を切り替えます。データベースのコネクションは切断されますが、再度アプリケーションからコネクションを確立することで、再接続可能となります。

1.2.2.2 自動リカバリ

マスターサーバで異常が発生し自動フェイルオーバした場合、異常となった旧マスターサーバのPodやコンテナは自動で再起動され、レプリカサーバとして再度クラスタに組み込まれます。

レプリカサーバで異常が発生した場合も、自動で再起動され、レプリカサーバとして再度クラスタに組み込まれます。

1.2.2.3 手動切り替え

手動でレプリカサーバをマスターサーバに切り替えることができます。この場合、旧マスターサーバはレプリカサーバになります。

1.2.3 バックアップリカバリ

1.2.3.1 自動バックアップ

定期的なバックアップを取ることで、データベースのダウンや、アプリケーションエラーによるデータ破損に備えることができます。ユーザーは、任意のスケジュールを設定して自動でバックアップを取得できます。バックアップの種類は、フルバックアップまたはインクリメンタルバッ

クアップです。データベースは、NFS永続ボリュームやAWS S3互換ストレージなどの共有ストレージにバックアップできます。バックアップは、任意の保存期間を設定することにより自動で削除が可能です。

1.2.3.2 ポイントインタイムリカバリ

ポイントインタイムリカバリを使用すると、特定の時間にデータを復旧したり、クラスタを複製して本番環境に移行したりできます。自動バックアップデータから、ポイントインタイムリカバリによりクラスタをリストアすることもできます。リストアは、既存のクラスタにデータをリストアするか、新しいクラスタにデータをリストアするか、選択することができます。また、最新のデータへリストアするか、指定した任意の時刻にリストアするか、選択することができます。

1.2.4 構成の変更

1.2.4.1 パラメータ変更

FEPを構成するパラメータを変更できます。PostgreSQLには2種類のパラメータがあります。即時適用されるものと、FEPサーバプロセスの再起動後に有効になるものです。

FEPを構成するパラメータは、以下のファイルで設定します。

- postgresql.conf
- pg_hba.conf
- pgaudit.conf

注意

即時適用されるパラメータの場合、オペレーターは変更をすべてのFEP Podに適用し、FEPサーバプロセスを自動的にリロードします。クラスタを停止する必要はありません。

FEPサーバプロセスの再始動後に有効になるパラメータの場合、オペレーターはすべてのFEP Podの構成ファイルを更新します。ただし、すべてのFEP PodでFEPプロセスを手動で再起動するには、FEPActionカスタムリソースを使用する必要があります。クラスタを一時的に停止するため、ユーザーはサービスの中断が最も少ない時間に実行する必要があります。

1.2.4.2 リソース変更

FEPサーバコンテナ、FEPバックアップコンテナ、またはFEP pgpool2コンテナに割り当てるCPUおよびメモリリソースの割り当て量は、FEPClusterカスタムリソースを変更することによって変更できます。オペレーターは、変更をStatefulsetに適用します。ただし、新しいリソース割り当てを有効にするためには、すべてのPodを再起動する必要があります。

注意

リソース割り当ての変更は、すぐには有効になりません。新しいリソース割り当てを有効にするためには、すべてのPodを再起動する必要があります。クラスタを一時的に停止するため、ユーザーはサービスの中断が最も少ない時間に実行する必要があります。

1.2.4.3 ディスク拡張

FEPサーバコンテナにマウントされているディスクサイズは、FEPClusterカスタムリソースを変更して拡張することができます。

ディスク拡張は、Prometheusと連携して、ディスクの使用量に応じて自動的に拡張することも可能です。

ディスクの拡張は、KubernetesのPVC拡張機能に対応しているディスクを利用する必要があります。

1.2.5 マイナーバージョンのアップグレード

1.2.5.1 マイナーバージョンのアップグレード

最新のコンテナイメージが提供されている場合、ユーザーはFEPClusterカスタムリソースを変更してマイナーバージョンアップを実行できます。オペレーターはローリングアップデートを実行し、システムの中断を最小限に抑えてマイナーバージョンのアップグレードを実施します。

注意

マイナーバージョンのアップグレードはすぐに有効になります。一時的にデータベースへのコネクションが切断されてしまうため、業務への影響が最も少ない時間に実行することを推奨します。

1.2.6 FEP機能

1.2.6.1 FEP機能のサポート範囲

注意

これらの機能には、FEPクライアントも必要です。

作成されたFEPクラスタは、OSSのPostgreSQL機能に加えて、以下の機能をサポートしています。透過的データ暗号化によってセキュリティとパフォーマンスを強化し、データベースの盗難時にデータの損失を防ぎます。また、カラムナ型インデックスとインメモリレジデント機能により、集計処理を高速化します。各機能の詳細については、“Fujitsu Enterprise Postgres 解説書”を参照してください。

鍵管理システムを利用した透過的データ暗号化については“1.2.11 鍵管理システムを利用した透過的データ暗号化”を参照してください。

機能分類	機能
運用	pgAdmin
	Global Meta Cache
セキュリティ	透過的データ暗号化
	監査ログ
	秘匿化
高性能	インメモリ機能
	高速データロード
アプリケーションインタフェース	Java連携
	ODBC連携
	.NET Framework連携
	埋め込みSQL連携(C言語)
	埋め込みSQL連携(COBOL)

1.2.7 監査ログ運用の自動化

監査ログ運用の自動化は、以下の要件を簡単に実現するための機能です。

- データベース管理者として監査するスキーマを変更したい
- セキュリティ担当者として、ログイン/ログアウトイベントの成功または失敗を監査できるようにしたい
- 会社のポリシーの一環として、監査ログを外部システムに一定期間保存したい

データベース管理者として、以下を指定してください。

- FEPClusterのenableパラメータでpgaudit機能を有効にすると、オペレータにより自動で以下が設定されます
  - pgauditが「shared_preload_libraries」に追加される
  - pgauditログディレクトリが構成される
  - pgauditファイル名が設定される
  - pgaudit 拡張機能が作成される
- pgaudit設定用の外部 ConfigMap を指定します
  - このConfigMapには、pgaudit.confのすべての内容が含まれています
  - Session Audit LoggingとObject Audit Loggingの両方の設定が可能です
  - FEPCluster CRで指定されたpgauditの設定を、このConfigMapの設定内容で上書きします
- pgaudit ログファイルを定期的にアップロードできるように、宛先を指定します
  - Webサーバ
  - Azure Blob
  - AWS S3 ストレージ

注意

Azure BLOB は s390x プラットフォームではサポートされていません。

1.2.8 モニタリングとアラート

1.2.8.1 モニタリング

インフラ管理者は、OSSの監視ツールを使用して、データベースの構築とほぼ同時にデータベースの監視を開始できます。

監視の項目は、PrometheusおよびGrafanaによりビュー形式で提供されます。

監視項目は次のとおりです。

- データベースの状態
- OSパフォーマンス情報
- ディスク使用率
- バックアップの状態
- クライアントの接続情報
- サーバログ
- 監査ログ

1.2.8.2 アラートとイベント

アラートにより、インフラ管理者は異常を即座に把握して対応できます。アラートルールに異常条件を定義し、Prometheusで通知を設定します。メール、Slack、SMS、バックオフィスシステムなどの他のサービスとアラートを統合して、通知を行うことができます。

