FUJITSU

Fujitsu
Enterprise Postgres 15 for Kubernetes

ユーザーズガイド

Linux

J2UL-UG15-03Z0(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システム構成、設計、インストール、セットアップ、および操作手順について説明します。

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。

付録D 透過的データ暗号化で利用できる鍵管理システム

透過的データ暗号化で利用できる鍵管理システムを説明しています。

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
Vertical Clustered Index	VCI
Transparent Data Encryption	TDE
Point in time recovery	PITR
Persistent Volume	PV
Persistent Volume Claim	PVC
Universal Base Image	UBI
OpenShift Container Platform	OCP
Mutual TLS	MTLS

マニュアル名の略称

本書では、以下のマニュアルの略称を使用しています。

正式名称	略称
Fujitsu Enterprise Postgres for Kubernetes 概説書	概説書
Fujitsu Enterprise Postgres for Kubernetes ユーザーズガイド	ユーザーズガイド
Fujitsu Enterprise Postgres for Kubernetes リファレンス	リファレンス

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、RPM、CentOS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　6月 第1.1版
2023年　4月　　初版

著作権

目　次

第1章 システム要件

本章では、システム要件について説明します。

1.1 組み込みコンポーネント

FEPサーバコンテナには、以下のコンポーネントが組み込まれています。ただし、これらの構成要素は、保守フェーズにおいてアップグレードが必要となる場合があります。

No	コンポーネント	バージョン	説明
1	Red Hat UBI minimal	9	コンテナ用のOSイメージ
2	Fujitsu Enterprise Postgres Server	15.5	サーバ機能
3	Patroni	3.1.0	クラスタにHA機能およびその他の管理機能を提供

1.2 CPU

以下のCPUアーキテクチャをサポートしています。

No	CPU アーキテクチャ
1	x86
2	s390x
3	ppc64le

1.3 サポートするプラットフォーム

以下のプラットフォームをサポートしています。

No	プラットフォーム		バージョン
1	OpenShift Container Platform		4.11, 4.12, 4.13
2	Rancher Kubernetes Engine (on Linux hosts) (注)		1.4.0+
3	Vmware Tanzu Kubernetes Grid (注)		1.6+
4	Full Managed Kubernetes Service	・ Azure Kubernetes Service ・ Amazon Elastic Kubernetes Service ・ Alibaba Cloud Container Service for Kubernetes ・ Google Kubernetes Engine ・ IBM Cloud Kubernetes Service	1.24, 1.25, 1.26

注: Kubernetes 1.24 - 1.26

OpenShiftまたはKubernetes (AKS、EKS、RKE、ACK, GKE、IKSおよびTKG)でサポートされているストレージが利用可能です。

ただし、バックアップとアーカイブWALのボリュームとして、NFSのような共有ストレージ、またはオブジェクトストレージが必要です。オブジェクトストレージはAmazon Simple Storage Service、Azure Blob Storage、Google Cloud Storageをサポートしています。

1.4 連携ツール

以下のツールとの連携をサポートしています。

No	ツール	バージョン	入手方法
1	Prometheus	・OpenShift インストールされているOpenShiftのバージョン ・Kubernetes - Prometheus v2.36.2以降 - AlertManager v0.24.0以降 ・Rancher Rancher Monitoring Chartで提供されるバージョン	・OpenShift OpenShiftにプレインストールされています。 ・Kubernetes 以下からprometheus-operator(v0.61.1以降)を入手します。 https://github.com/prometheus-operator/prometheus-operator ・Rancher Rancher Monitoring Chartを使用します。
2	Alertmanager		
3	Grafana	・OpenShiftおよびKubernetes Grafana v7.5.17以降 ・Rancher Rancher Monitoring Chartで提供されるバージョン	・OpenShift OperatorHubから入手可能です。 ・Kubernetes 以下からgrafana-operator(v4.7.1以降)を入手します。 https://github.com/grafana-operator/grafana-operator ・Rancher Rancher Monitoring Chartを使用します。
4	Helm	3.10.0 以降	・Kubernetes Helm Web Siteから入手可能です。 https://helm.sh/docs/intro/install/
5	Rancher	v2.7 以降	Rancher Web Siteから入手可能です。 https://rancher.com/
6	Prometheus Adapter	・OpenShiftおよびKubernetes v0.9.1以降 ・Rancher Rancher Monitoring Chartで提供されるバージョン	・OpenShiftおよびKubernetes 以下からPrometheus Adapterを入手します。 https://github.com/kubernetes-sigs/prometheus-adapter ・Rancher Rancher Monitoring Chartを使用します。
7	Elastic Search	8.5.2以降	・OpenShift OperatorHubから入手可能です。 ・Kubernetes 以下から入手可能です。 https://github.com/elastic/helm-charts/tree/main/elasticsearch
8	Velero	v1.11以降	以下で公開されているドキュメントを参照してください。 https://velero.io/docs/main/basic-install/

第2章 オペレーターの設計

本章ではオペレーターの設計について説明します。

2.1 設計作業

オペレーターを利用した導入/運用と設計の要否を以下に示します。

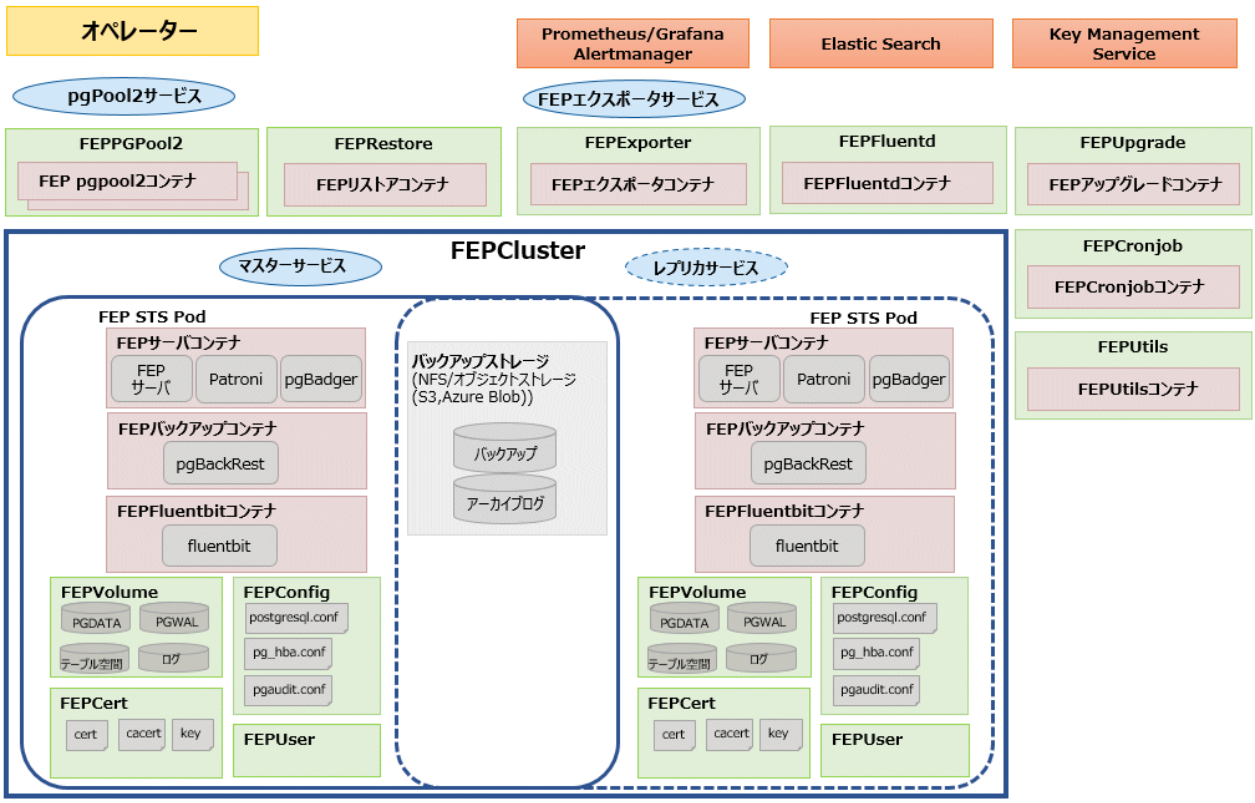
作業	FEPを動作させるための設計の要否	参照先
FEPセットアップ	必須	2.3.1 導入
高可用運用	推奨 (高可用性の動作をチェックまたは変更する場合に実施します。ただし、デフォルトでも高可用なシステム運用は可能です。)	2.3.2 高可用性
ボリューム設定	推奨 (ボリュームを設定する場合に実施します。ただし、デフォルトでも固定ボリュームを割り当てます。)	2.3.3 クラスタごとに構成可能なボリューム
Pgpool-IIのセットアップ	推奨 (Pgpool-IIを使用する場合に実施します。)	2.3.4 Pgpool-IIのデプロイとオペレーターからのFEPClusterへの接続
バックアップ/リストア	推奨 (バックアップおよびリストアを行う場合に実施します。)	2.3.5 オペレーターからのバックアップのスケジュール設定 2.3.6 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
モニタリングおよびアラート(FEPExporter)	推奨 (モニタリングおよびアラートを使用する場合に実施します。)	2.3.8 モニタリングおよびアラート(FEPExporter)
レプリカの拡張	推奨 (スケール機能を使用する場合に実施します。)	2.3.9 レプリカの拡張
鍵管理システム	推奨 (透過的データ暗号化のマスタ暗号化キーの管理を鍵管理システムで行う場合に実施します。)	2.3.11 鍵管理システムを利用した透過的データ暗号化

2.2 システム構成

システム構成について説明します。

2.2.1 サーバ構成

以下にサーバ構成の概要を示します。



構成要素

サーバを構成する各種リソースを以下に示します。

種類	説明
オペレーター	利用者のリクエストを受け付け、データベース構築や運用操作の自動化処理を担うコンテナです。
FEPサーバコンテナ	FEPサーバのコンテナです。
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処理を実行するコンテナです。FEPコンテナと同じPodに作成されます。
FEPFluentbitコンテナ	FEPデータベースのCSVログを収集し、fluentdコンテナに転送して処理します。
FEP pgpool2コンテナ	負荷分散およびコネクションプールを提供するPGPool2のコンテナです。利用しない場合は作成不要です。
FEPリストアコンテナ	リストア処理を実行するコンテナです。リストア処理の実行時に一時的に作成されるコンテナです。
FEPエクスポータコンテナ	統計情報の収集をモニタリングするためにhttp/httpsエンドポイントを公開するコンテナです。
FEPFluentdコンテナ	FEPログの重要度をPrometheusが利用するメトリクスとしてまとめます。必要に応じて、詳細なログ分析のためにElasticsearchにログエントリを転送します。
FEPアップグレードコンテナ	サーバコンテナのメジャーバージョンアップグレード処理を実行するコンテナです。アップグレード処理の実行時に一時的に作成されるコンテナです。
FEPCronjobコンテナ	オペレーターの各機能の定期的な処理の実行時に起動されるコンテナです。
FEPUtilsコンテナ	クラウドシークレット管理のためのコンテナです。
バックアップストレージ	バックアップデータが格納されるストレージです。バックアップを取得する必要がない場合は作成不要です。
FEPCluster	FEPクラスタ定義およびクラスタ構成のためのカスタムリソースです。.
FEPBackup	バックアップ構成のためのカスタムリソースです。

種類	説明
FEPVolume	ボリューム設定のためのカスタムリソースです。
FEPConfig	FEPの環境設定のためのカスタムリソースです。
FEPCert	システム証明書のためのカスタムリソースです。
FEPUser	データベース利用者のためのカスタムリソースです。
FEPAction	アクションを実行するためのカスタムリソースです。
FEPExporter	モニタリングのためのカスタムリソースです。
FEPUpgrade	メジャーアップグレード機能のためのカスタムリソースです。
マスターサービス	マスターのFEPサーバに接続するためのサービスです。
レプリカサービス	レプリカのFEPサーバに接続するためのサービスです。
Pgpool2サービス	Pgpool-IIに接続するためのサービスです。
FEPエクスポータサービス	すべてのFEPClusterノードからメトリクスを収集するサービスです。

2.2.2 ユーザーアカウント

本機能で使用するユーザーアカウントを以下に示します。

インフラ管理者によってオペレーターを操作するアカウントを役割ごとに明確に分離し管理することで、セキュリティ性を高めることが可能です。また、1つのコンテナ管理基盤上で複数のテナントを管理するといった運用も可能となります。

ユーザー種別	ユーザー名	説明
インフラ管理者	任意	オペレーターをインストールするコンテナ管理基盤を管理するシステム管理者(スーパーユーザー)です。
データベース管理者	任意	セットアップや定常運用、保守運用を行う管理者です。
機密管理者	任意	データベースの利用者に対して、各データベース資源に対する適切な権限を設定する管理者です。
アプリケーション開発者	任意	データベースアプリケーションの開発・実行を行います。

2.2.3 コンテナの基本情報

コンテナの基本情報について説明します。

FEPサーバコンテナ

FEPサーバコンテナの命名規則は次のとおりです。

fujitsu-enterprise-postgres-15-server:*OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

バージョンごとに、以下を指定します。

フィールド	値	説明
OS	ubi8/ubi9	
FEPBaseVersion	15	
MajorVersion	1,2, ...	サーバのパッチを含む、イメージのメジャー変更時に使用
MinorVersion	0,1,2 ...	コンテナスクリプトのバグ修正など、イメージのマイナー変更時に使用

最初のパブリッシュでは、以下の名前/タグ付けとなります (マニフェストと子イメージ)。

— fujitsu-enterprise-postgres-15-server:ubi8-15-1.1

- fujitsu-enterprise-postgres-15-server:ubi8-15-1.1-amd64
- fujitsu-enterprise-postgres-15-server:ubi8-15-1.1-s390x
- fujitsu-enterprise-postgres-15-server:ubi8-15-1.1-ppc64le

FEPバックアップコンテナ

FEPバックアップコンテナには、FEPサーバコンテナと同じ命名規則を使用します。

fujitsu-enterprise-postgres-15-backup:*OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

バージョンごとに、以下を指定します。

フィールド	値	説明
OS	ubi8/ubi9	
FEPBaseVersion	15	
MajorVersion	1,2, ...	サーバのパッチを含む、イメージのメジャー変更時に使用
MinorVersion	0,1,2 ...	コンテナスクリプトのバグ修正など、イメージのマイナー変更時に使用

最初のパブリッシュでは、以下の名前/タグ付けとなります (マニフェストと子イメージ)。

— fujitsu-enterprise-postgres-15-backup:ubi8-15-1.1

- fujitsu-enterprise-postgres-15-backup:ubi8-15-1.1-amd64
- fujitsu-enterprise-postgres-15-backup:ubi8-15-1.1-s390x
- fujitsu-enterprise-postgres-15-backup:ubi8-15-1.1-ppc64le

FEPリストアコンテナ

FEPリストアコンテナには、FEPサーバコンテナと同じ命名規則を使用します。

fujitsu-enterprise-postgres-15-restore:*OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

バージョンごとに、以下を指定します。

フィールド	値	説明
OS	ubi8/ubi9	
FEPBaseVersion	15	
MajorVersion	1,2, ...	サーバのパッチを含む、イメージのメジャー変更時に使用
MinorVersion	0,1,2 ...	コンテナスクリプトのバグ修正など、イメージのマイナー変更時に使用

最初のパブリッシュでは、以下の名前/タグ付けとなります (マニフェストと子イメージ)。

— fujitsu-enterprise-postgres-15-restore:ubi8-15-1.1

- fujitsu-enterprise-postgres-15-restore:ubi8-15-1.1-amd64
- fujitsu-enterprise-postgres-15-restore:ubi8-15-1.1-s390x

- fujitsu-enterprise-postgres-154-restore:ubi8-15-1.1-ppc64le

FEPpgpool2コンテナ

FEPpgpool2コンテナには、FEPサーバコンテナと同じ命名規則を使用します。

fujitsu-enterprise-postgres-15-pgpool2: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

バージョンごとに、以下を指定します。

フィールド	値	説明
OS	ubi8/ubi9	
FEPBaseVersion	15	
MajorVersion	1,2, ...	サーバのパッチを含む、イメージのメジャー変更時に使用
MinorVersion	0,1,2 ...	コンテナスクリプトのバグ修正など、イメージのマイナー変更時に使用

最初のパブリッシュでは、以下の名前/タグ付けとなります (マニフェストと子イメージ)。

- fujitsu-enterprise-postgres-15-pgpool2:ubi8-15-1.1
  - fujitsu-enterprise-postgres-15-pgpool2:ubi8-15-1.1-amd64
  - fujitsu-enterprise-postgres-15-pgpool2:ubi8-15-1.1-s390x
  - fujitsu-enterprise-postgres-15-pgpool2:ubi8-15-1.1-ppc64le

FEPExporterコンテナ

FEPExporterコンテナには、FEPサーバコンテナと同じ命名規則を使用します。

fujitsu-enterprise-postgres-exporter: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

バージョンごとに、以下を指定します。

フィールド	値	説明
OS	ubi8/ubi9	
FEPBaseVersion	15	
MajorVersion	1,2, ...	サーバのパッチを含む、イメージのメジャー変更時に使用
MinorVersion	0,1,2 ...	コンテナスクリプトのバグ修正など、イメージのマイナー変更時に使用