フェイルオーバー後のアプリケーション・レイヤーでのリカバリ処理の実行、データベースバックアップとの同期、アプリケーション・バックアップの実行などを行うことができます。

1.2.9 レプリカの拡張

参照レプリカの負荷に応じて、参照レプリカを動的に拡張できます。

1.2.9.1 自動スケールアウト

オペレーターは事前に定義されたポリシーに従って参照レプリカを自動的に拡張します。

参照レプリカインスタンスのCPU負荷または接続数によって制御され、指定した閾値を超えると自動的に拡張されます。

自動スケールアウト機能を利用することにより、見込まれる最大の負荷に備えて不要なリソースを使用する必要がなくなります。また、負荷の増加に応じた手動での運用作業が減少します。

注意

- 自動スケールアウト機能によるレプリカの追加は、一度に1つずつ行われます。また追加されたレプリカがサービスを提供できるようになるまでには、環境や格納されたデータ量に依存した時間がかかります。そのため、レプリカの増加が負荷の増加に追随できない場合があります。
- 自動スケールアウト機能によってレプリカ数が増えたとしても、新たに到着する要求は増えたレプリカに優先的に割り当てられるわけではありません。そのためレプリカ数が増えた後も既存のFEPインスタンスの負荷が高い状態が一時的に継続する場合があります。
- 自動スケールアウト機能によって増加するレプリカが処理できる要求は、データベースに対する参照要求のみです。更新を伴う要求は引き続きプライマリFEPインスタンスで処理されます。そのため自動スケールアウト機能が動作してもプライマリFEPインスタンスの負荷が減少しない場合があります。
- 現時点の自動スケールアウト機能では、レプリカの削除(レプリカ数の減少)は行われません。一時的な負荷の増加に伴ってレプリカ数が増加した後に負荷が減少しても、レプリカ数は増えたままとなります。必要に応じて手動でレプリカ数を変更してください。

1.2.9.2 手動スケールイン/アウト

FEPクラスタのカスタムリソースを操作することで、参照レプリカは、いつでもスケールアウトまたはスケールインできます。

1.2.10 ディザスタリカバリ

OSS(pgBackRest)の機能を利用してオブジェクトストレージにバックアップデータを格納することで、異なるOCP環境のデータベースクラスタへのデータの移行を実現します。

災害などでデータベースクラスタを配備したOCP環境での運用が困難な場合でも、異なるOCP環境で運用を継続することが可能になります。

利用可能なディザスタリカバリの方式には、以下の3つがあります。

- バックアップ/リストア方式

  本番環境のデータのバックアップをオブジェクトストレージに格納し、災害発生後にオブジェクトストレージからデータをリストアします。

- 継続的リカバリ方式

  災害発生前から本番環境と災害対策環境の2つのコンテナ環境を作成し、本番環境のデータをオブジェクトストレージに格納し、災害対策環境へ継続的にリストアします。

- ストリーミングレプリケーション方式

  継続的リカバリ方式と同様、災害発生前から本番環境のコンテナ環境と災害対策環境のコンテナ環境を作成します。災害対策環境へのデータの同期には、ストリーミングレプリケーション方式を使用します。

1.2.11 鍵管理システムを利用した透過的データ暗号化

透過的データ暗号化機能で使用するマスタ暗号化キーを鍵管理システムで管理することができます。鍵管理システムを利用することで、業務のセキュリティ要件に沿って暗号化キーを一元的に管理することができます。

オペレーターで利用可能な鍵管理システムは以下のいずれかです。

- KMIPプロトコルを使用する鍵管理サーバ
- AWS の鍵管理サービス
- Azure の鍵管理サービス

注意

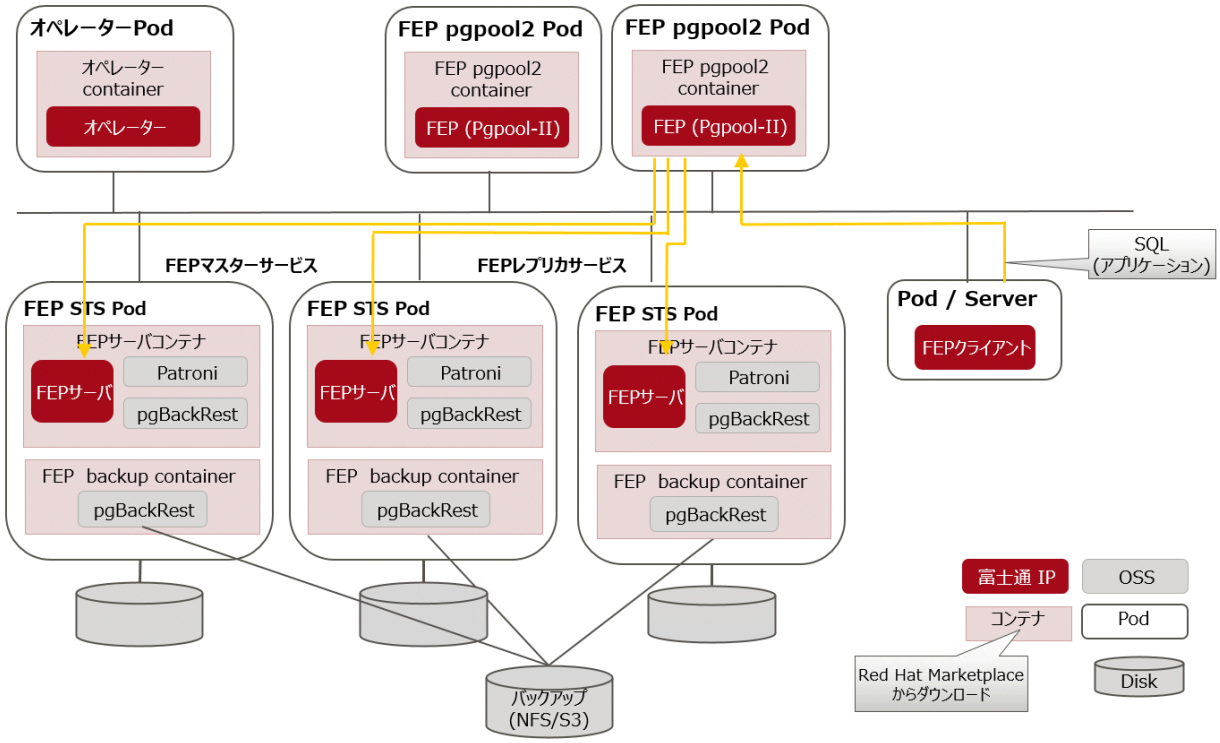
AWSおよびAzureの鍵管理サービスは、CPUアーキテクチャがx86の環境だけで利用できます。