最初のパブリッシュでは、以下の名前/タグ付けとなります (マニフェストと子イメージ)。

- fujitsu-enterprise-postgres-exporter:ubi8-15-1.1
  - fujitsu-enterprise-postgres-exporter:ubi8-15-1.1-amd64
  - fujitsu-enterprise-postgres-exporter:ubi8-15-1.1-s390x
  - fujitsu-enterprise-postgres-exporter:ubi8-15-1.1-ppc64le

FEPFluentdコンテナ

FEPFluentdコンテナには、FEPサーバコンテナと同じ命名規則を使用します。

fujitsu-enterprise-postgres-fluentd: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

バージョンごとに、以下を指定します。

フィールド	値	説明
OS	ubi8/ubi9	
FEPBaseVersion	15	
MajorVersion	1,2, ...	サーバのパッチを含む、イメージのメジャー変更時に使用
MinorVersion	0,1,2 ...	コンテナスクリプトのバグ修正など、イメージのマイナー変更時に使用

最初のパブリッシュでは、以下の名前/タグ付けとなります (マニフェストと子イメージ)。

- fujitsu-enterprise-postgres-fluentd:ubi8-15-1.1
  - fujitsu-enterprise-postgres-fluentd:ubi8-15-1.1-amd64
  - fujitsu-enterprise-postgres-fluentd:ubi8-15-1.1-s390x
  - fujitsu-enterprise-postgres-fluentd:ubi8-15-1.1-ppc64le

FEPFluentbitコンテナ

FEPFluentbitコンテナには、FEPサーバコンテナと同じ命名規則を使用します。

fujitsu-enterprise-postgres-fluentbit: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

バージョンごとに、以下を指定します。

フィールド	値	説明
OS	ubi8/ubi9	
FEPBaseVersion	15	
MajorVersion	1,2, ...	サーバのパッチを含む、イメージのメジャー変更時に使用
MinorVersion	0,1,2 ...	コンテナスクリプトのバグ修正など、イメージのマイナー変更時に使用

最初のパブリッシュでは、以下の名前/タグ付けとなります (マニフェストと子イメージ)。

- fujitsu-enterprise-postgres-fluentbit:ubi8-15-1.1
  - fujitsu-enterprise-postgres-fluentbit:ubi8-15-1.1-amd64
  - fujitsu-enterprise-postgres-fluentbit:ubi8-15-1.1-s390x
  - fujitsu-enterprise-postgres-fluentbit:ubi8-15-1.1-ppc64le

FEPCronjobコンテナ

FEPCronjobコンテナには、FEPサーバコンテナと同じ命名規則を使用します。

fujitsu-enterprise-postgres-cronjob: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

バージョンごとに、以下を指定します。

フィールド	値	説明
OS	ubi8/ubi9	
FEPBaseVersion	15	
MajorVersion	1,2, ...	サーバのパッチを含む、イメージのメジャー変更時に使用
MinorVersion	0,1,2 ...	コンテナスクリプトのバグ修正など、イメージのマイナー変更時に使用

最初のパブリッシュでは、以下の名前/タグ付けとなります (マニフェストと子イメージ)。

- fujitsu-enterprise-postgres-cronjob:ubi8-15-1.1
  - fujitsu-enterprise-postgres-cronjob:ubi8-15-1.1-amd64
  - fujitsu-enterprise-postgres-cronjob:ubi8-15-1.1-s390x
  - fujitsu-enterprise-postgres-cronjob:ubi8-15-1.1-ppc64le

FEPアップグレードコンテナ

FEPアップグレードコンテナには、FEPサーバコンテナと同じ命名規則を使用します。

fujitsu-enterprise-postgres-15-upgrade: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

バージョンごとに、以下を指定します。

フィールド	値	説明
OS	ubi8/ubi9	
FEPBaseVersion	15	
MajorVersion	1,2, ...	サーバのパッチを含む、イメージのメジャー変更時に使用
MinorVersion	0,1,2 ...	コンテナスクリプトのバグ修正など、イメージのマイナー変更時に使用

最初のパブリッシュでは、以下の名前/タグ付けとなります (マニフェストと子イメージ)。

- fujitsu-enterprise-postgres-15-upgrade:ubi8-15-1.1
  - fujitsu-enterprise-postgres-15-upgrade:ubi8-15-1.1-amd64
  - fujitsu-enterprise-postgres-15-upgrade:ubi8-15-1.1-s390x
  - fujitsu-enterprise-postgres-15-upgrade:ubi8-15-1.1-ppc64le

FEPUtilsコンテナ

FEPUtilsコンテナには、FEPサーバコンテナと同じ命名規則を使用します。

fujitsu-enterprise-postgres-15-utils: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

バージョンごとに、以下を指定します。

フィールド	値	説明
OS	ubi8/ubi9	

フィールド	値	説明
FEPBaseVersion	15	
MajorVersion	1,2, ...	サーバのパッチを含む、イメージのメジャー変更時に使用
MinorVersion	0,1,2 ...	コンテナスクリプトのバグ修正など、イメージのマイナー変更時に使用

最初のパブリッシュでは、以下の名前/タグ付けとなります (マニフェストと子イメージ)。

- fujitsu-enterprise-postgres-15-utils:ubi8-15-1.1
  - fujitsu-enterprise-postgres-15-utils:ubi8-15-1.1-amd64
  - fujitsu-enterprise-postgres-15-utils:ubi8-15-1.1-s390x
  - fujitsu-enterprise-postgres-15-utils:ubi8-15-1.1-ppc64le

2.3 各機能の設計観点

各機能を設計する上での設計のポイントを説明します。

postgresql-cfg のフォーマット

postgresql-cfgは、PostgreSQLlパラメータを含むConfigMapを表します。このファイルには、インスタンスのpostgresql.confに反映する必要があるパラメータが含まれています。Patroniはpostgresql.confによって認識されていないすべてのパラメータを無視するため、FEPパラメータを特別な方法で処理するためのアプローチが定義されています。

ConfigMapの内容は、key=value形式で定義します。以下の表に詳細を示します。

仕様	例	補足
コンテンツには複数のキーと値のペアが含まれる場合があります。	foo=bar foo1=bar1	-
引用符がない限り、値にスペースを含めることはできません。	foo=bar bar2	無効
引用符で囲まれた値の後に、別の値は指定できません。	foo='bar bar2' something	無効
キーと値のペアには「=」記号が必要です。	-	-
キーと値のペアの前/後/間に空白を使用できます。	foo = bar	-
'#'以降のコンテンツはすべて無視されます。	#コメントと見なされますfoo=bar #コメントと見なされます	-
値は一重引用符で囲むことができます。	foo='bar bar2'	-
一重引用符は、2つの一重引用符でエスケープできます。	foo='It''s ok'	注: 一重引用符は、Patroniedit-configコマンドではサポートされていません。
patronictl edit-configコマンドを呼び出すと、円記号「¥」は「¥¥」に置き換えられます。	-	コマンドラインのエスケープを回避できます。

仕様	例	補足
キーと値のペアが無効な場合、それらは無視されます。アップデートは次のペアの処理を続行します。	foobar foo2=bar2	“foobar”は無視されます。
コンテナスクリプトは、正しい形式である限り、キーと値を検証しません。	-	-

psqlのshowコマンドを使用して、パラメータが正しく設定されていることを確認することをお勧めします。

2.3.1 導入

FEPClusterのための情報

Kubernetesコマンド: kubectl apply -f FEPClusterCR.yaml

この操作により、FEPClusterCR.yamlで提供された情報を使用してFEPClusterが作成されます。

詳細については、“リファレンス”の“FEPClusterパラメータ”を参照してください。

2.3.2 高可用性

高可用性機能を使用するための設定について説明します。

裁定

Patroniは、FEPインスタンスの起動、シャットダウン、ステータスを制御および監視し、マスターインスタンスに障害が発生した場合にフェイルオーバーをトリガーするために使用されます。これは、ソリューションで重要な役割を果たします。特にマスターPodでPatroniプロセスが通知なしに停止した場合、PodはPatroniクラスタロックを更新しません。これにより、実行中のマスターで対応する修正措置が行われずに、レプリカの1つへの不要なフェイルオーバーがトリガーされる可能性があります。また、これにより、スプリットブレインの問題が発生する可能性があります。Patroniのステータスを監視して、実行されていることを確認することが重要です。これは、活性プローブを使用して行われます。ユーザーによる設定は想定していないことに注意してください。

```
livenessProbe:
  httpGet:
    scheme: HTTP
    path: /liveness
    port: 25001
  initialDelaySeconds: 30
  periodSeconds: 6
  timeoutSeconds: 5
  successThreshold: 1
  failureThreshold: 3
```

2.3.3 クラスタごとに構成可能なボリューム

クラスタノード(Pod)のボリュームは、FEPClusterカスタムリソースのfepChildCrValのstorageセクションに設定された値に従って作成されます。

注意

- 最初にFEPClusterを作成した後は、新しいボリュームを後で追加したり、storageClassまたはaccessModesを変更したりすることはできません。
- 基となるstorageClassがサイズの動的変更をサポートしている場合に限り、最初に作成されたボリュームのサイズを変更できます。

以下は、FEPClusterカスタマリソースのstorageセクションのスキーマです。

フィールド	省略可否	サブフィールド	省略値	説明
archivewalVol	可	size	1Gi	アーカイブログのボリュームサイズです。

フィールド	省略可否	サブフィールド	省略値	説明
				“Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”の“データベースのディスク容量の見積り”を参照し、サイズ設計を行ってください。
		storageClass	プラットフォームのデフォルトになります。	ストレージクラスは開始時にのみ設定されます。
		accessModes	ReadWriteOnce	アクセスモードは開始時にのみ設定されます。
backupVol	可	size	2Gi	バックアップのボリュームサイズです。 下記の計算式をもとに見積もりを行ってください。 (full backupの世代数 + incr backupの世代数 + 1) * dataVolのサイズ
		storageClass	プラットフォームのデフォルトになります。	ストレージクラスは開始時にのみ設定されます。
		accessModes	ReadWriteOnce	アクセスモードは開始時にのみ設定されます。
dataVol	不可	size	2Gi	データのボリュームサイズです。 “Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”の“データベースのディスク容量の見積り”を参照し、テーブルサイズ／インデックスサイズを基に設計を行ってください。
		storageClass	プラットフォームのデフォルトになります。	ストレージクラスは開始時にのみ設定されます。
		accessModes	ReadWriteOnce	アクセスモードは開始時にのみ設定されます。
logVol	可	size	1Gi	ログのボリュームサイズです。 ログ出力のレベル(デフォルト:WARNING)を変更して運用する場合や監査ログ機能を有効化する場合には、実際のログ出力量をテスト環境で計測してください。
		storageClass	プラットフォームのデフォルトになります。	ストレージクラスは開始時にのみ設定されます。
		accessModes	ReadWriteOnce	アクセスモードは開始時にのみ設定されます。
tablespaceVol	可	size	512Mi	テーブル空間のボリュームサイズです。 テーブル空間を利用する場合、dataVolと同様に“Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”の“データベースのディスク容量の見積り”を参照し、サイズ設計を行ってください。

フィールド	省略可否	サブフィールド	省略値	説明
		storageClass	プラットフォームのデフォルトになります	ストレージクラスは開始時にのみ設定されます。
		accessModes	ReadWriteOnce	アクセスモードは開始時にのみ設定されます。
walVol	不可	size	1200Mi	トランザクションログのボリュームサイズです。 “Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”の“データベースのディスク容量の見積り”を参照し、サイズ設計を行ってください。 なお、max_wal_sizeのデフォルト値は1GBです。
		storageClass	プラットフォームのデフォルトになります	ストレージクラスは開始時にのみ設定されます。
		accessModes	ReadWriteOnce	アクセスモードは開始時にのみ設定されます。

“accessMode”は、後でpgBadgerレイヤーを含めるために組み込まれています。共有ボリューム機能を提供すると、pgBadgerコンテナが複数のサーバーインスタンス(マスター/レプリカ)からログを読み取り、Webサーバを介して公開できるようになります。

2.3.3.1 ディスク容量の管理

急激な問合せの増加などにより、データやWALの量が増加し、ディスク容量を圧迫することで、データベースの運用が停止してしまう可能性があります。ディスクの使用量が閾値を超えた場合、またはディスクの容量不足でデータベースの運用が停止した場合には、以下の方法でディスクの容量不足を解決します。

- ディスク容量の増加
- ディスク使用量の削減

2.3.3.1.1 ディスク容量の増加

ディスク容量を増やすには、以下の2つの方法があります。

- ディスク容量の拡張

  KubernetesのPVC拡張機能が使えるボリュームを使っている場合は、ディスク容量を拡張し、容量不足を解消します。

- ディスク容量が大きいデータベースクラスタへ移行

  KubernetesのPVC拡張機能が利用できないボリュームを利用している場合、または、上限等でボリュームの拡張がそれ以上できない場合は、データベースクラスタの移行を検討します。容量の大きいディスクを利用しているクラスタにデータを移行することで容量不足を解消します。

ディスク容量の拡張

KubernetesのPVC拡張機能を利用してディスク容量を拡張します。PVC拡張機能に対応しているディスクのみ容量を拡張できます。ディスクがPVC拡張機能に対応しているかどうかは、利用しているCSIドライバの仕様を確認してください。

ディスク容量の拡張は、利用者が任意のタイミングで実行する手動拡張と、オペレーターが監視機能と連携して使用量が閾値を超えたときに実行する自動拡張があります。

手動拡張は、FEPClusterカスタムリソースのストレージ定義を変更することで、PVCを拡張することができます。AlertMnagerからディスク使用量の閾値を超えた通知が出たときや、ディスク容量の不足によりデータベースが停止してしまったときに、カスタムリソースを書き換えて

PVCを拡張します。手動拡張の詳細は、“5.3.2 PVCのサイズ変更”を参照してください。また、AlertManagerのアラートルールの定義例は、“リファレンス”の“デフォルトのアラートルール”を参照してください。

自動拡張は、拡張する上限になるまでは、データベース管理者の監視および手動のメンテナンス作業（ボリュームの容量拡充）が不要です。

自動拡張の詳細は、“2.3.3.2 PVC自動拡張機能の設定”を参照してください。

PVC拡張機能に対応していないディスクを利用している場合や、ディスク容量が拡張できない場合は、“ディスク容量が大きいデータベースクラスタへ移行”と“2.3.3.1.2 ディスク使用量の削減”を参照してください。

ディスク容量が大きいデータベースクラスタへ移行

バックアップ/リストア機能を利用して、別のディスクに新規データベースクラスタを構築し、データを移行します。以下の場合に、この方法を使用してください。

- AlertMnagerからディスク使用量の閾値を超えた通知が出た場合
- ディスク容量の不足によりデータベースが停止してしまった場合

容量の大きなディスクに変更することで、ディスク容量不足を解消することができます。

新規データベースクラスタへの移行は、FEPRestoreカスタムリソースのspec.changeParamsを設定してリストア元から定義を変更することで、新規FEPClusterを構築しデータを復元できます。詳細は、“リファレンス”の“FEPRestoreカスタムリソースパラメータ”を参照してください。

2.3.3.1.2 ディスク使用量の削減

ディスク容量不足の予防保守として、データ格納先(dataVol、walVol、tablespaceVol)で、REINDEX文を実行します。詳細は、“Fujitsu Enterprise Postgres 運用ガイド”の“インデックスの再編成”を参照してください。

トランザクションログ格納先とバックアップデータ格納先の容量不足に対する予防保守として、ディスク使用量の削減を検討してください。トランザクションログ格納先の容量不足の場合、ログファイルのローテーション設定や出力レベルを見直し、変更を検討してください。バックアップデータまたはトランザクションログのアーカイブ格納先の容量不足の場合、バックアップの保存世代数の削減を検討してください。

バックアップの保存世代数は、FEPActionカスタムリソースのspec.fepAction.typeにbackup_expireを指定することで削減できます。詳細は、“リファレンス”の“FEPActionカスタムリソースパラメータ”を参照してください。

2.3.3.2 PVC自動拡張機能の設定

FEPClusterカスタムリソースのspec.fepChildCrVal.storage.autoresize.enableにtrueを設定すると、ディスク使用量が閾値を超えた場合にディスク容量を自動的に拡張するPVC自動拡張機能を有効にすることができます。

PVC自動拡張機能を有効にするためには、下記の2つの条件を満たしている必要があります。

- StorageClassにPVC拡張機能に対応しているボリュームを指定
- StorageClassのallowVolumeExpansionフィールドにtrueを指定

指定したボリュームがPVC拡張機能に対応しているかどうかは、利用しているCSIドライバの仕様を確認してください。

また、ストレージの使用状況を監視するためにPrometheusが必要です。下記のkubeletによって取得されるメトリクスをPrometheusでスクレイプしてください。

- kubelet_volume_stats_used_bytes（Volumeの使用中の容量(バイト)）
- kubelet_volume_stats_capacity_bytes（Volumeの容量(バイト)）

Prometheusの設定ファイルのscrape_configセクションで、各Nodeに対して、kubeletがhttpsで提供する/metricsを参照していることを確認してください。詳細は、Prometheusのドキュメントを確認してください。

PVC自動拡張機能は、FEPCluster構築後でも有効/無効を切り替えることができます。

PVC自動拡張機能を有効にすると、pvc-auto-resizeコンテナを含むfep-tuning podが構築されます。pvc-auto-resizeコンテナは、定義されている各PVCのメトリクスを定期的にPrometheusから取得します。PVCのボリューム使用率が、定義されている閾値を超えていた場合、自動的にFEPClusterカスタムリソースの定義を書き換えます。カスタムリソースが書き換えられたことで、対象のPVCが自動的に拡張されます。

FEPClusterが複数台構成で構築されている場合、1台でもボリュームの使用率が閾値を超えるとPVCの拡張を実行します。

PVC自動拡張機能では下記のパラメータを利用します。各パラメータの詳細は、“リファレンス”を参照してください。

- spec.fep.autoTuningセクション:メトリクスを取得するPrometheusの接続情報
- spec.fepChildCrVal.storage.autoresizeセクション:ストレージ共通の拡張設定
- spec.fepChildCrVal.storage.xxxVolセクション:各ストレージの定義と個別の拡張設定

拡張設定には、ボリューム使用率の閾値、サイズの拡張量、拡張可能なサイズの上限などを定義することが可能です。

下記にPVC自動拡張機能を有効化する場合のFEPClusterカスタムリソースの定義例を示します。

```
spec:
  fep:
    autoTuning:
      prometheus:
        prometheusUrl: http://prometheus-prometheus-oper-prometheus.prometheus.svc:9090
  fepChildCrVal:
    storage:
      autoresize:
        enable: True
        threshold: 20
        increase: 20
      dataVol:         # dataボリュームはautoresize配下の定義そのまま利用する
        size: 10Gi
        storageClass: resizable-storage
      walVol:           # walボリュームは閾値と拡張の上限を変更する
        size: 2Gi
        storageClass: resizable-storage
        threshold: 50
        storageLimit: 10
      backupVol:     # backupボリュームは拡張しない
        size: 20Gi
        storageClass: share-storage
        accessModes: ReadWriteMany
        storageLimit: 0
```

監視機能との組み合わせの考え方

PVC自動拡張機能では、拡張可能なストレージの上限を決めることができます。しかし、想定以上のデータ投入などが発生し設定した上限よりもデータ量が増えてしまう可能性があります。これを避けるために、PVC自動拡張機能と合わせて、監視機能の利用を推奨します。

FEPExporter機能利用時にデフォルトで作成されるアラートルールでは、ボリューム使用率が90%を超えたときにアラートが発信されます。PVC自動拡張機能のデフォルトの閾値は80%です。これにより、ディスクの拡張上限に達して自動拡張がされない状態のときに、ボリューム使用量が増えてしまったとしても、AlertManagerからアラートが発信されるため、ディスク容量の不足に気づくことができます。

自動拡張機能の閾値をアラートルールの閾値より低い値に設定することで、より安全なディスク容量を保つことができます。

2.3.4 Pgpool-IIのデプロイとオペレーターからのFEPClusterへの接続

Kubernetesコマンド: kubectl create FEPpgpool2

この操作により、FEP pgpool2コンテナが作成されます。

詳細については、“リファレンス”の“FEPPgpool2カスタムリソースパラメータ”を参照してください。

2.3.5 オペレーターからのバックアップのスケジュール設定

FEPClusterを作成する場合、ユーザーはバックアップ定義を設定することにより、スケジュールされたバックアップを取得できます。ユーザーは、作成されたバックアップカスタムリソースを変更することにより、バックアップスケジュールを変更することもできます。

バックアップ定義には次のものが含まれます。

- 取得時間（crontab形式で指定）
- バックアップの種類（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）

バックアップはマスターPodでのみ行われます。

バックアップ処理はpgBackRestによって実行されます。

カスタムリソース定義でパラメータをpgbackrestParamsに設定できます。

バックアップスケジュールの最大数は5です。

パラメータの詳細については、pgBackRestユーザーズガイドを参照してください。

ただし、一部のパラメータは制限されています。詳細は以下のとおりです。

2.3.5.1 重要な設定項目

pgBackRestを設定するための重要なパラメータは次のとおりです。このパラメータは、バックアップ情報の保持期間を設定します。自動バックアップが設定されていて、このパラメータが設定されていない場合、バックアップ領域がオーバーフローするリスクが高まります。

パラメータ	説明	設定値
Full Retention Option (repo retention -full)	保持するフルバックアップの値を指定します。 デフォルト値なし(ユーザーのバックアップポリシーに従って設定する必要があります)	正の整数値
Full Retention Type Option (repo retention-full-type)	spec.retention -full設定が保持日数(time)であるか保持世代数(count)であるかを指定します。 デフォルト値なし(ユーザーのバックアップポリシーに従って設定する必要があります)	time/count

以下は、上記のパラメータを設定して、バックアップ保持期間(PITRが有効な期間)をFEPClusterのデプロイ後30日に変更するカスタムリソースの例です。

```
apiVersion: fep.fujitsu.io/v1
kind: FEPBackup
metadata:
  name: fepcluster-backup
spec:
  pgBackrestParams: |
    # define custom pgbackrest.conf parameters below to override defaults.
    [global]
    repo-retention-full = 30
    repo-retention-full-type = time
...
```

2.3.5.2 設定できないパラメータ

pgBackRest構成ファイルの以下のパラメータは設定できません。

パラメータ	説明	理由
Copy Archive Option (--archive -copy)	一貫性を保つために必要なWALセグメントをバックアップにコピー	内部固定値を使用

パラメータ	説明	理由
Check Archive Mode Option (--archive-mode-check)	PostgreSQLのarchive_modeの設定を確認	マスターからのバックアップに限定
Backup from Standby Option (--backup-standby)	スタンバイクラスタからのバックアップ	マスターからのバックアップに限定
Stop Auto Option (--stop-auto)	以前に失敗したバックアップを新しいバックアップで停止	9.6では未サポート
pgBackRest Command Option (--cmd)	pgBackRestコマンド	内部固定値を使用
SSH client command Option (--cmd-ssh)	sshクライアント実行可能ファイルへのパス	sshは未使用
Compress Option (--compress)	ファイル圧縮を使用	廃止されたオプション（代わりにcompresse-typeオプションを使用してください）
Config Option (--config)	pgBackRest構成ファイル	内部固定値を使用
Config Include Path Option (--config-include-path)	追加のpgBackRest設定ファイルへのパス	内部固定値を使用
Config Path Option (--config-path)	pgBackRest構成ファイルのベースパス	内部固定値を使用
Delta Option (--delta)	チェックサムを使用したリストアまたはバックアップ	新規リストアのみ
Dry Run Option (--dry-run)	command.lesのドライランを実行	コマンドラインのみのオプション
Lock Path Option (--lock-path)	ロックファイルが保存されているパス	内部固定値を使用
Keep Alive Option (--sck -keep-alive)	ソケット接続でキープアライブメッセージを有効にする	内部固定値を使用
Spool Path Option (--spool-path)	非同期archive-pushコマンドおよびarchive-getコマンドの一時データを格納するパス	FEPClusterカスタムリソース値からの自動判別用
Stanza Option (--stanza)	スタンザを定義	内部固定値を使用
Console Log Level Option (--log-level-console)	コンソールログレベル	Podでの動作は想定外
Std Error Log Level Option (--log-level-stderr)	標準エラーログレベル	Podでの動作は想定外
Log Path Option (--log-path)	ログファイルの保存先	FEPClusterカスタムリソース値からの自動判別用
Repository Host Option (--repo-host)	SSH経由のリモート操作用のリポジトリホスト	リポジトリホストは未使用
Repository Host Command Option (--repo-host-cmd)	リポジトリホスト上のpgBackRestのパス	
Repository Host Configuration Option (--repo-host-config)	リポジトリホスト構成ファイルのパス	
Repository Host Configuration Include Path Option (--repo-host-config-include-path)	インクルードパスを構成するリポジトリホスト	
Repository Host Configuration Path Option (--repo-host-config-path)	リポジトリホスト構成パス	

パラメータ	説明	理由
Repository Host Port Option (--repo-host-port)	“repo-host”が構成されている場合のリポジトリホストポート	
Repository Host User Option (--repo-host-user)	“repo-host”が構成されている場合のリポジトリホストユーザー	
Repository Path Option (--repo-path)	バックアップとアーカイブが保存されるパス	FEPClusterカスタムリソース値からの自動判別用
Archive Retention Option (--repo-retention-archive)	保持する連続したWALバックアップの数	このオプションは推奨されません。WALの保持は、Full Retention OptionとFull Retention Type Optionによって制御されます。
Archive Retention Type Option (--repo-retention-archive-type)	WAL保持のバックアップタイプ	デフォルトから変更しないことを推奨
Differential Retention Option (--repo-retention-diff)	保持する増分バックアップの数	増分バックアップなし
Archive Mode Option (--archive-mode)	リストアされたクラスタのアーカイブを保持または無効にする	内部固定値を使用
Exclude Database Option (--db-exclude)	指定されたデータベースを除いてリストア	すべてのデータベースを含むFEPクラスタ全体をリストアするため
Include Database Option (--db-include)	指定されたデータベースのみをリストア	すべてのデータベースを含むFEPクラスタ全体をリストアするため
Link All Option (--link-all)	すべてのシンボリックリンクを復元	内部固定値を使用
Link Map Option (--link-map)	シンボリックリンクのリンク先を変更	内部固定値を使用
Recovery Option Option (--recovery-option)	postgreSQLのrecovery.confでのオプションの設定	内部固定値を使用
Tablespace Map Option (--tablespace-map)	テーブルスペースを指定されたディレクトリにリストア	FEPClusterカスタムリソース値からの自動判別用
Map All Tablespaces Option (--tablespace-map-all)	すべてのテーブルスペースを指定されたディレクトリにリストア	FEPClusterごとにテーブルスペースが1つしかないため、テーブルスペースは不要
TLS Server Address Option (--tls-server-address)	TLSサーバのアドレス	TLSサーバは未使用
TLS Server Authorized Clients Option (--tls-server-auth)	TLSサーバ承認済みクライアント	
TLS Server Certificate Authorities Option (--tls-server-ca-file)	TLSサーバ認証局	
TLS Server Certificate Option (--tls-server-cert-file)	TLSサーバ証明書ファイル	
TLS Server Key Option (--tls-server-key-file)	TLSサーバのファイル	
TLS Server Port Option (--tls-server-port)	TLSサーバポート	
PostgreSQL Database Option (--pg-database)	PostgreSQLデータベース	内部固定値を使用

パラメータ	説明	理由
PostgreSQL Host Option (--pg-host)	SSH経由のリモート操作用のPostgreSQLホスト	SSH接続は不要
PostgreSQL Host Command Option (--pg-host-cmd)	PostgreSQLホスト上のpgBackRestexeのパス	内部固定値を使用
PostgreSQL Host Configuration Option (--pg-host-config)	pgBackRest構成ファイルのパス	内部固定値を使用
PostgreSQL Host Configuration Include Path Option (--pg-host-config-include-path)	PostgreSQLホストインクルードパスでのpgBackRestの設定	内部固定値を使用
PostgreSQL Host Configuration Path Option (--pg-host-config-path)	PostgreSQLホスト上でpgBackRestを設定するためのパス	内部固定値を使用
PostgreSQL Host Port Option (--pg-host-port)	SSHポート仕様	SSH接続は不要
PostgreSQL Host User Option (--pg-host-user)	pg-hostが設定されている場合、PostgreSQLをホストするときのログオンユーザー	SSH接続は不要
PostgreSQL Path Option (--pg-path)	PostgreSQLデータディレクトリ	FEPClusterカスタムリソース値からの自動判別用
PostgreSQL Port Option (--pg-port)	PostgreSQLポート	FEPClusterカスタムリソース値からの自動判別用
PostgreSQL Socket Path Option (--pg-socket-path)	PostgreSQL Unixソケットパス	FEPClusterカスタムリソース値からの自動判別用
PostgreSQL Database User Option (--pg-user)	PostgreSQLデータベースユーザー	内部固定値を使用

2.3.5.3 制限されているパラメータ

pgBackRest構成リファレンスのパラメータのうち、次のパラメータは設定可能な値を制限しています。

パラメータ	説明	設定可能な値
repoX-gcs-key-type	Google Cloud Storageを使用する際に指定する鍵ファイルのタイプ	service

2.3.5.4 設定ファイルのセクション

FEPBackupカスタムリソースでは、pgbackrest.confの内容を書き込むことができますが、スタンザ（pgBackRestのバックアップスペース）の設定は内部で指定されます。

次のセクションは許可されていません。

[stanza: command] , [stanza]

2.3.6 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、バックアップデータを既存のFEPClusterにリストアする方法と、新しいFEPClusterを作成してバックアップデータをリストアする方法があります。

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

リストアプロセスでは、FEPリストアコンテナをデプロイします。FEPリストアコンテナは、FEPClusterのマスターサーバにリストアされるバックアップデータからpgBackRestリストア操作を実行します。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、FEPClusterが作成されます。

ユーザーが新しいFEPClusterを作成すると、特に指定しない限り、新しく作成されたFEPClusterはソースクラスタの設定を継承します。ただし、ユーザーはFEPRestoreカスタムリソースに設定を含めることで、ソースクラスタとは異なる設定でクラスタを作成することもできます。

新しいクラスタへの接続の切り替え

リストアにより、新しいFEPClusterが作成されます。必要に応じて、Pgpool-IIをセットアップし、アプリケーションのアクセスポイントを新しいクラスタまたは新しいPgpool-IIに変更する必要があります。

FEPClusterのリカバリ

既存のFEPClusterに障害が発生し、FEPが実行されていない場合でも、バックアップ領域のボリュームが安全であれば、バックアップデータからリカバリできます。

2.3.7 デフォルトで有効なFEP機能

デフォルトでは、以下のFEP機能が有効です。

- データ秘匿化
- 透過的データ暗号化(TDE)

データ秘匿化

FEPClster カスタムリソースの例では、データ秘匿化がデフォルトで有効になっています（Openshift UIの場合）。 コンテナ内のpostgresql.confには、次のパラメータが含まれています。

```
shared_preload_libraries = 'pgx_datamasking,pg_prewarm'
session_preload_libraries = 'pg_prewarm'
max_worker_processes= 20
```

ユーザーは、config mapのこれらの値を上書きできます。

透過的データ暗号化(TDE)

透過的データ暗号化(TDE)はデフォルトで有効になっています。透過的データ暗号化で用いるマスタ暗号化キーを保管するキーストアとして以下のいずれかを選択します。

- ファイルベースのキーストア
- 外部の鍵管理サービス

キーストアとして鍵管理サービスを利用する場合、FEPクラスタのデプロイ後でも、他の鍵管理サービスにキーストアを変更できます。ファイルベースのキーストアから鍵管理サービスへの変更、鍵管理サービスからファイルベースのキーストアへの変更はどちらもできません。

鍵管理システムを使用する場合の設計観点は、“2.3.11 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。

2.3.8 モニタリングおよびアラート(FEPExporter)

オペレーターはFEPコンテナに関するさまざまなメトリクスを公開します。

FEPは、さまざまなビューを介して多くの有用なデータベース統計を生成します。デフォルトの統計情報は、pg_stat_statementsのようなエクステンションを使用することでさらに拡張することができます

FEPExporterコンテナは、デフォルトで、有用なデータベース統計を抽出し、ププラットフォーム上のPrometheusがメトリクスを使用できるように設定されています。外部コンポーネントとユーティリティを使用して、表示されたメトリクスに基づいて、視覚化、分析、アラートのトリガー、および運用上の決定を行うことができます。

FEPExporterは、FEPクラスタのアクティブなモニタリングに役立つPrometheusメトリクスに基づいてデフォルトのアラートルールも設定します。

2.3.8.1 FEPExporterカスタムリソース

FEPExporterカスタムリソースのパラメータについては、“リファレンス”の“FEPExporterカスタムリソース”を参照してください。

- メトリクスをスクレイピングするカスタムクエリは、カスタムリソースのオプションセクションに追加できます。

- Prometheusのカスタムアラートルールは、ユーザーが手動で作成します。

2.3.8.2 FEPClusterへの変更-メトリクスユーザー

ユーザーは、対象のFEPClusterにpgMetricsUser、pgMetricsPassword、およびpgMetricsUserTlsを定義できます。定義されている場合、FEPExporterはメトリクスユーザーの詳細を使用してFEPクラスタマシンに接続します。すべてのメトリクスユーザーフィールドはオプションであり、FEPClusterでは省略できます。

FEPClusterのパラメータについては、“リファレンス”の“FEPClusterパラメータ”を参照してください。

2.3.8.3 FEPClusterのFEPExporterカスタムリソースの自動作成

ユーザーは、FEPClusterをモニタリングするために、FEPClusterカスタムリソースの一部としてenableMonitoringフラグを定義できます。FEPCluster固有のFEPExporterが自動的に作成されるため、FEPClusterのマトリックススクレイピングが機能します。

FEPClusterのパラメータについては、“リファレンス”の“FEPClusterパラメータ”を参照してください。

- FEPExporterは、<cluster-name> -fepexporterという名前になります。
- FEPExporterが自動的に作成されると、ユーザーはFEPExporterカスタムリソースから手動で変更できます。
- FEPClusterが削除されると、依存するFEPExporterも削除されます。
- FEPExporterのMTLSは、PrometheusとFEPExporterの両方の仕様でtls構成が定義されている場合にのみサポートされます。