1.3 オペレーターシステムの設定

Pod、コンテナ、サービスの基本的な関係は以下のとおりです。

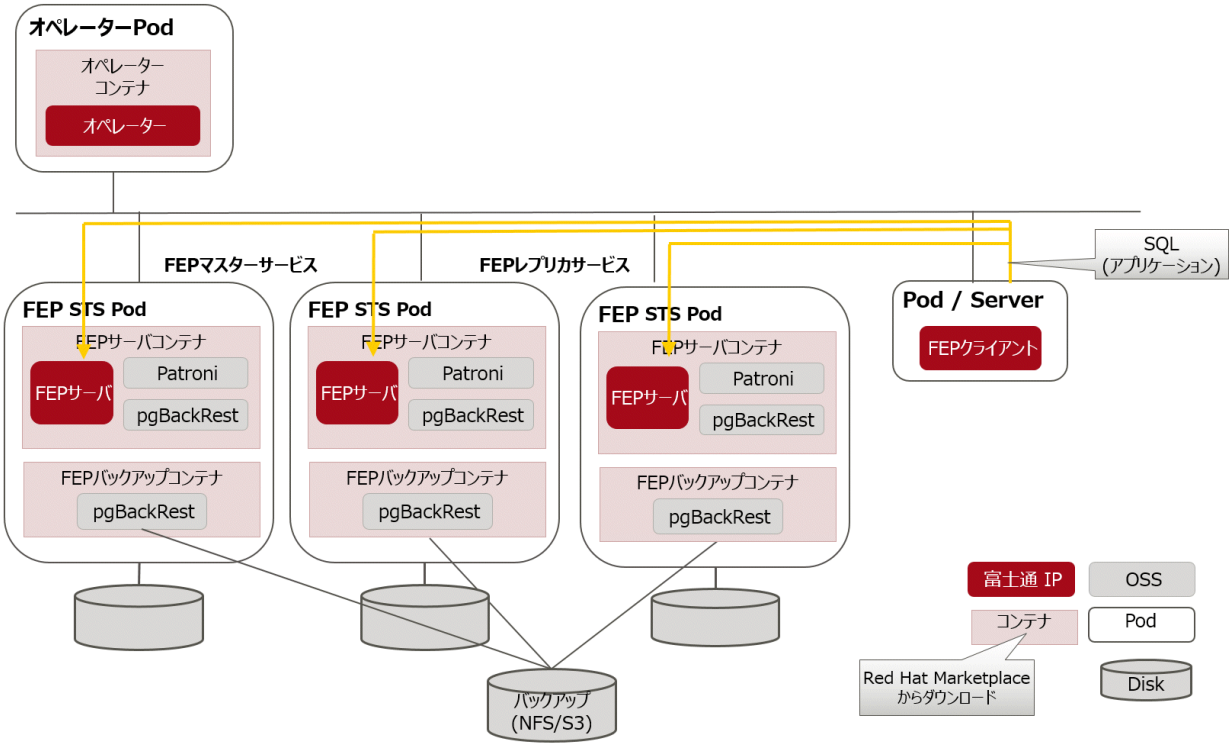
例) Pgpool-IIをデプロイした場合

Pgpool-IIを使用してコネクションプーリングと負荷分散を行います。アプリケーションは、Pgpoolサービスへの接続をポイントします。トランザクションの種類に応じて、Pgpoolは接続をマスターPodまたはレプリカPodに転送します。フェイルオーバ/切り替えが発生すると、FEP pgpool2 Podはトラフィックを新しいFEPマスターPodに送ります。アプリケーションは意識する必要ありません。



例) Pgpool-IIをデプロイしない場合

Pgpool-IIをデプロイしなくても、SQLなどのアプリケーションをFEPクラスタに対して直接実行することもできます。アプリケーションは、FEPマスターサービスへの接続をポイントします。フェイルオーバ/切り替えが発生すると、FEPマスターサービスは自動的に新しいFEPマスターPodをポイントします。アプリケーションは切断されます。接続が再確立されると、新しいFEPマスターPodに接続されます。アプリケーション接続文字列を再構成する必要はありません。


オペレーターPod
オペレーター
コンテナ
オペレーター
FEPマスターサービス
FEPレプリカサービス
SQL
(アプリケーション)
FEP STS Pod
FEPサーバコンテナ
FEPサーバ
Patroni
pgBackRest
FEPバックアップコンテナ
pgBackRest
FEP STS Pod
FEPサーバコンテナ
FEPサーバ
Patroni
pgBackRest
FEPバックアップコンテナ
pgBackRest
FEP STS Pod
FEPサーバコンテナ
FEPサーバ
Patroni
pgBackRest
FEPバックアップコンテナ
pgBackRest
Pod / Server
FEPクライアント
富士通 IP
OSS
コンテナ
Pod
Red Hat Marketplace
からダウンロード
Disk
バックアップ
(NFS/S3)

第2章 必要な操作

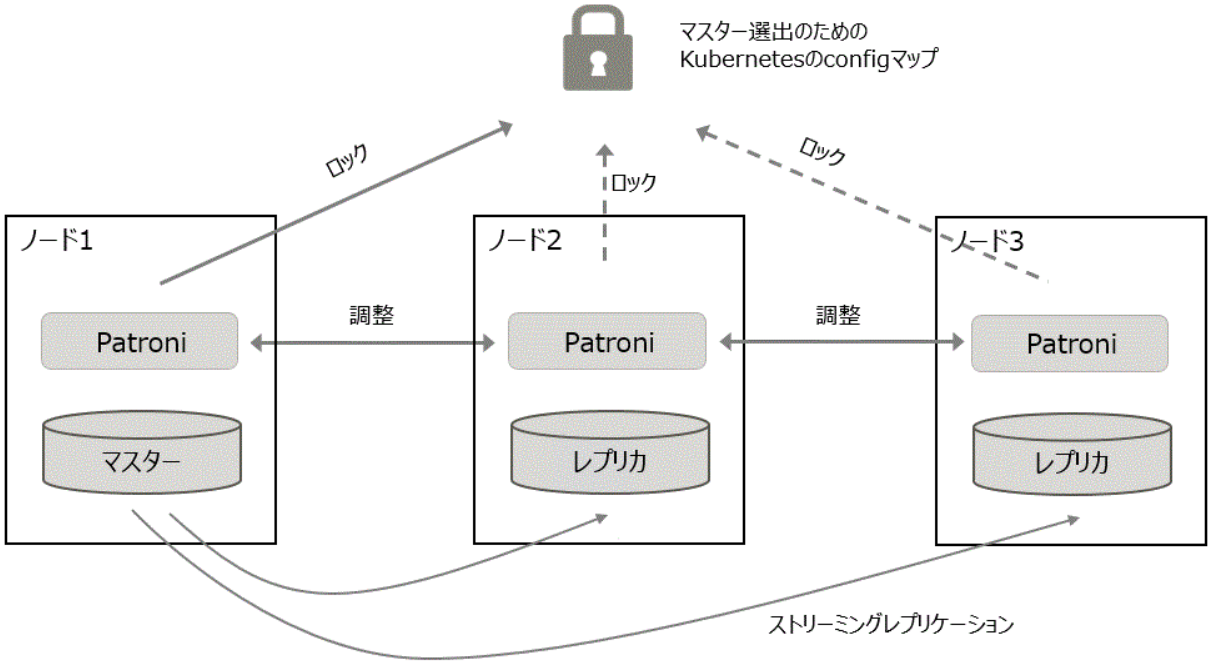
本章では、必要な操作について説明します。

2.1 導入

オペレーターは、バックアップコンテナなどの関連するすべてのコンテナとともに、高可用なFEPクラスタを配備します。

2.2 高可用性(自動フェイルオーバー/自動リカバリ)

FEPの高可用性およびフェイルオーバー管理は、Patroniによって提供されます。PatroniとFEPの両方が同じコンテナイメージにインストールされます。その後、PatroniはFEPインスタンスを初期化して開始します。Patroniは共有リソースのロックを取得します。ロックを取得できるPodがマスターになります。後続のFEPサーバコンテナーが開始されると、PatroniはそのPodをストリーミングレプリケーションのレプリカとして初期化します。



Postgresプロセスのクラッシュや、Postgresが動作しているコンテナの停止など、Patroniがクラスタ内で障害を検出した場合、Patroniは自動的にフェイルオーバーを開始します。

2.3 構成の変更

postgresql.conf、pg_hba.conf、TLS証明書、キーストアのパスフレーズなどのFEP設定の変更には、FEPサーバコンテナの再配備が必要でした。そのため、高可用性環境でクラスタの停止が発生していました。

これらの構成をカプセル化するために、新しいカスタムリソース定義FEPConfigが定義されています。オペレーターは、このカスタムリソース定義を使用してカスタムリソースを監視し、カスタムリソース定義に従って動作し、停止を最小限に抑えます。例えば、リロードで適用が可能なpostgresql.confパラメータが変更された場合、オペレーターはFEPサーバコンテナを再配備する代わりに、FEPデーモンをリロードします。パラメータの変更でFEPの再起動が必要な場合（例えばmax_connections）、オペレーターは構成ファイルを更新しますが、再起動はしません。ユーザーは、定義された手順に従って、スケジュールされたメンテナンス時間にクラスタを手動で再起動できます。

2.4 アップグレード

2.4.1 マイナーバージョンのアップグレード

FEPのマイナーバージョンのアップグレードを実行するには、カスタムリソースを新しいFEPイメージ名で更新します。

まずは、レプリカサーバを新しいコンテナイメージで再起動してアップグレードします。レプリカサーバが複数台ある場合は、1台ずつ順番にアップグレードします。すべてのレプリカサーバがアップグレードされると、そのうちの1台がスイッチオーバにより、新しいマスターサーバに昇格します。次に、元のマスターサーバのコンテナイメージをアップグレードします。

これにより、業務停止を最小限にしてすべてのサーバをアップグレードすることができます。

2.4.2 メジャーバージョンのアップグレード

最新のFEPクラスタを作成する際に、アップグレードに関するパラメタを指定することで、FEPのメジャーバージョンのアップグレードを実行できます。旧版のFEPクラスタからデータをダンプし、新しい最新のクラスタにデータをリストアする処理を自動で実行します。

メジャーバージョンのアップグレード時には、アプリケーションの停止が必要になります。

2.5 クラスタごとに構成可能なボリューム

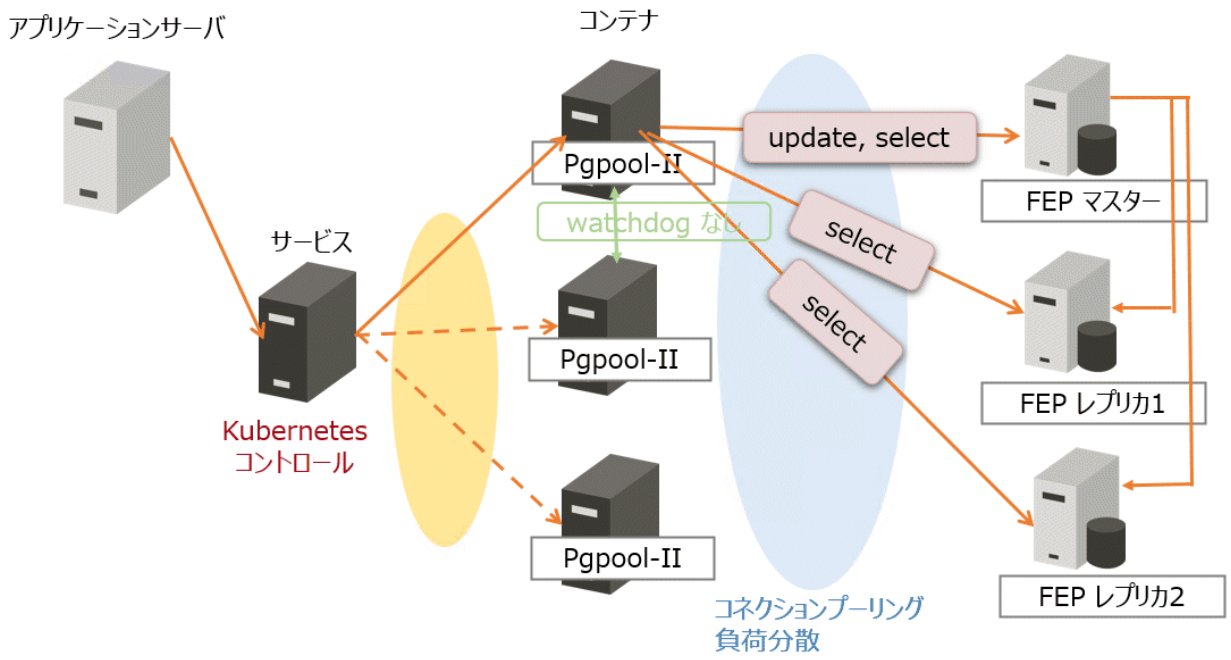
パフォーマンスを向上させるために、データベースファイルとWALファイルを格納するボリュームを分けてください。新しいテーブルスペースに専用のボリュームを使用することもできます。オペレーターは、複数の永続ボリュームを持つFEPクラスタを柔軟に作成し、永続ボリュームに適したストレージクラスを選択できます。

例えば、SSDでバックアップされたストレージクラスのWALボリューム、HDDによってバックアップされたストレージクラス上のログボリュームを持つFEPクラスタを作成することができます。

2.6 Pgpool-IIのデプロイとオペレーターからのFEPクラスタへの接続

ユーザーは、FEP pgpool2コンテナをデプロイし、Pgpool-IIを介してデータベースにアクセスし、負荷分散機能とコネクションプーリング機能を使用できます。

複数のFEP pgpool2コンテナをデプロイして、ロードシェアと高可用性を実現することもできます。ユーザーはKubernetesサービスに依頼して、処理を複数のFEP pgpool2コンテナに分散させることができます。