2.3.9 レプリカの拡張

データベースの平均CPU使用率またはコネクション数がしきい値を超えると、自動スケールアウトが発生します。データベースのボトルネックとなっているリソースに応じて、自動スケールアウトの基準をCPU使用率とするか、コネクション数とするかを選択してください。

マスターコンテナを除くレプリカコンテナの最大数は15です。

CPU使用率に基づくスケールアウト

FEPClusterに含まれるすべてのPod(プライマリPodとすべてのレプリカPod)のCPU使用率の平均が、一定の期間、しきい値を超えていた場合にスケールアウトを実行します。

CPU使用率は、FEPClusterカスタムリソースに指定する、spec.fep.mcSpec.requests.cpu に指定する値を分母として計算されます。

コネクション数に基づくスケールアウト

FEPClusterに含まれるすべてのPod(プライマリPodとすべてのレプリカPod)のコネクション数の平均が、一定の期間、しきい値を超えていた場合にスケールアウトを実行します。

自動スケールアウトを実行するコネクション数のしきい値を、FEPサーバのmax_connectionsパラメータ以下の値で指定します。

コネクション数に基づくスケールアウト機能を利用する場合の前提条件は、以下のとおりです。

- モニタリング機能(“2.3.8 モニタリングおよびアラート(FEPExporter)”参照)が有効になっている
- モニタリング機能によってFEPサーバのコネクション数のメトリクスが収集されている
- OCP/Kubernetesクラスタにカスタムメトリクスサーバが導入されている
- カスタムメトリクスサーバが、モニタリング機能によって収集されたコネクション数の平均値を公開している

コネクション数に基づくスケールアウト機能を使用する場合、自動スケールアウト機能はカスタムメトリクスサーバに、以下のKubernetesリソースに紐づいたメトリクスを要求します。

- kind: FEPCluster
- apiVersion: fep.fujitsu.io/v2
- name: FEPClusterの名称
- namespace: FEPClusterを配備した名前空間の名称

要求するメトリクスの名称は、metricNameパラメタに指定した名前です。

このメトリクスは、指定されたFEPClusterの各Podのコネクション数の平均値の数値を表現している必要があります。

制限事項

- コネクション数に基づくスケールアウト機能を利用する場合は、FEPExporterは“4.3 FEPExporterのデプロイ”の手順で配備してください。
- FEPClusterのメトリクスがスタンドアロンモードのFEPExporter(“4.4 スタンドアロンモードでのFEPExporter”参照)で収集される場合、コネクション数に基づくスケールアウト機能を利用できません。

注意

自動スケールアウト機能を使用する場合、FEPCluster同期モードは“off”である必要があります。

自動スケールアウトの設計時の注意事項

- 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。

2.3.9.1 FEPClusterカスタムリソースへの変更-自動スケールアウト

自動スケールアウトを使用する場合は、パラメータをFEPClusterカスタムリソースに設定します。

FEPClusterのパラメータについては、“リファレンス”の“FEPClusterパラメータ”を参照してください。

2.3.10 ディザスタリカバリ

オブジェクトストレージを利用することで異なるコンテナ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コンテナ環境での運用が困難な場合でも、異なるコンテナ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としてはバックアップ/リストア方式およびホットスタンバイ方式があります。

バックアップ/リストア方式

災害発生後に新規コンテナ環境を構築し(コールドスタンバイ)、オブジェクトストレージからデータをリストアします。この方式では後述の方式と比較して、2つのコンテナ環境を構築しておく必要がない分、コストを抑えることが可能ですが、新規コンテナ環境のFEPClusterを配備し、その後にデータの復元処理を実行するため、復旧時間を要します。

ホットスタンバイ方式

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稼働させます。ホットスタンバイ構成でのディザスタリカバリを実現することで、災害発生時により迅速に業務システムを復旧することができます。

また、Veleroを利用してKubernetes環境にあるシステムをオブジェクトストレージにバックアップすることで、異なるKubernetesへシステムをリストアするディザスタリカバリが実現できます。

2.3.11 鍵管理システムを利用した透過的データ暗号化

Fujitsu Enterprise Postgresは、PostgreSQLのセキュリティを強化する独自の機能を提供します。これらのセキュリティ機能により、ユーザーはデータへの不正アクセスからデータの安全性を確保できます。このようなセキュリティ機能の1つに透過的データ暗号化(TDE)があり、保存データ、つまりディスク/永続ボリュームに保存されたデータが暗号化されます。

一方、TDEのデフォルト形式では、マスター暗号化キーはパスワードで保護されたファイルに格納されます。鍵管理システムを使用すると、クラウド・ベースのキーストアにマスター暗号化キー(MEK)を格納することができ、セキュリティを次のレベルに引き上げることができます。

透過的データ暗号化で利用できる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(x86のみ)
- Azureの鍵管理サービス(x86のみ)

鍵管理システムの詳細な要件は、“付録D 透過的データ暗号化で利用できる鍵管理システム”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は、FEPClusterを最初に作成したときのみ構成できます。ユーザーは鍵管理システムを利用した透過的データ暗号化を既存のFEPClusterに設定することはできません。

鍵管理システム上のマスタ暗号化キーが失われた場合、暗号化されたデータ/バックアップデータを復号することはできません。マスタ暗号化キーで暗号化されたデータが有効である限り、マスタ暗号化キーも利用可能であるよう鍵管理システム上で管理する必要があります。

鍵のローテーションによってマスタ暗号化キーが更新されたとしても、古い暗号化キーでバックアップデータを暗号化している場合、古い暗号化キーも保持し続ける必要があります。古い暗号化キーが保持されない場合、バックアップから復元されたデータベースを開くことができなくなります。

また、鍵の管理者は古いマスタ暗号化キーで暗号化されたデータが有効な期間、参照されているマスタ暗号化キーを保持し続ける必要があります。

2.3.12 データベースロール管理

オペレーターでは、データのアクセス制御を管理するために、データベース運用にかかわるロールの作成と権限の割り当てや、ログイン権限を持つデータベースロールの有効期限の管理を容易に実現することができます。

データベースには個人情報などの重要なデータが含まれており、データの保護が重要です。

データの保護は、セキュリティ規約で定められており、運用において大事な観点です。

データの保護において、第三者にデータを参照されないようにするため、データベースロールのアクセス制御が適切に設定されていることが必要です。

本機能では、下記のデータベースロールに分けることを推奨しています。

- データベース管理者:データベースシステムの構築/運用
- 機密管理者:各データベース資源に対する適切な権限を設定
- 一般利用者:データベースのエンドユーザー

データベースの運用/管理者を複数用意し、それぞれに権限を割り当てることで、権限の分散を図ることができます。これにより、強い権限を持つユーザーによる、データの参照や改ざんなどを防ぐことが可能になります。

本機能で作成するデータベースロールの役割について説明します。

データベース管理者は、システムテーブルの参照/バックエンドの問い合わせのキャンセルなど、データベースの運用にかかわる操作を実行することができます。

機密管理者は、テーブルやロールに対して適切な権限を付与し、第三者によるデータの参照を防ぎます。本機能では機密管理者が“機密管理支援機能”を利用可能なように権限を付与し、各データベース資源に対する適切な権限を付与することが可能になります。

また、データの保護において、すべてのデータを参照できるSUPERUSERやBYPASSRLS権限が付与されたロールを利用することは推奨されません。そのため、本機能では、SUPERUSER(postgres)のパスワードを隠蔽することで隔離し、作成するデータベースロールにSUPERUSERやBYPASSRLS権限を付与しません。

2.3.12.1 データベース運用にかかわるロールの作成

2.3.12.1.1 SUPERUSERの隔離

データベース構築時にオペレーターがSUPERUSER権限を持つデータベースロール“postgres”を作成します。

FEPClusterカスタムリソースの“spec.fepChildCrVal.sysUsers.pgAdminPassword”を省略することで、postgresロールのパスワードはランダムな値で作成され、一般ユーザーがSUPERUSER権限を利用できなくなります。ただし、データベース運用において管理者がSUPERUSER権限を必要としたときに“postgres”ロールを利用する手段を別途提供するため、想定外の利用がされていないかpgAuditの監査機能を利用して監視してください。

2.3.12.1.2 データベース管理者ロール

データベースを管理するためのデータベースロールです。このロールの定義は必須です。

ユーザー名とパスワードは、FEPClusterカスタムリソースのspec.fepChildCrVal.sysUsers配下の“pguser”と“pgpassword”に定義します。CREATE DATABASE権限を持ち、システムテーブルの参照/バックエンドの問い合わせのキャンセルを実行することができます。

データベース管理者ロールは下記の権限を保持します。

- NOSUPERUSER
- NOREPLICATION
- NOBYPASSRLS
- CREATEDB
- INHERIT
- LOGIN
- CREATEROLE

ただし、機密管理者ロールが作成された時は、NOCREATEROLEの権限が付与されます。

また、下記のロールに属しています。

- pg_monitor
- pg_signal_backend

2.3.12.1.3 機密管理者ロール

機密管理支援機能を利用して、データベースの利用者に対して、各データベース資源に対する適切な権限を設定するデータベースロールです。このロールの作成は任意です。

ユーザー名とパスワードは、FEPClusterカスタムリソースの“spec.fepChildCrVal.sysUsers”配下の“pgSsecurityUser”と“pgSsecurityPassword”に定義します。

機密管理者ロールは、FEPCluster構築した後から定義することが可能です。ただし、このロールを定義した後にロール名の変更とロールの削除はできません。

このロールは下記の権限を保持します。

- LOGIN
- CREATEROLE
- NOSUPERUSER
- NOREPLICATION
- NOBYPASSRLS
- NOCREATEDB
- NOINHERIT

機密管理者ロールはFEPClusterカスタムリソースの“spec.fepChildCrVal.sysUsers.pgdb”で定義されているデータベースに対してALL権限を与えられており、対象のデータベースに対してテーブルなどのデータベースオブジェクトを作成することができます。

機密管理者ロールは、Fujitsu Enterprise Postgresの「機密管理支援機能」を操作するために必要な下記の権限が割り当てられるため、ロールが作成された直後から機密管理支援機能を利用できます。

- CREATEROLE権限
- 機密管理支援機能の拡張に含まれるすべてのテーブルへのSELECT権限、INSERT権限、UPDATE権限、および、DELETE権限

機密管理支援機能で管理するデータベースオブジェクトは機密管理者ロールに所有権を付与してください。

また、機密管理者ロールが機密管理支援機能を操作するために必要な権限を他のデータベースロールに付与することで機密管理を行うユーザーを増やし、権限を分散することができます。

2.3.12.2 ログイン権限を持つデータベースロールの有効期限管理

ログイン権限を持つデータベースロールのパスワード有効期限を管理することができます。

CREATE ROLE文、またはALTER ROLE文でログイン権限を持つデータベースロールのパスワードを定義するときに、指定した期間内の有効期限を指定することを強制することが可能です。

この機能を有効にするためにFEPClusterカスタムリソースに下記のパラメータを指定します。

- spec.fepChildCrVal.customPgParamsのshared_preload_librariesに“fsep_operator_security”を指定
- spec.fepChildCrVal.sysUsers.passwordValid.daysにパスワードを変更したタイミングから指定可能な有効期限の日数を指定

FEPClusterカスタムリソースの定義例

```
fepChildCrVal:
  customPgParams: |
    shared_preload_libraries='pgx_datamasking,pg_prewarm,pg_stat_statements,fsep_operator_security'
    …
  sysUsers:
    passwordValid:
      days: 30
    pgdb: mydb
    pgpassword: mydbpassword
    pguser: mydbuser
```

この機能が有効な時には、FEPClusterカスタムリソースのspec.fepChildCrVal.sysUsersで定義される、pgpasswordとpgSecurityPasswordのパスワード有効期限は、パスワードが変更されたときからspec.fepChildCrVal.sysUsers.passwordValid.daysで指定した期間後に定義されます。

ただし、pgAdminPassword、pgreplpassword、pgRewindUserPassword、pgMetricsUserPasswordで定義されるパスワードはデータベースの運用管理をするために必要なデータベースロールのパスワードであるため、有効期限は管理しません。

また、CREATE ROLE文、またはALTER ROLE文でログイン権限を持つデータベースロールのパスワードを定義/変更するときに、対象のデータベースロールのパスワードの有効期限が指定されていない、または、spec.fepChildCrVal.sysUsers.passwordValid.daysで指定された期間より長い場合は、実行したSQLが失敗します。

spec.fepChildCrVal.sysUsers.passwordValid.daysは、FEPClusterを構築したあとに、定義や変更をすることができます。このパラメータを変更すると、有効期限が切れていないすべての管理しているデータベースロールのパスワード有効期限が更新されます。

spec.fepChildCrVal.sysUsers.passwordValid.daysを削除、または0にするとパスワードの有効期限管理は停止します。

FEPExporterカスタムリソースの機能を利用して、データベースロールのパスワード有効期限の監視や有効期限が近いまたは過ぎたデータベースロールがあるときにAlertManagerを利用してアラートを通知することが可能です。

第3章 オペレーターのインストール

本章では、オペレーターをインストールする方法について説明します。

インストールされたオペレーターコンテナに割り当てられるリソースやその変更方法については、“6.5 オペレーターコンテナの割り当てリソース”を参照してください。

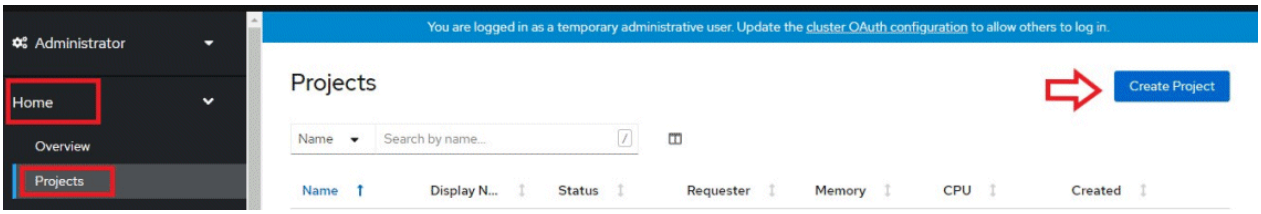
3.1 OperatorHubを使用する場合

OperatorHubを使用して、オペレーターをOpenshift上の新規の名前空間にインストールする方法を説明します。

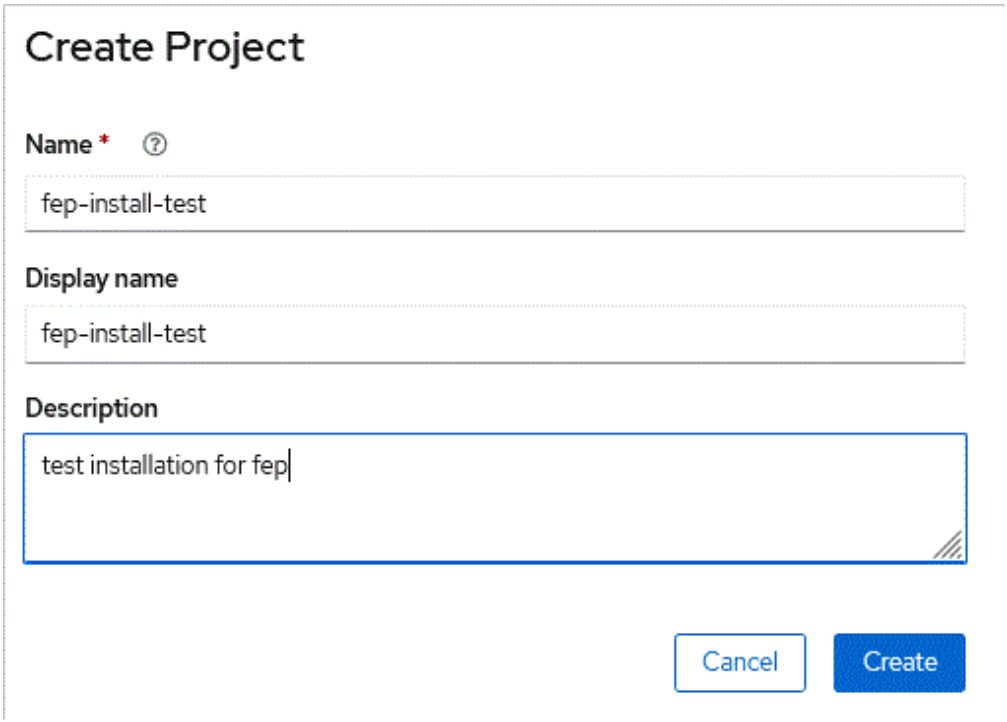
3.1.1 前提条件

Openshiftのプロジェクトは、基本的に名前空間です。オペレーターは別の名前空間にインストールすることをお勧めします。

RedHat OpenShiftプラットフォームで、[Projects]メインメニューの[Home]をクリックし、[Create Project]をクリックします。