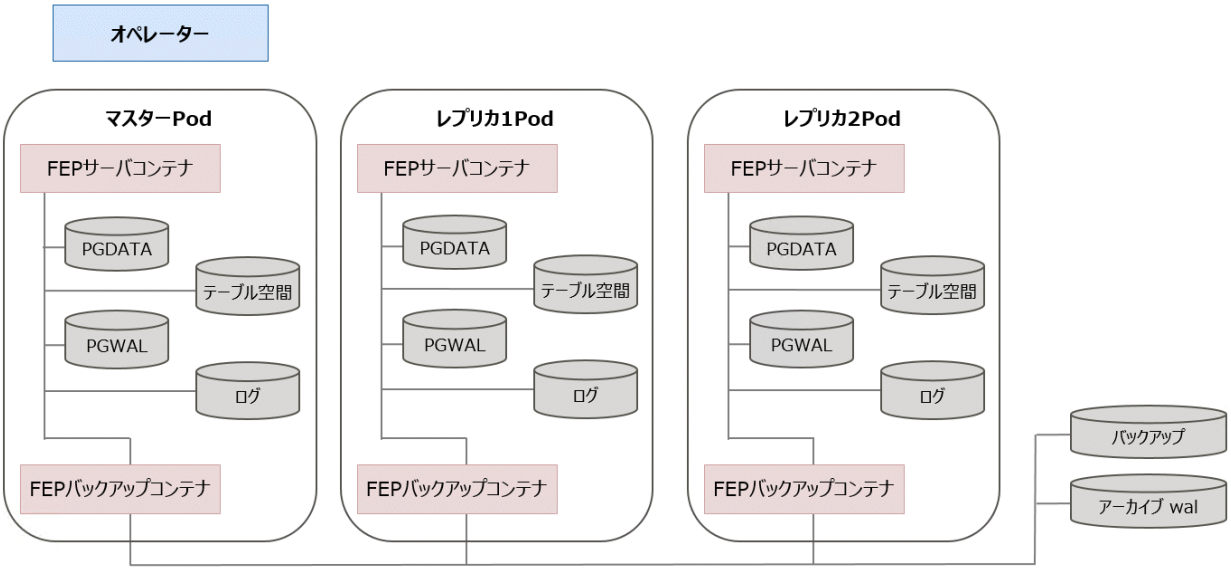


2.7 バックアップ

2.7.1 オペレーターによるバックアップのスケジュール設定

FEPバックアップコンテナは、各FEPサーバPodにサイドカーとしてデプロイされます。バックアップは、ユーザーが設定したスケジュール時刻(crontabのように)に実行されます。FEPバックアップコンテナは、Pod内のFEPサーバがマスターであるかレプリカであるかを判断し、マスターPod上でのみバックアッププロセスを実行します。バックアップとアーカイブされたWALファイルを保存するボリュームは、NFSやAWS S3のような共有ストレージ上になければなりません。

バックアップとWALアーカイブはpgBackRestで行います。



2.7.2 オンデマンドバックアップ

計画的なメンテナンス前や、設定変更後などあらかじめ設定したスケジュール以外の任意のタイミングにオンデマンドバックアップを取得することも可能です。

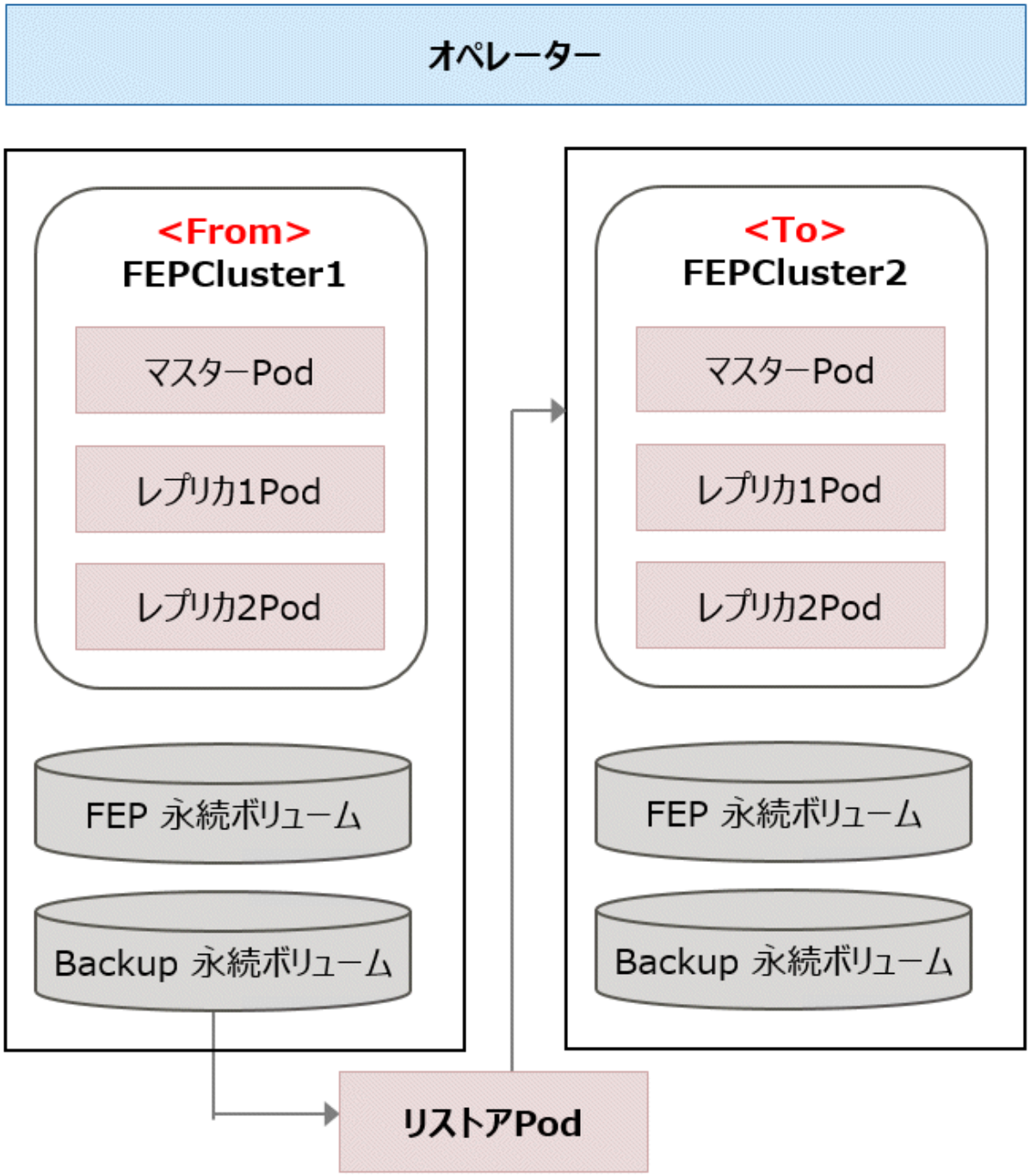
バックアップの格納先や保持期間の設定はスケジュールバックアップと共通の設定となります。

2.8 オペレーターからのPITRおよび最新のバックアップリストアの実行

リストアには、既存のFEPクラスタにバックアップデータをリストアする方法と、新しいFEPクラスタを作成してバックアップデータをリストア方法があります。

前者はIPアドレスや名前などのFEPクラスターの属性を保持し、後者は最初から作成されます。

リストア処理では、リストアコンテナが展開されます。リストアコンテナは、FEPクラスタのマスターサーバにリストアするバックアップデータから復元操作を実行します。データがマスターサーバにリストアされた後、データを2つのレプリカサーバに同期することにより、FEPクラスタが作成されます。



2.9 モニタリングとアラート

モニタリングおよびアラートシステムは、OCP(OpenShift Container Platform)およびKubernetesにデプロイされた標準のGAPスタック（Grafana、Alertmanager、Prometheus）を利用します。オペレーターとFEPClusterをデプロイする前に、GAPスタックが存在している必要があります。

Prometheusは、時系列メトリクスを格納するための簡単な方法です。Grafanaは、Prometheusに保存されているFEPメトリクスのグラフを表示するための、柔軟で視覚的なインターフェースを提供します。

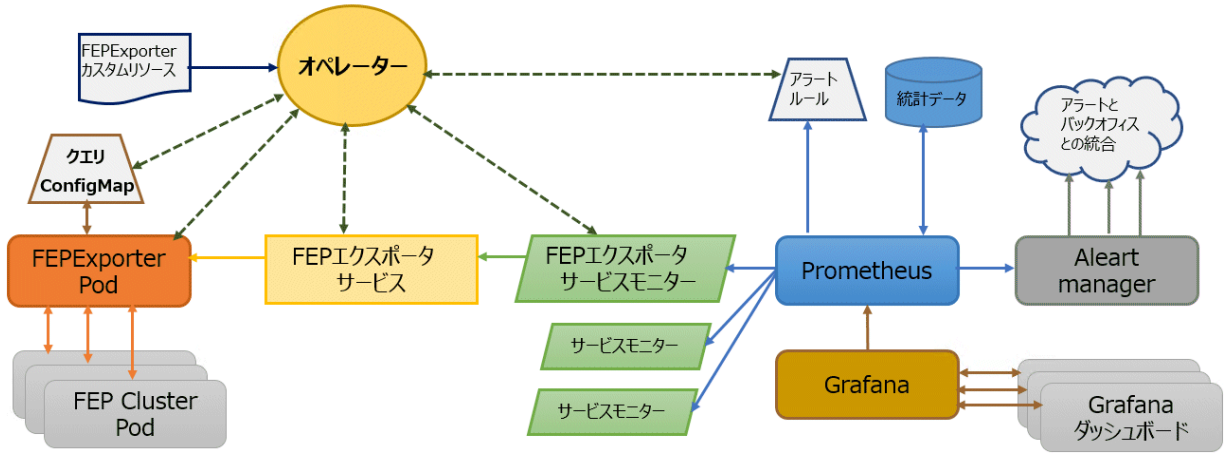
これらを組み合わせることで、ユーザーがスライスおよび分解してFEPデータベースの動作を確認できる大量のメトリクスを保存できます。また、これらのセットアップや使用方法などの問題に対処するための強力なコミュニティがあります。

Prometheusは、FEPコンテナの時系列データのストレージおよびポーリングコンシューマとして機能します。Grafanaは、Prometheusにクエリを実行して、有益なグラフを表示します。

Prometheusルールが定義されている場合は、ルールを定期的に評価して、条件が満たされた場合にAlertmanagerにアラートを送信します。

さらに、Alertmanagerをメール、Slack、SMS、バックオフィスなどの外部システムと統合して、発生したアラートに対してアクションを実行できます。

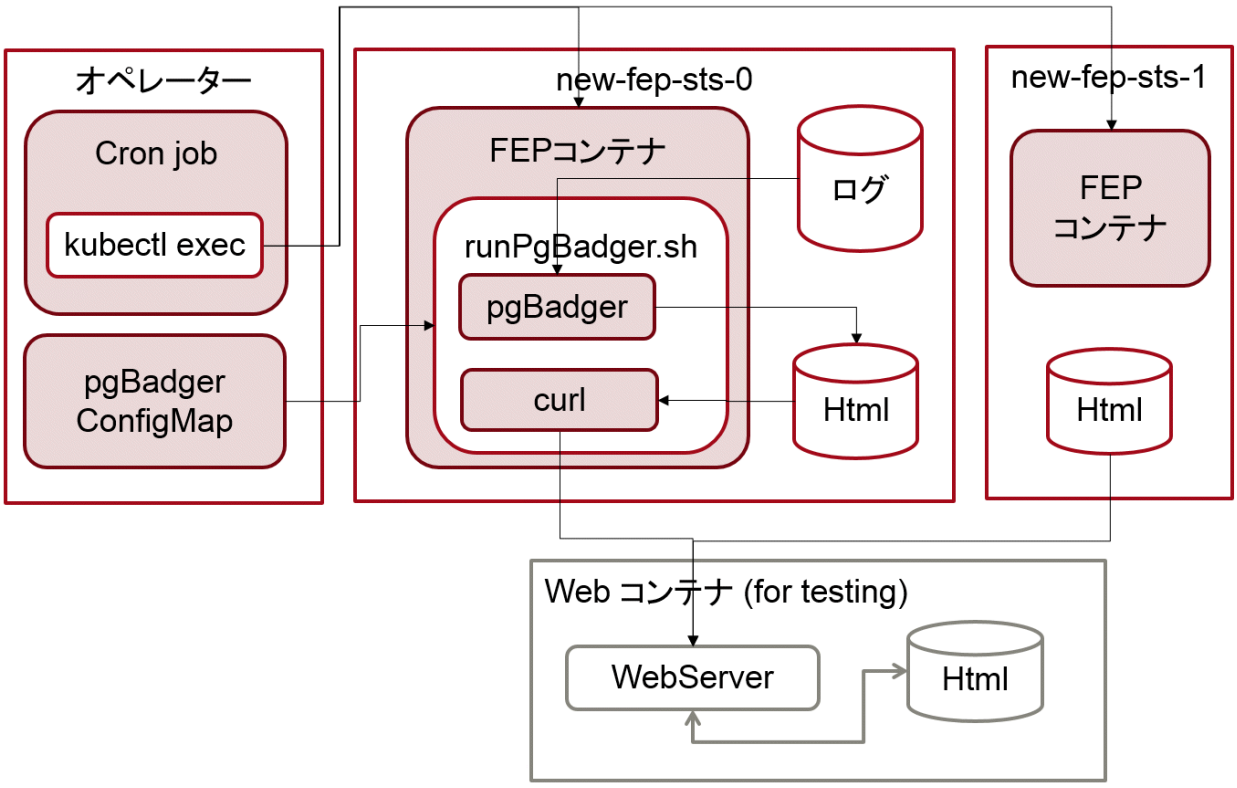
FEPクラスタからのメトリクスは、FEP Exporterを使用してデプロイされたオプションのコンポーネントを介してPrometheusによって収集され、デフォルトのメトリクスセットと対応するPrometheusルールを使用してアラートを生成します。ユーザーは、カスタムメトリクスクエリを定義し、アラート用のカスタムPrometheusルールを定義することで、メトリクスを拡張または上書きできます。