以下のダイアログボックスで、名前空間での一意の名前、およびオプションの表示名と説明を指定します。



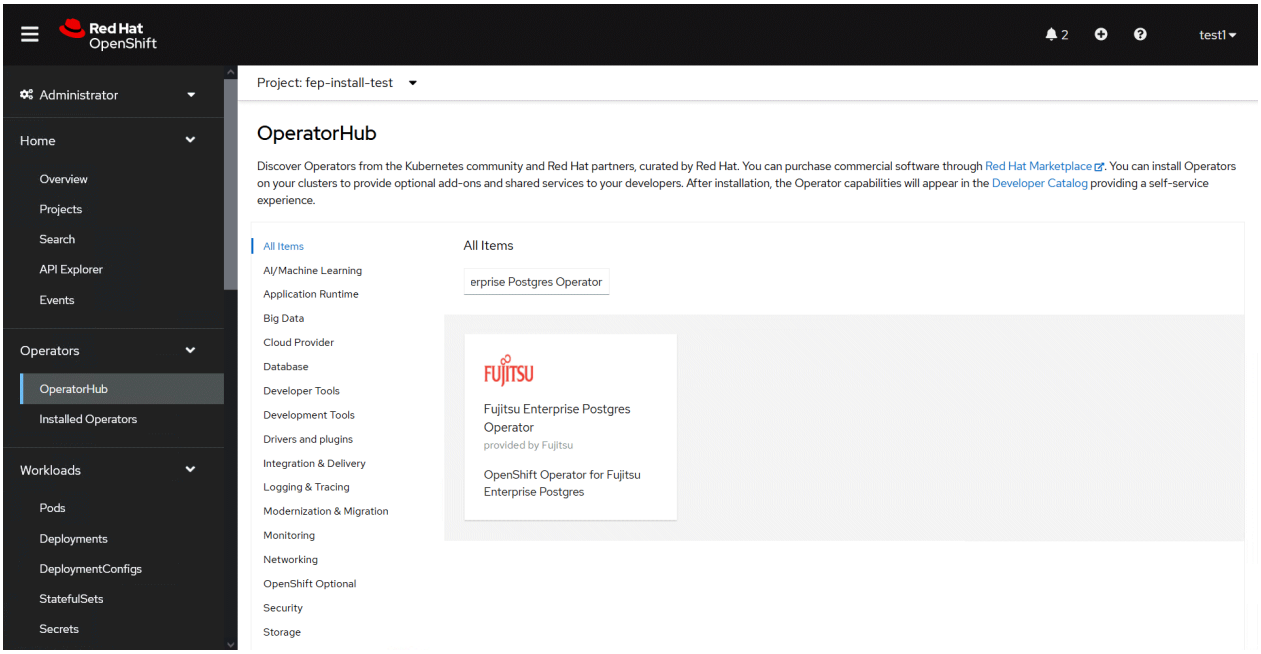
注意

オペレーターのインストールでは、PrometheusをOpenshiftクラスタにプリインストールする必要があります。

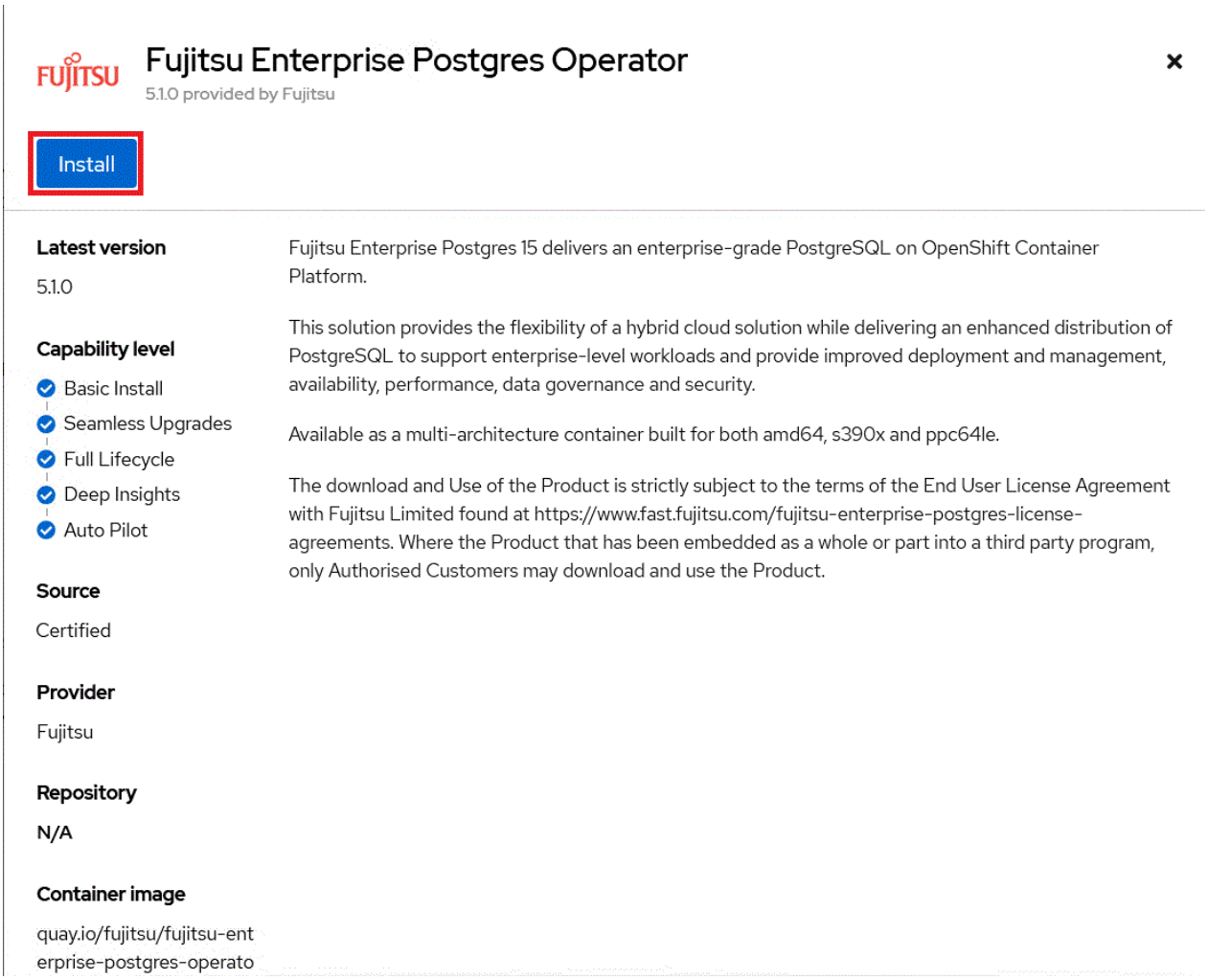
3.1.2 オペレーターのデプロイ

RedHatによって認定されたオペレーターは、すべてのRedHat OpenShiftコンテナプラットフォームのOperatorHubで利用できるようになります。

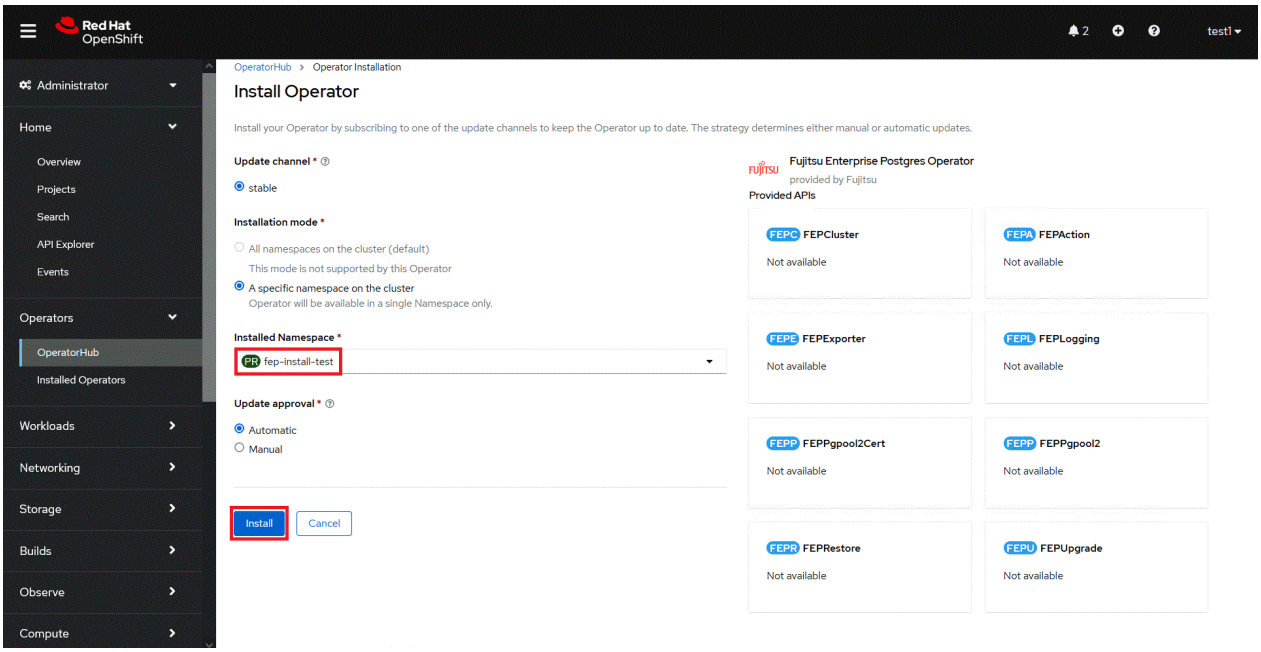
OpenShiftプラットフォームで、オペレーターをインストールする権限を持つクレデンシャルを使用してログオンします。Operators配下のメニュー項目で[OperatorHub]をクリックし、フィルターキーワード[Fujitsu Enterprise Postgres Operator]を入力して、Fujitsu Enterprise Postgres Operatorを検索します。



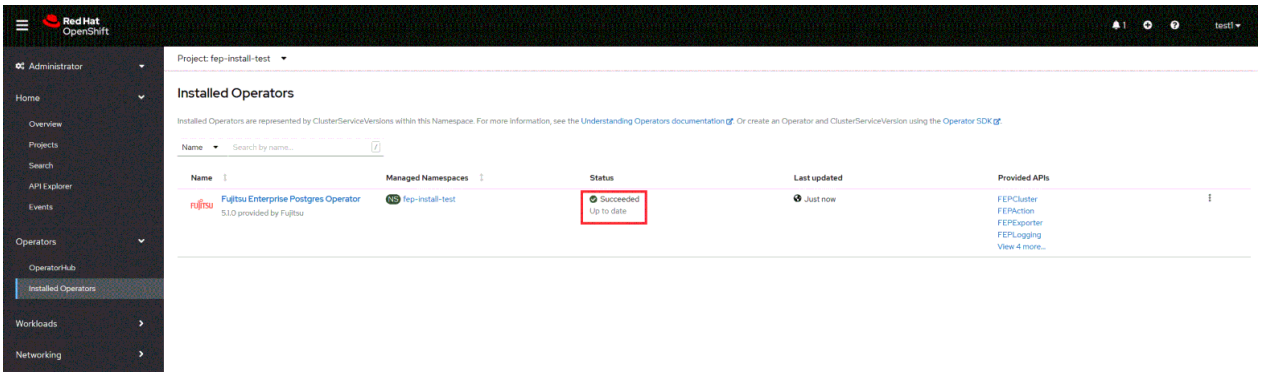
[Fujitsu Enterprise Postgres Operator]をクリックして、オペレーターをインストールします。以下のような、[Install]ボタンのある詳細ページが表示されます。



[Install]ボタンをクリックして、次の画面を表示し、名前空間と承認方法を選択します。[A specific namespace on the cluster]を選択し、目的の名前空間を選択します。その他はすべてデフォルトのままにして、[Install]をクリックします。



インストールが完了し、ステータスが[Succeeded]に変わるまで待ちます。



3.2 Helm Chartを利用する場合

Helm機能を用いて、オペレーターをKubernetes上の新規の名前空間にインストールする方法を説明します。

3.2.1 オペレーターのデプロイ

1. オペレーター用のHelm Chartリポジトリを追加します。

```
helm repo add fep-repo https://fujitsu.github.io/fep-operator-helm/v1
```

2. オペレーターをインストールする名前空間を作成します。

```
kubectl create namespace fep-operator
```

注意

オペレーターのインストールでは、事前にPrometheusをKubernetesクラスタにインストールする必要があります。

3. helmコマンドを実行し、オペレーターをインストールします。

```
helm install fep-operator-release fep-repo/fujitsu-enterprise-postgres-operator --namespace fep-operator
```

3.2.2 オペレーターのアップグレード

1. Helm Chartリポジトリの情報を最新に更新します。

```
helm repo update
```

2. 最新のオペレーターのHelm Chartバージョンを確認します。

```
helm search repo fujitsu-enterprise-postgres-operator
```

3. helmコマンドを実行し、オペレーターをアップグレードします。

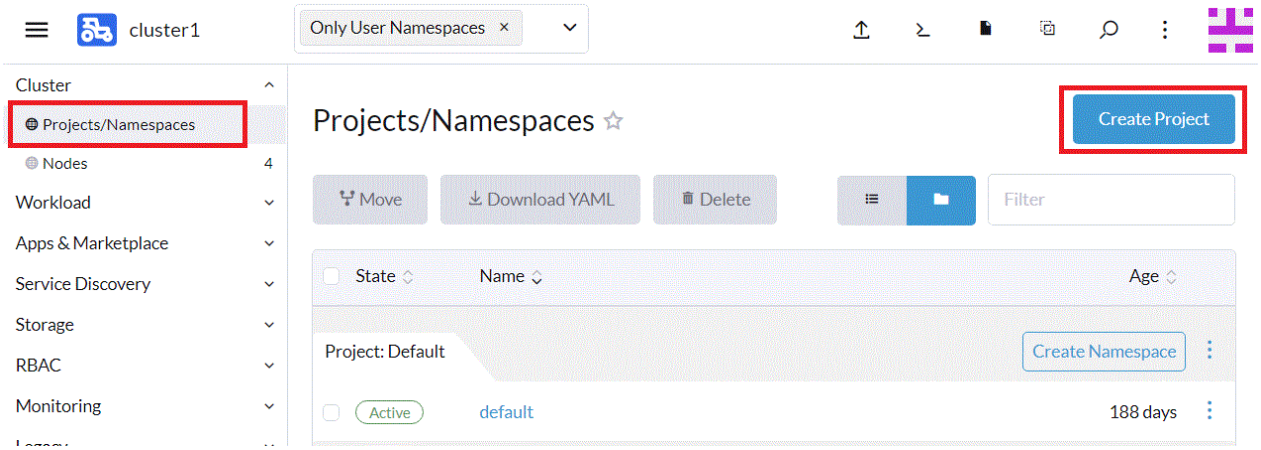
```
helm upgrade fep-operator-release fep-repo/fujitsu-enterprise-postgres-operator --namespace fep-operator
```

3.3 Rancher UIを利用する場合

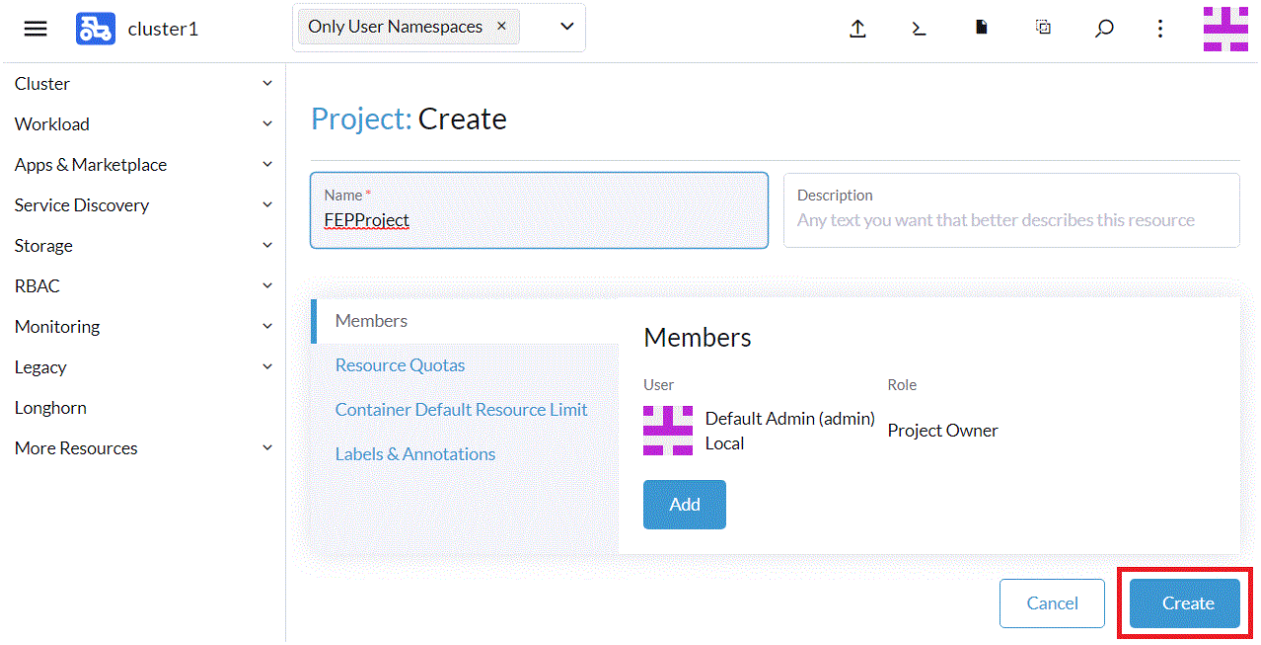
Rancher UI上で、オペレーターをKubernetes上の新規の名前空間にインストールする方法を説明します。