2.10 サーバログモニタリング

2.10.1 pgBadgerによるログモニタリング

pgBadgerは、PostgreSQLのログファイルを解析して、多数の視点からの統計レポートを、日次、週次単位で作成することができます。

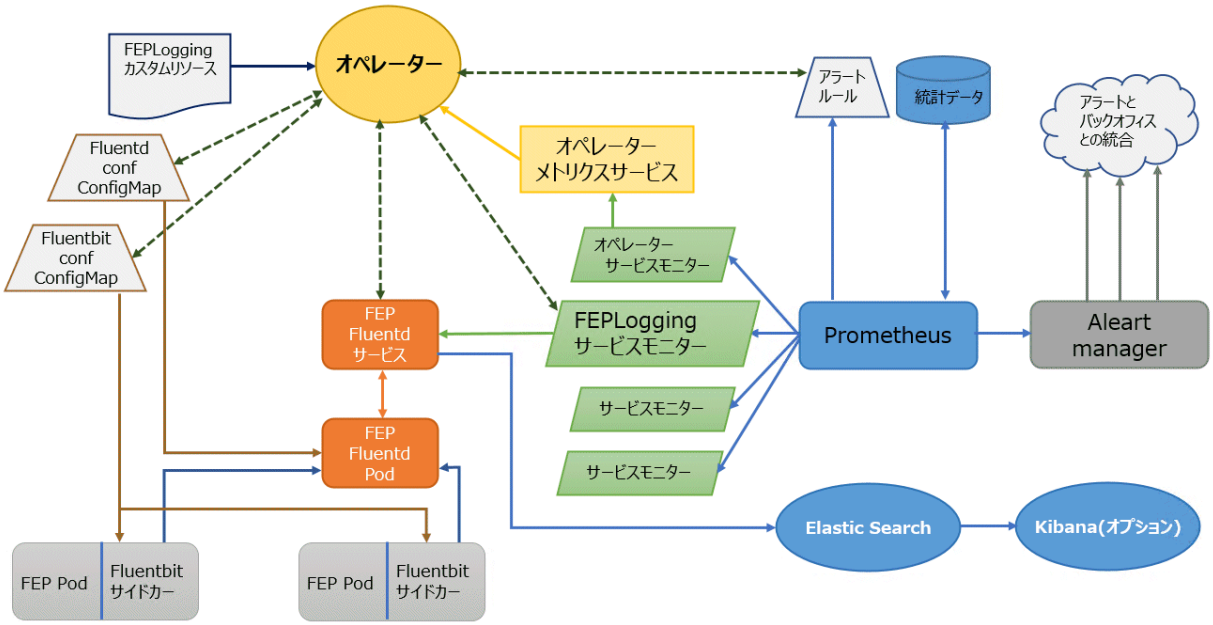


2.10.2 PrometheusとElasticsearchによるログモニタリング

Fluentbitは、patroniとともにFEPClusterのサイドカーとしてデプロイされ、postgresデータベースログを収集します。Fluentdは、Prometheusとの統合に必要なfluent-plugin-prometheusプラグイン、およびElasticsearchとの統合に必要なfluent-plugin-elasticsearchプラグインとともにインストールされます。

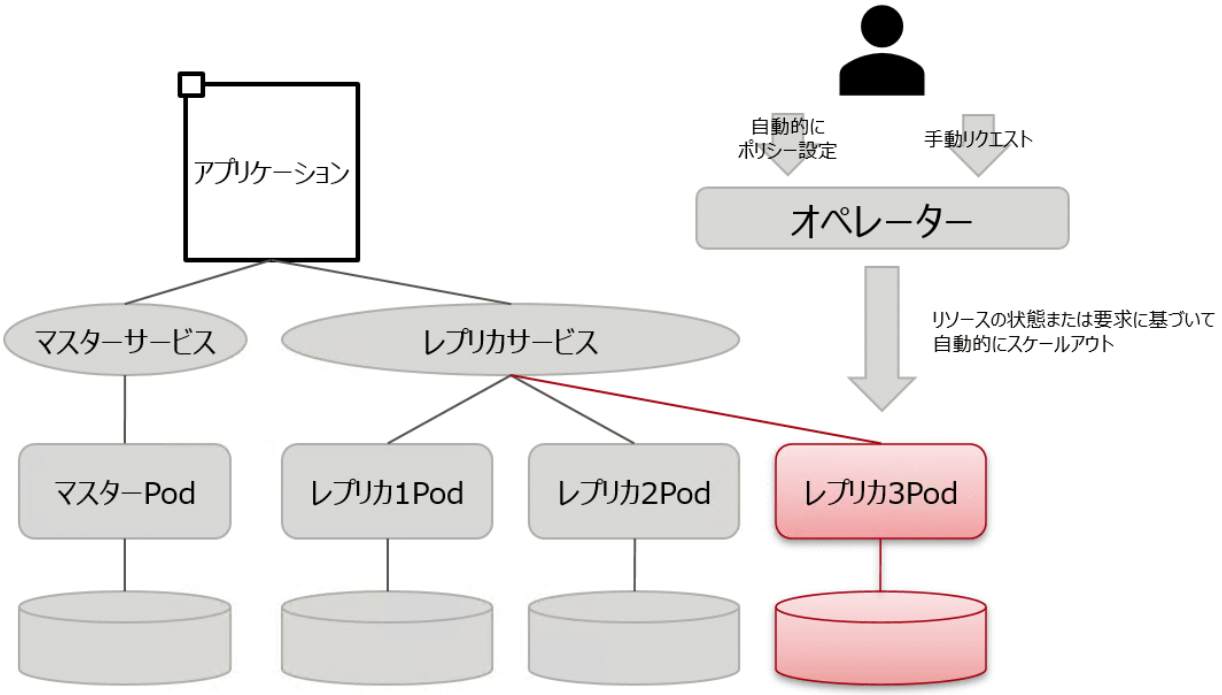
Fluentdは、受信したログレコードをフィルタリングすることによって、さまざまな重要度レベル(パニック、致命的エラー、警告、デバッグなど)の発生をカウントし、そのカウントがログファイル名とともにPrometheusに渡されます。ログファイル名を使用すると、重大度/問題の原因を調査しやすくなります。

elastic searchが有効で設定されている場合、fluentdはkibanaで表示できるelastic searchにログを送信します。



2.11 レプリカの拡張

レプリカの拡張では、参照レプリカのレプリカを自動的に、または、ユーザーが手動で作成します。参照レプリカサービスにクエリを実行すると、自動的に追加されたレプリカインスタンスにクエリを送信できます。



2.11.1 自動スケールアウト機能を利用する場合

自動スケールアウト機能は、Kubernetesクラスタ内のオブジェクトの性能の指標（以下メトリクス）に基づいて動作します。メトリクスはメトリクスサーバと呼ばれる、メトリクスAPIを実装するサービスによって提供されます。

OpenShift/Kubernetesにおけるメトリクスサーバ(メトリクスAPI)には、以下の3種類があります。

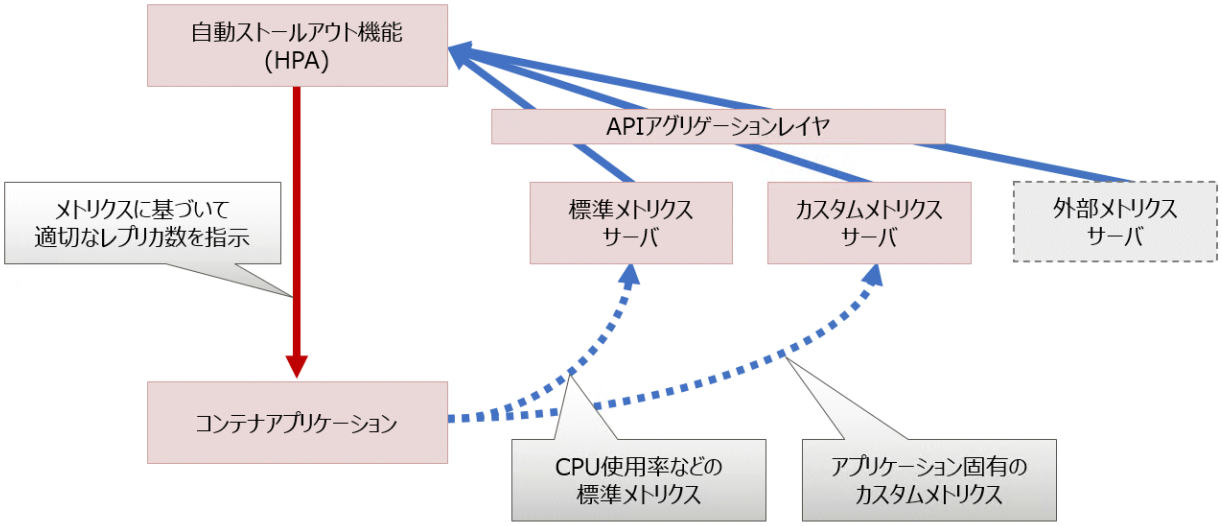
- 標準メトリクスサーバ
- カスタムメトリクスサーバ
- 外部メトリクスサーバ