3.3.1 前提条件

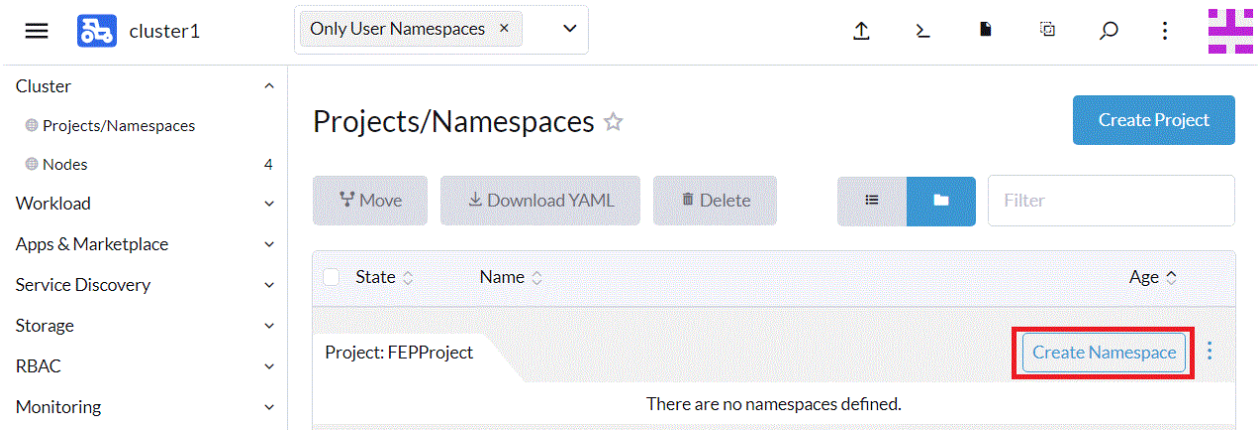
Rancher UI上でプロジェクトと、それに紐づく名前空間を作成します。FEPを別の名前空間にインストールすることをお勧めします。Rancher UIで、[Projects/Namespaces]をクリックし、表示された[Create Project]をクリックします。



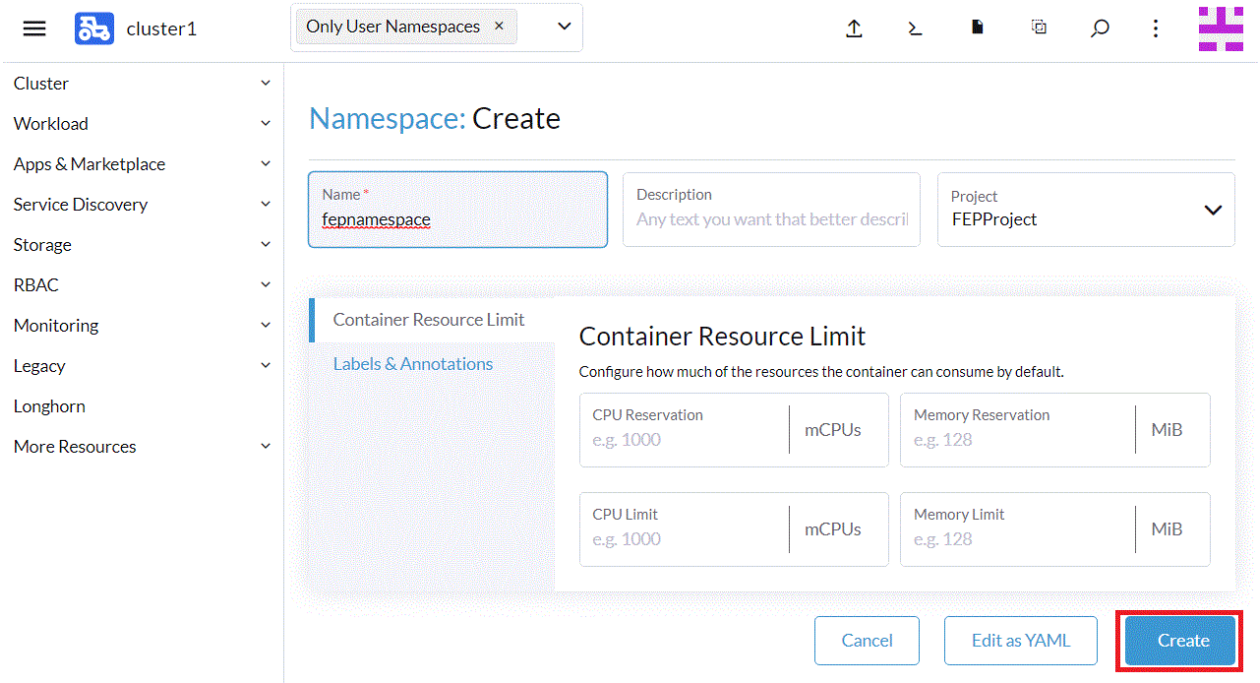
プロジェクトの一意の名前を指定し、[Create]をクリックします。



指定したプロジェクト上に表示されている[Create Namespace]をクリックします。



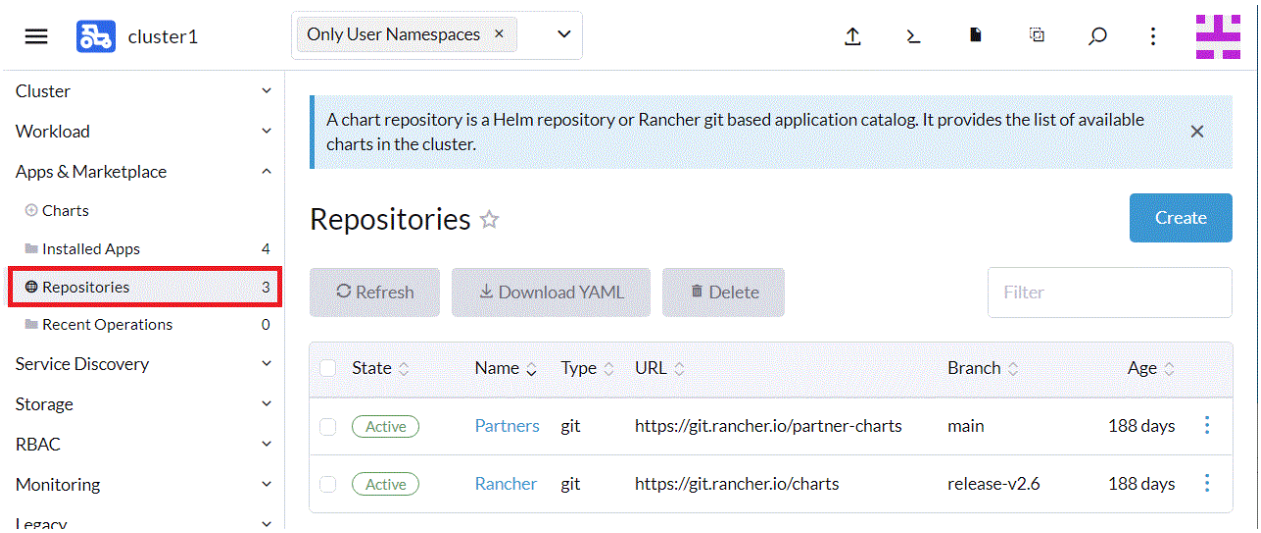
名前空間の一意の名前を指定し、[Create]をクリックします。



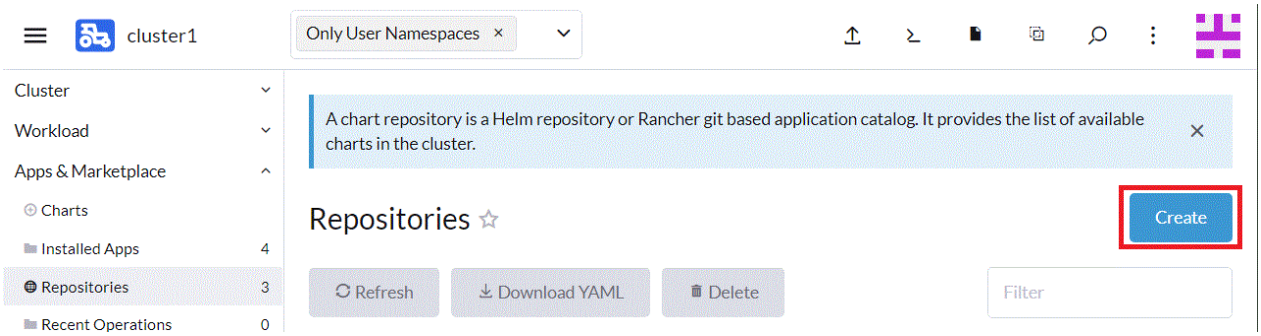
3.3.2 Helm Chartリポジトリの登録

Rancher UI上で、オペレーター機能のHelm Chartリポジトリを登録します。

Rancher UIで、[Apps & Marketplace]をクリックし、表示された[Repositories]をクリックします。

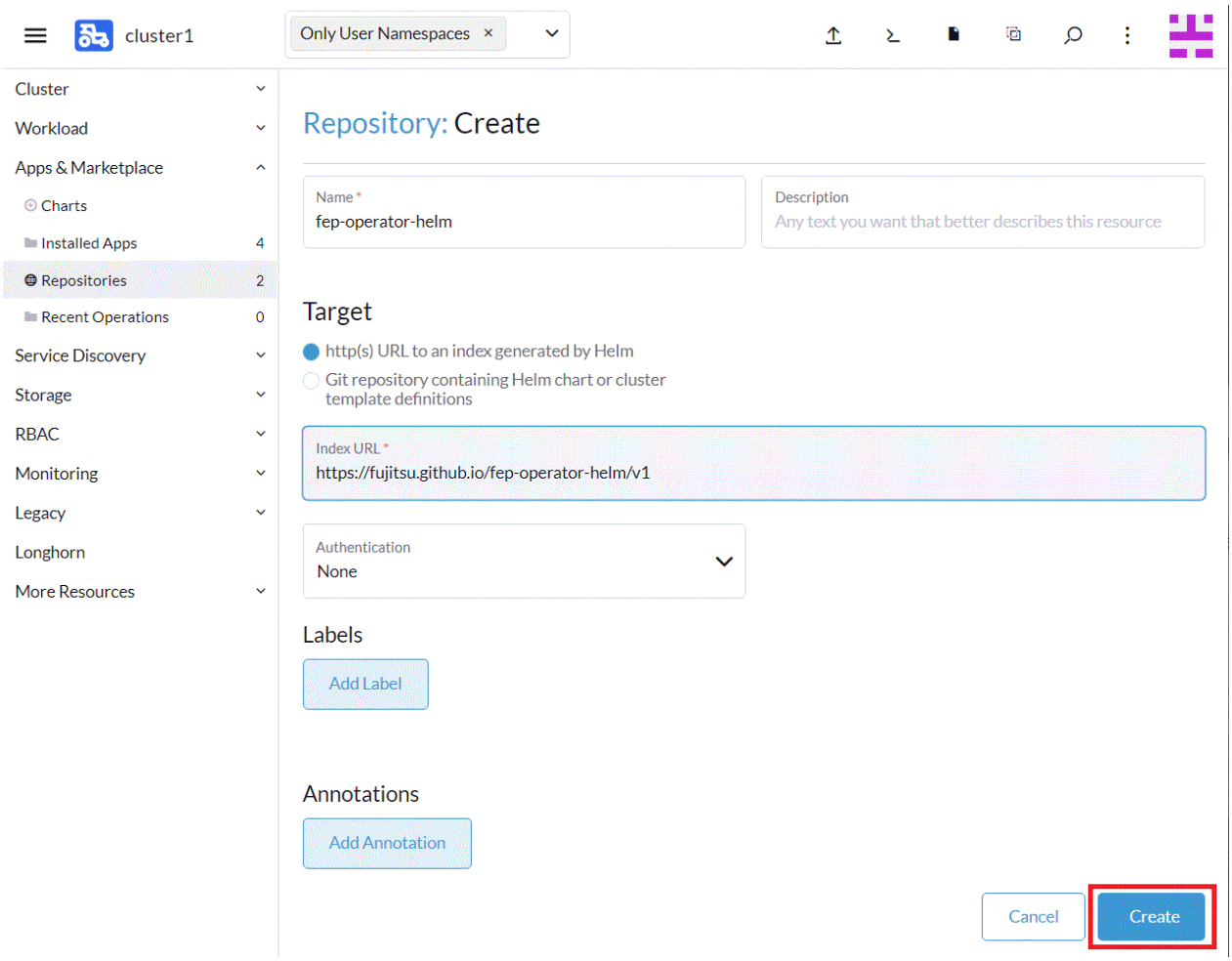


Helm Chartリポジトリを作成するために、[Create]をクリックします。



カタログの一意の名前と、下記カタログのURLを記載し、[Create]をクリックします。

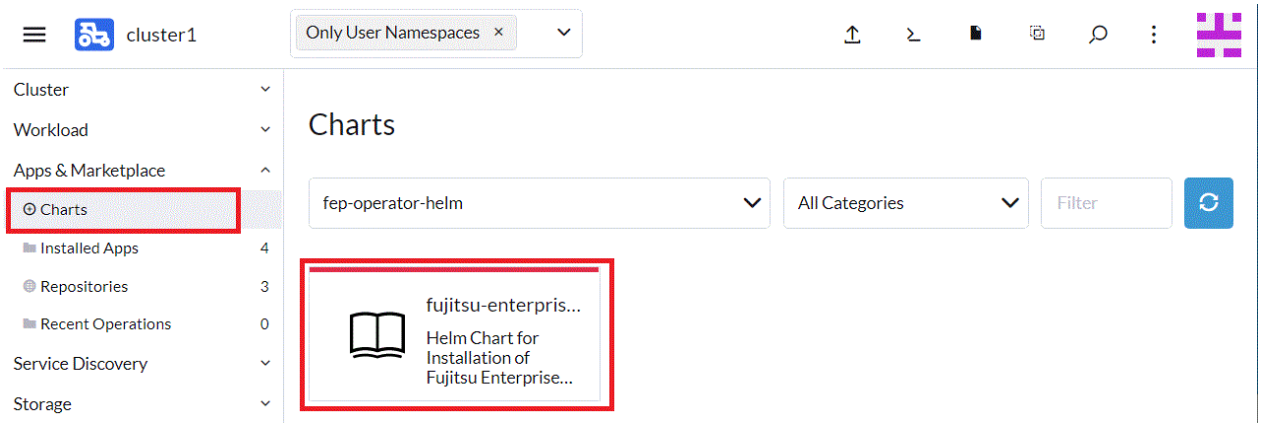
```
https://fujitsu.github.io/fep-operator-helm/v1
```



3.3.3 オペレーターのデプロイ

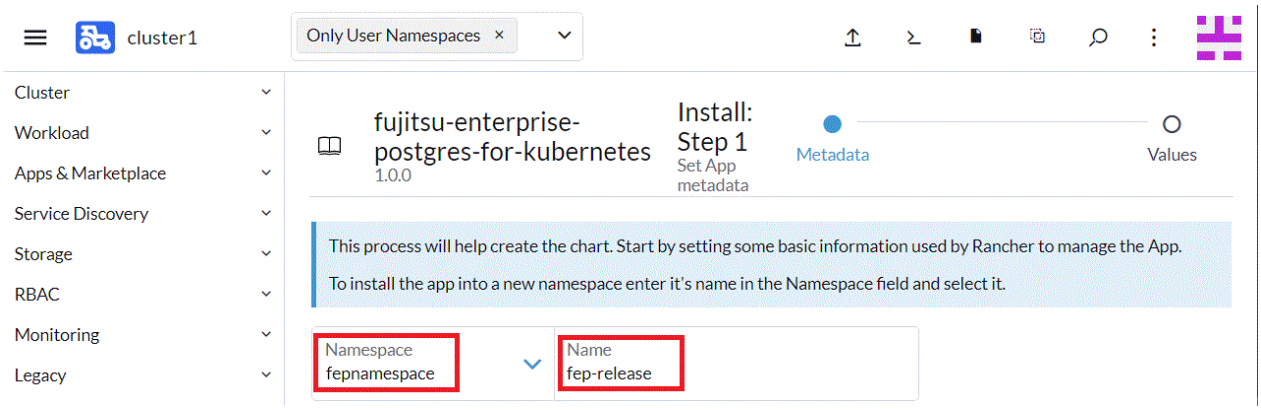
Rancher UI上で、“3.3.1 前提条件”で作成したプロジェクト/名前空間に対して、オペレーター機能のHelm Chartを適用し、オペレーターをインストールします。

左端のタブから、[Charts]をクリックし、[fujitsu-enterprise-postgres-operator]をクリックします。

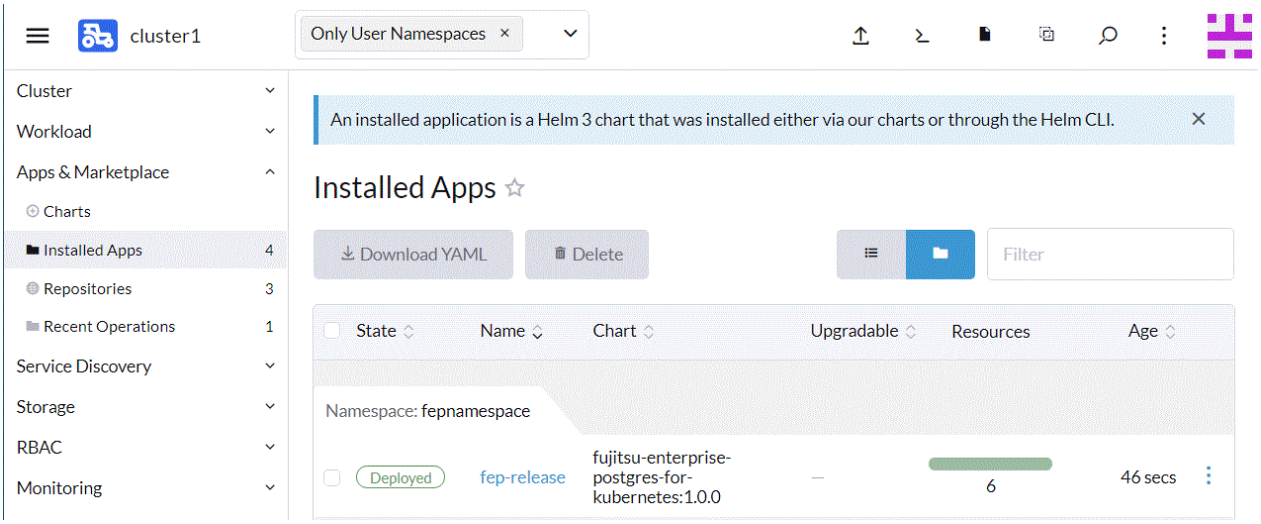


表示された画面上の[Install]をクリックします。

[Namespace]の項目を、“3.3.1 前提条件”で作成した名前に変更し、[Name]の項目にリリース名を入力して[Next]をクリックし、次の画面でそのまま[Install]をクリックします。



対象の名前空間上にオペレーターがデプロイされます。



3.4 連携するモニタリングツールの導入

3.4.1 GAPスタックの導入

FEPExporterを実行するためには、以下の前提条件があります。

- GAP（Grafana、Alertmanager、Prometheus）スタックがOpenShiftまたはKubernetesクラスタにインストールされている
- スクレイプする必要のあるFEPClusterがデプロイされ、適切に実行されている

- FEPClusterのpostgresql.confには以下が設定されている
  - pg_stats_statementsライブラリがプリロードされている
  - track_activitiesとtrack_countsが“on”になっている

PrometheusおよびAlertmanagerの場合、Openshiftにプリインストールされている監視スタックを使用します。展開情報については、以下を参照してください。

(https://docs.openshift.com/container-platform/4.11/monitoring/monitoring-overview.html#understanding-the-monitoring-stack_monitoring-overview)

Grafanaの場合、OperatorHubから提供されているGrafanaOperatorをインストールして使用します。

Grafanaはs390xおよびppc64leのOperatorHubによって公開されないため、Helmを使用してGrafanaをビルドします。詳細な手順は、次のサイトを参照してください。

(https://www.postgresql.fastware.com/knowledge-base/how-to/setting-up-grafana-on-ibm-linuxone)

OpenShiftにGrafanaがプリインストールされていますが、OperatorHubで公開されているGrafanaを使用して、ダッシュボードをカスタマイズし、FEPパフォーマンス情報を監視することをお勧めします。

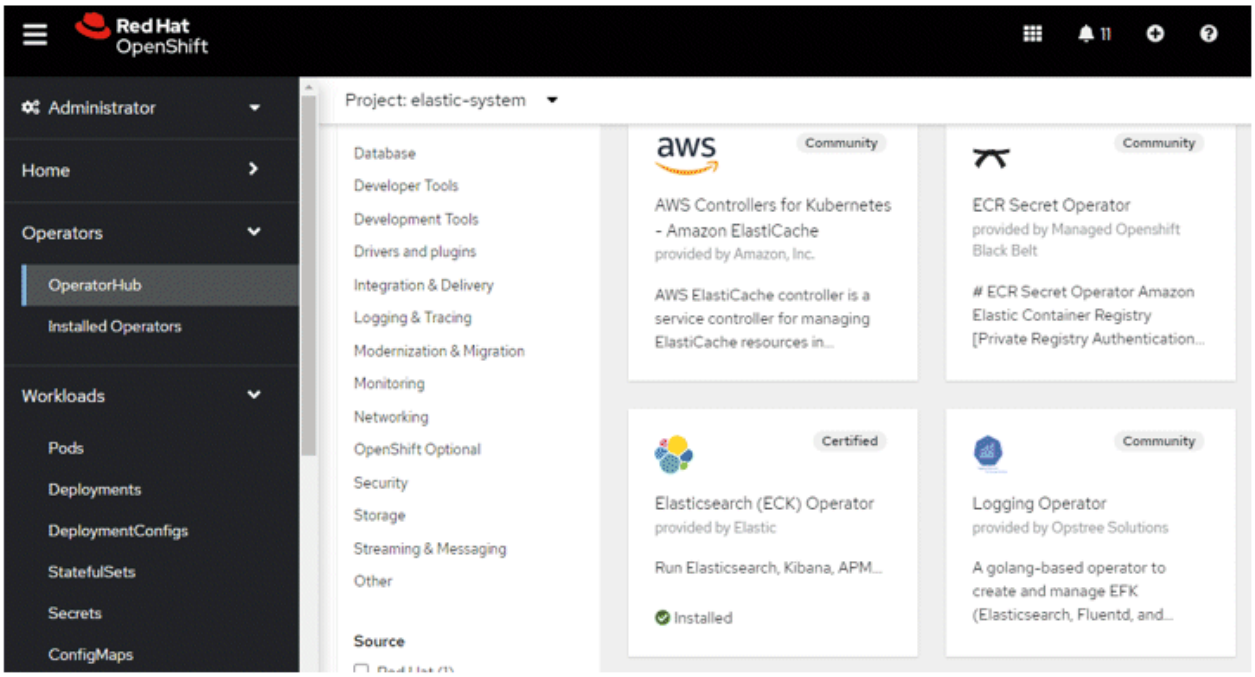
なお、以下で公開されているサンプルダッシュボードをご利用いただけます。

https://github.com/fujitsu/fep-operator-examples/blob/v4/Monitoring/dashboard/fep-dashboard.json

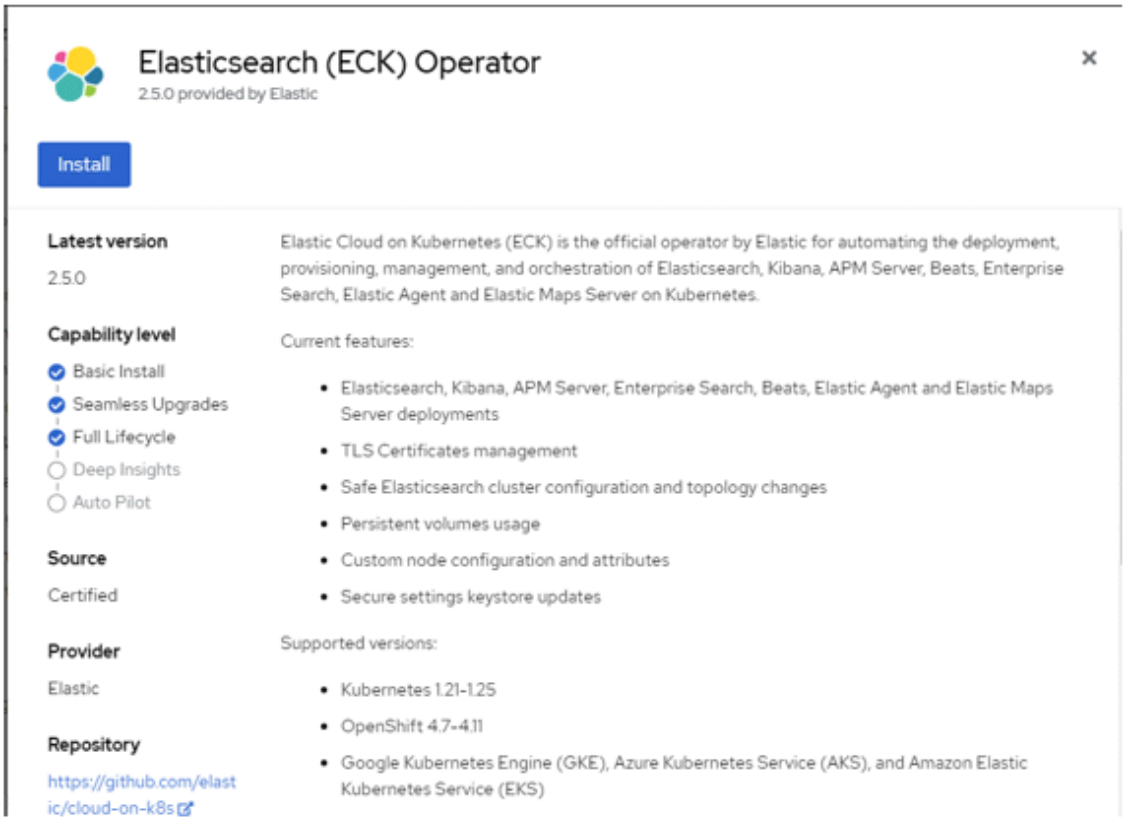
3.4.2 Elastic Cloud on Kubernetesの導入

3.4.2.1 ECKオペレーターのデプロイ

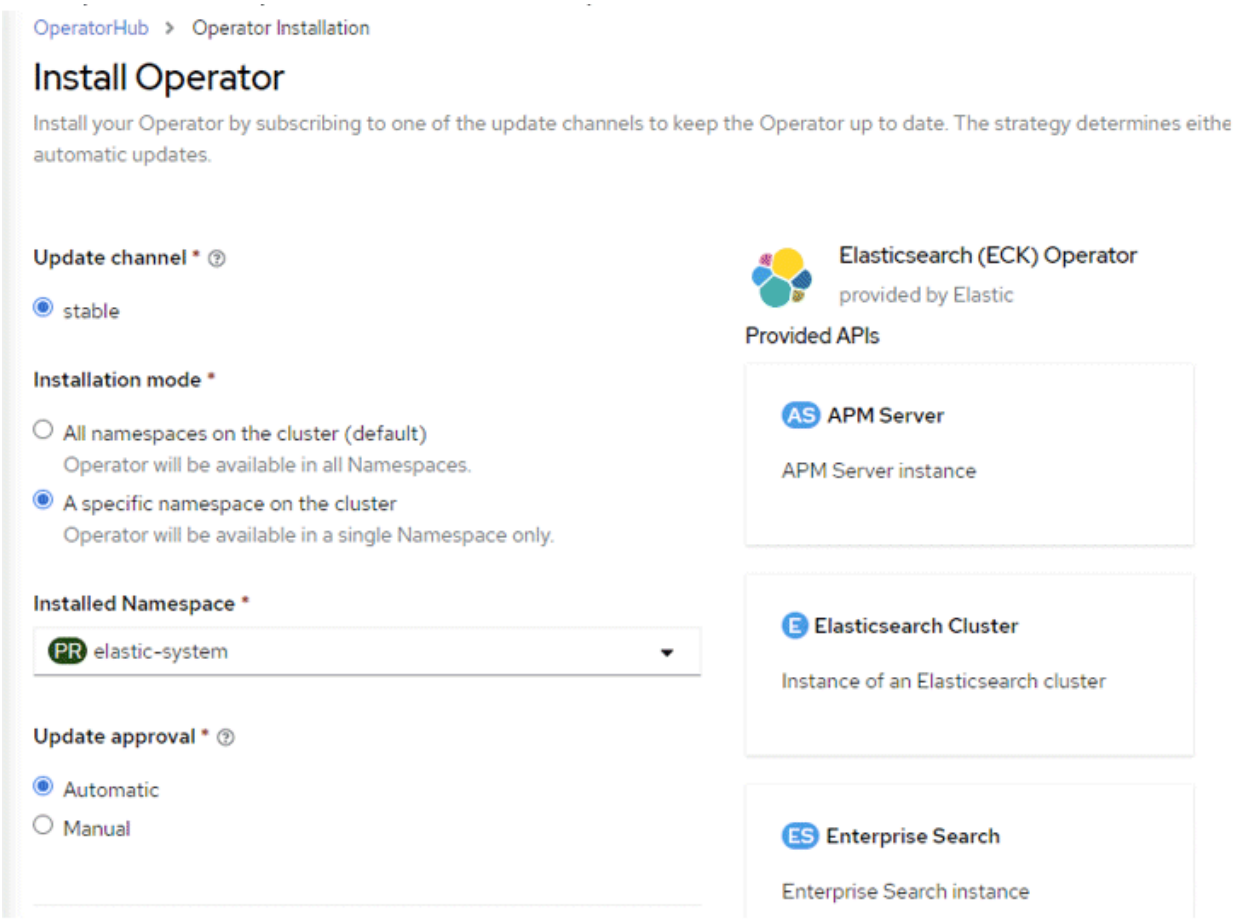
1. elastic-system名前空間(プロジェクト)を作成します。
2. OperatorHub に、Elastic が提供するElasticsearch(ECK) Operator をインストールします。



3. [Install] をクリックして続行します。

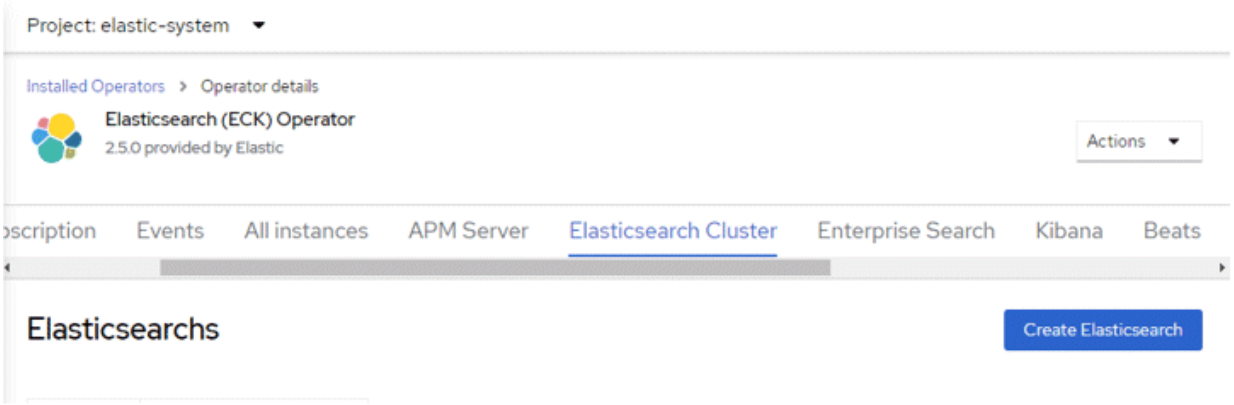


4. [Installation mode]を[A specific namespace on the cluster]に変更し、名前空間[elastic-system]を選択します。[Install]をクリックしてインストールを完了します。



3.4.2.2 Elasticsearchクラスタのデプロイ

1. [Installed Operators]で、[Elasticsearch (ECK) Operator]を選択します。
2. [Elasticsearch Cluster]と[Create Elasticsearch]を選択します。



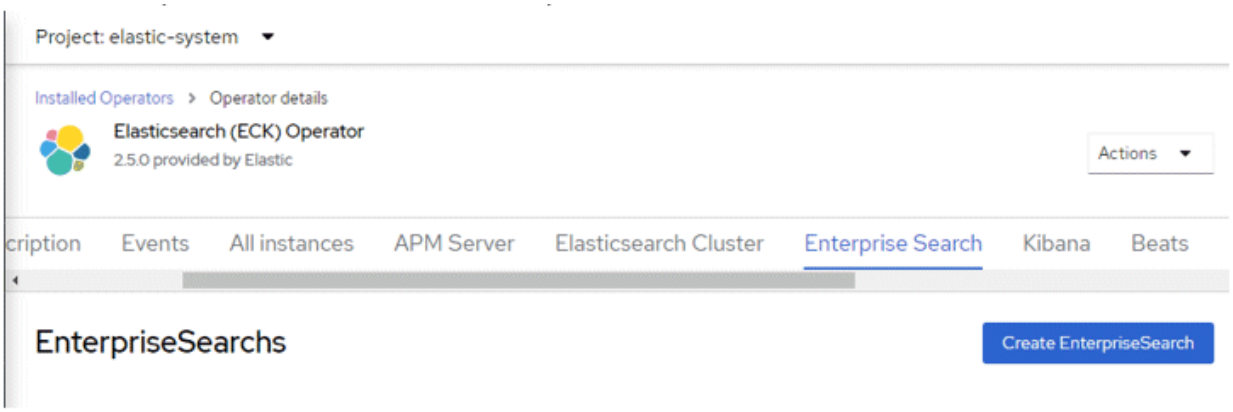
3. YAMLビューで以下を入力し、[Create]をクリックします。

```
apiVersion: elasticsearch.k8s.elastic.co/v1
kind: Elasticsearch
metadata:
  name: quickstart
```

```
spec:
  version: 8.5.2
  nodeSets:
  - name: default
    count: 1
    config:
      node.store.allow_mmap: false
```