CPU使用率に基づく自動スケールアウトは、標準メトリクスサーバから取得されるメトリクスを利用して動作します。
コネクション数に基づく自動スケールアウトは、カスタムメトリクスサーバから取得されるメトリクスを利用して動作します。

カスタムメトリクスサーバは、モニタリング機能によってPrometheusに収集されるFEPクラスタのメトリクス情報を参照し、それをメトリクスAPIの形式で公開します。

カスタムメトリクスサーバは、OpenShift/Kubernetesクラスタで共有されるリソースであるため、FEPのインストールの延長では構築、構成されません。コネクション数に基づく自動スケールアウト機能を利用するためには、カスタムメトリクスサーバが存在する必要があります。

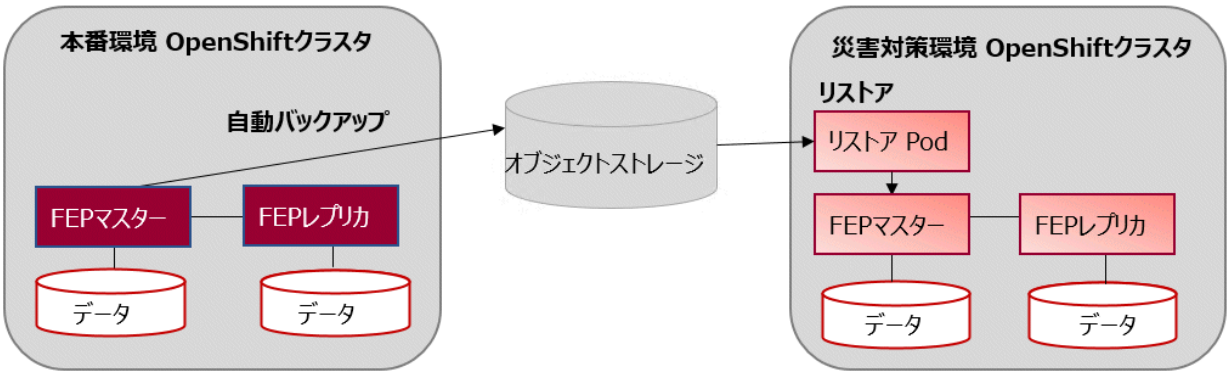
また、そのカスタムメトリクスサーバが、FEPクラスタのカスタムメトリクスを収集しているPrometheusを参照し、FEPクラスタのカスタムメトリクスを公開するように構成されている必要があります。



2.12 ディザスタリカバリ

本番環境の自動バックアップをオブジェクトストレージに取得します。

災害対策環境のOpenShiftクラスタ上でオブジェクトストレージ上のバックアップからFEPクラスタをリストアします。



2.13 FEPサーバコンテナイメージのカスタマイズ機能

Fujitsu Enterprise Postgresは、多数のOSSモジュールをバンドルして PostgreSQLの機能を拡張し、最も要求の厳しいワークロードを処理できます。PostgreSQLの機能をさらに拡張するために、追加のモジュールを含めることをユーザーが希望する場合があります。これに対応するため、FEP サーバコンテナイメージをベースとして使用し、追加モジュールをコンパイルしてベースイメージに追加し、プライベートコンテナレジストリでホストされるカスタマイズされたイメージを作成できます。

2.14 クラウドシークレットストアとの統合

いくつかの独自の機能を提供することで、PostgreSQL のセキュリティを強化します。そのような機能の 1 つに、クラウドシークレットストアとの統合があります。ユーザーは、データベースユーザー パスワードや証明書などのデータベースのシークレットを、次のような外部Secret Storeに保存することを選択できます。

- Azure Key Vault
- AWS Secrets Manager
- GCP Secret Manager
- HashiCorp Vault

クラウドシークレットストアは、Secret Store CSI Driver (https://secrets-store-csi-driver.sigs.k8s.io/) と、Azure、AWS、GCP、および HashiCorp によるそれぞれのプロバイダードライバーを活用して統合します。この統合により、次のことが可能になります。

- クラウドシークレットストアで Postgres のユーザー名とパスワードを管理する
- クラウドシークレットストアで SSL 証明書を管理する

この統合の利点:

- パスワードと証明書は、各 Kubernetes クラスターでローカルに保存されるのではなく、一元化されたクラウドシークレットストアに保存されます
- パスワードと証明書の自動ローテーションを許可します
- FEPクラスタの管理者とクラウドシークレットストアでパスワードと証明書の管理者を別の人にして職務を分離することができます
- クラウドシークレットストアのシークレットへのアクセスは、クラウドプロバイダーでの認証と承認によって制御されます

付録A　オペレーター機能がサポートするOSS

オペレーター機能がサポートするOSSを以下に示します。

OSS名	バージョンレベル	概要	参照先
PostgreSQL	15.5	データベース管理システム	“PostgreSQLDocumentation”
orafce	3.25.1	Oracle互換SQL関数拡張	“Fujitsu Enterprise Postgres アプリケーション開発ガイド”の“Oracleデータベースとの互換性”
Pgpool-II	4.4.4	フェイルオーバ、コネクションプーリング、ロードバランスなど	“Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”の“Pgpool-II”
pg_statsinfo	15.2	統計情報の収集および蓄積	“Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”の“pg_statsinfo”
pg_hint_plan	15.1.5.1	チューニング(統計情報管理、クエリチューニング)	・“Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”の“pg_hint_plan” ・“Fujitsu Enterprise Postgres アプリケーション開発ガイド”の“オプティマイザヒント”
pg_dbms_stats	14.0		・“Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”の“pg_dbms_stats” ・“Fujitsu Enterprise Postgres アプリケーション開発ガイド”の“統計情報の固定化”
pg_repack	1.4.8	テーブルの再編成	“Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”の“pg_repack”
pg_rman	1.3.14	バックアップ／リストア管理	“Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”の“pg_rman”
pgBadger	12.0	ログ解析	“Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”の“pgBadger”
pg_bigm	1.2	全文検索(マルチバイト用)	“Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”の“pg_bigm”
PostgreSQL JDBC driver	42.5.0	JDBCドライバ	“Fujitsu Enterprise Postgres アプリケーション開発ガイド”の“JDBCドライバ”
psqlODBC	13.02.0000	ODBCドライバ	“Fujitsu Enterprise Postgres アプリケーション開発ガイド”の“ODBCドライバ”
pgBackRest	2.46	バックアップ／リストア管理	“ユーザーズガイド”の“オペレーターからのバックアップのスケジュール設定”
patroni	3.1.0	Postgresクラスタ管理	“ユーザーズガイド”の“高可用”

OSS名	バージョンレベル	概要	参照先
postgres_exporter	0.10.1	PrometheusのPostgresqlメトリクス監視機能	“ユーザーズガイド”の“モニタリング”