3.4.2.3 Enterprise Searchのデプロイ

1. [Installed Operators]で、[Elasticsearch (ECK) Operator]を選択します。
2. [Enterprise Search]と[Create EnterpriseSearch]を選択します。

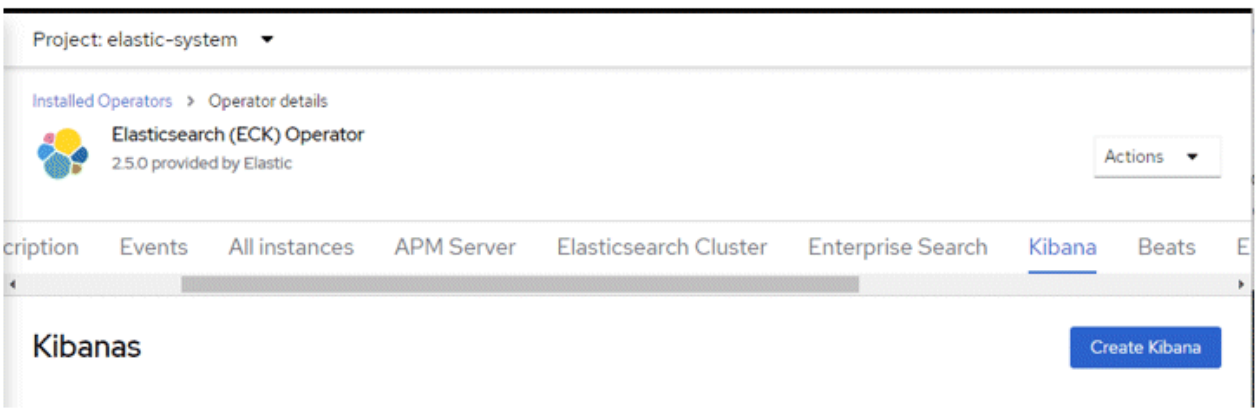


3. YAMLビューで以下を入力し、[Create]をクリックします。

```
apiVersion: enterprisesearch.k8s.elastic.co/v1
kind: EnterpriseSearch
metadata:
  name: enterprise-search-quickstart
spec:
  version: 8.5.2
  count: 1
  elasticsearchRef:
    name: quickstart
```

3.4.2.4 Kibanaのデプロイ

1. [Installed Operators]で、[Elasticsearch (ECK) Operator]を選択します。
2. [Kibana]と[Create Kibana]を選択します。



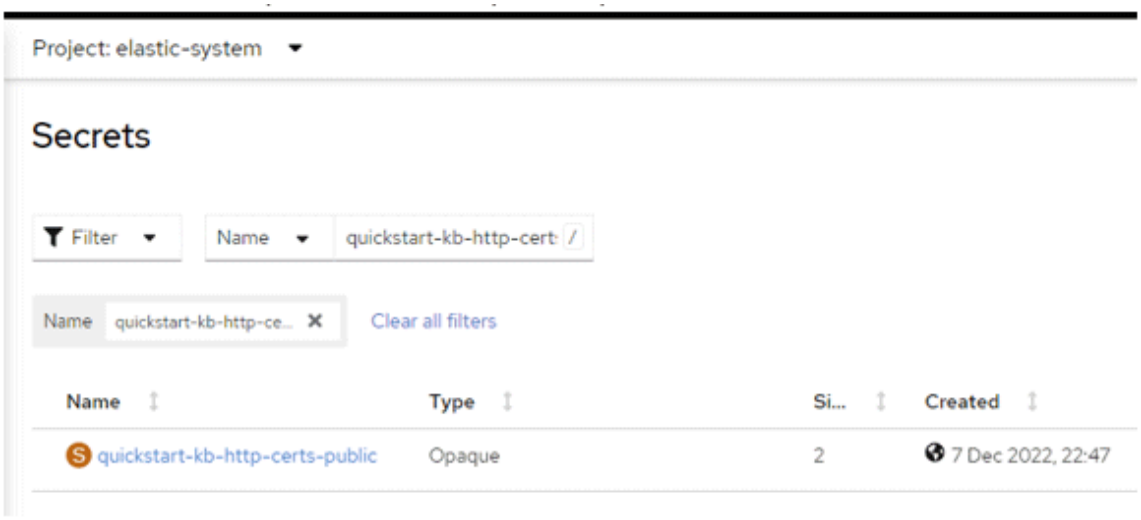
3. YAMLビューで以下を入力し、[Create]をクリックします。

```
piVersion: kibana.k8s.elastic.co/v1
kind: Kibana
metadata:
  name: quickstart
spec:
  count: 1
  elasticsearchRef:
    name: quickstart
  enterpriseSearchRef:
    name: enterprise-search-quickstart
  version: 8.5.2
```

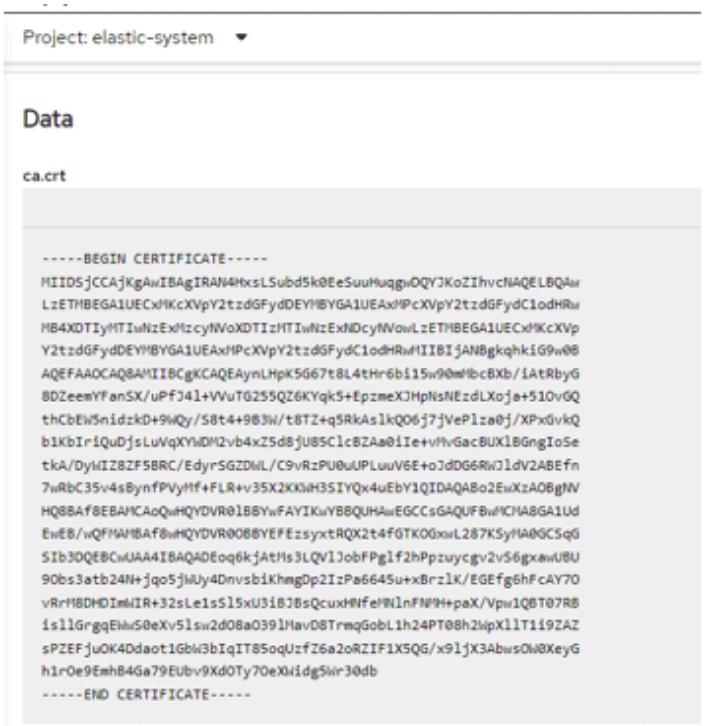
3.4.2.5 OpenShift Routeを使用したKibanaの公開

1. Kibana 証明書に署名するCA 証明書を取得します。

   シークレットの[quickstart-kb-http-certs-public]を選択します。



ca.crtの内容をcopyします。



2. インターネットアクセスのルートを作成します。

   [Navigate to Networking] -> [Route]を選択し、[Create Route]を選択します。

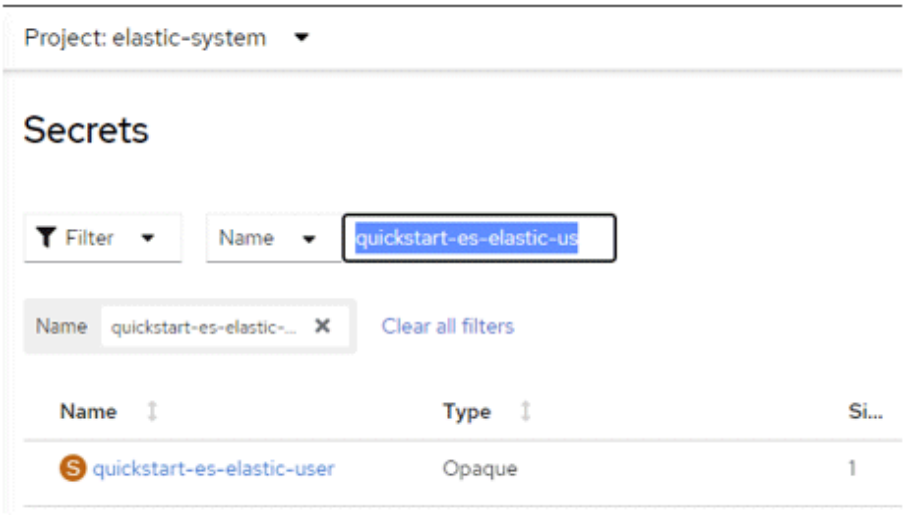
詳細を入力します。

Key	Value
Name	kibana
Hostname	Leave empty
Path	/
Service	quickstart-kb-http
Target port	5601 -> 5601 (TCP)
Secure Route	selected
TLS termination	Re-encrypt
Insecure traffic	Redirect
Destination CA certificate	Content of ca.crt in previous step

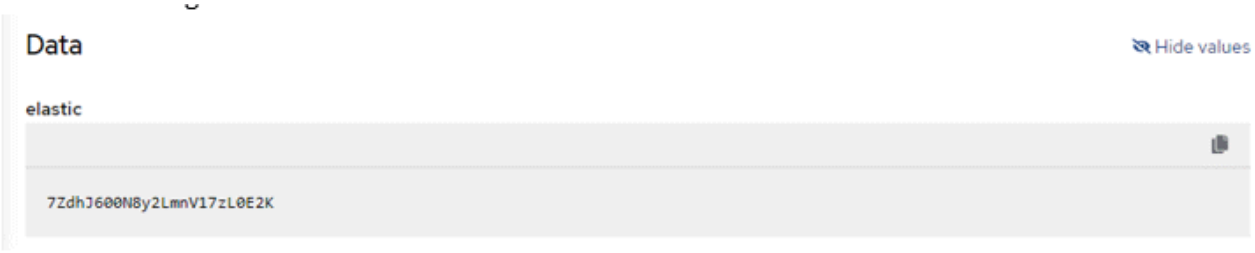
3.4.2.6 Kibanaへのログイン

1. Elasticsearch/Kibana ログインの詳細を取得します。

   シークレットの[quickstart-es-elastic-user]を選択します。

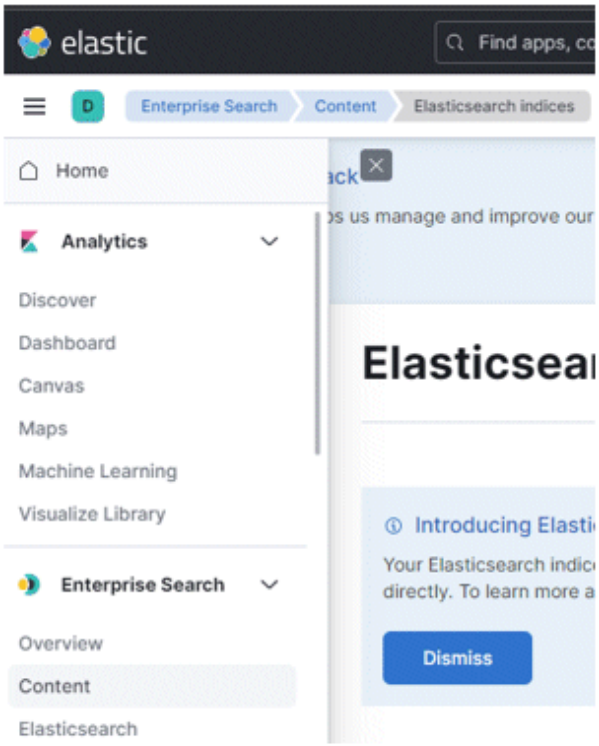


2. ログインの詳細を確認します。



3. 上記のルートで作成されたURL にアクセスし、上記の資格情報を使用してログインします。

4. 左上メニュー アイコンから[Enterprise Search] -> [Content] を選択します。



5. fluentdがlasticクラスタにログを転送している場合、ここにインデックスがあります。

3.5 クライアントの導入

FEPクライアントは、媒体のものをご利用いただくか、以下のサイトからrpm形式のモジュールをダウンロードしてご利用ください。

https://www.postgresql.fastware.com/fujitsu-enterprise-postgres-client-download

第4章 コンテナのデプロイ

本章では、コンテナのデプロイについて説明します。

GitHub上の下記のリポジトリで、Fujitsu Enterprise Postgres Operatorを使用して、データベースを運用するためのCRテンプレート(サンプルファイル)を公開しています。テンプレートを利用することで、コンテナのデプロイ作業が簡易化できます。

https://github.com/fujitsu/fep-operator-examples

注意

FEPClusterのデプロイによって作成されるPodの各ボリュームは、下記運用を想定したサイズがデフォルトで設定されます。

- データサイズ:1GB
- 1日当たりの更新量:50MB程度

実際の運用に応じて、“2.3.3 クラスタごとに構成可能なボリューム”を参照し、各ボリュームサイズの設計を行ってください。

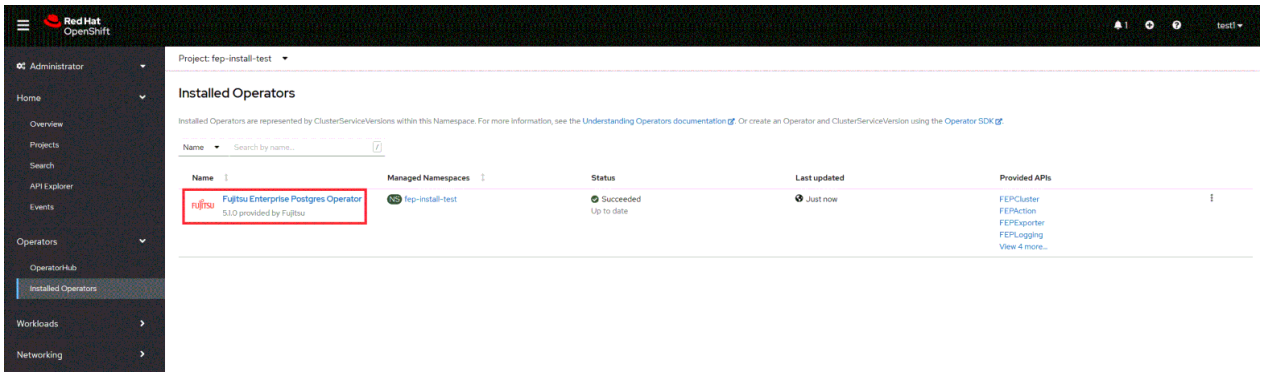
4.1 オペレーターを使用したFEPClusterのデプロイ

以下の手順で、指定された名前空間にFEPClusterをデプロイします。

注意

Kubernetesクラスタ上へデプロイする場合は、“リファレンス”の“カスタムリソースパラメータ”を参照してyamlファイルを作成し、適用してください。

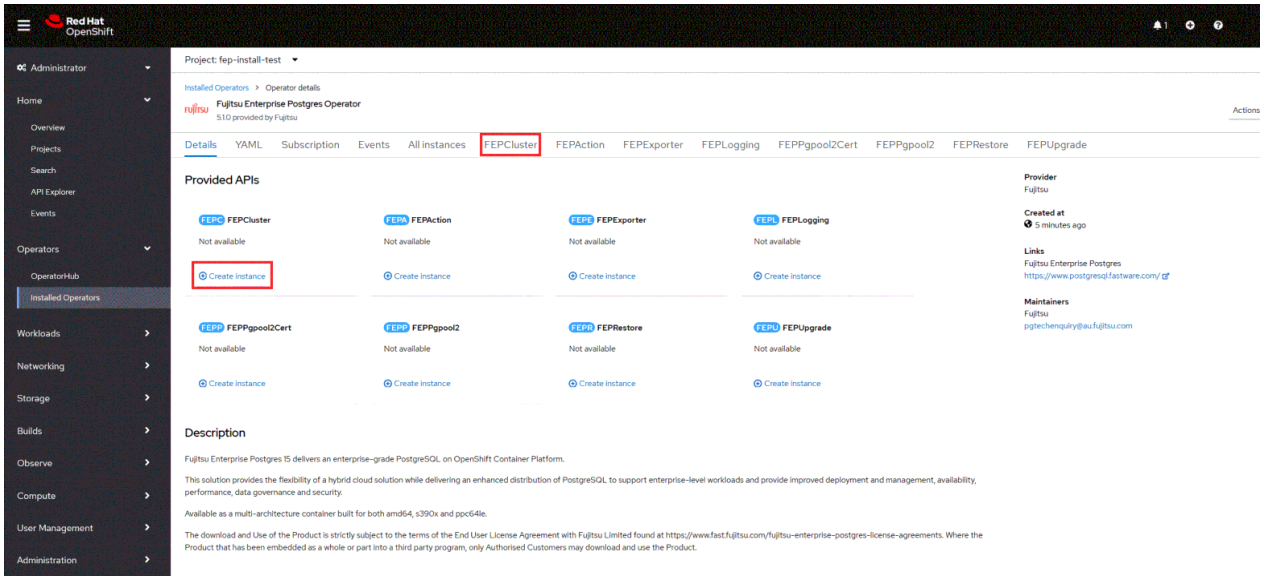
1. [Operators]メニュー項目で、[InstalledOperators]をクリックします。“第3章 オペレーターのインストール”でインストールしたオペレーターが表示されています。オペレーターの名前をクリックします。



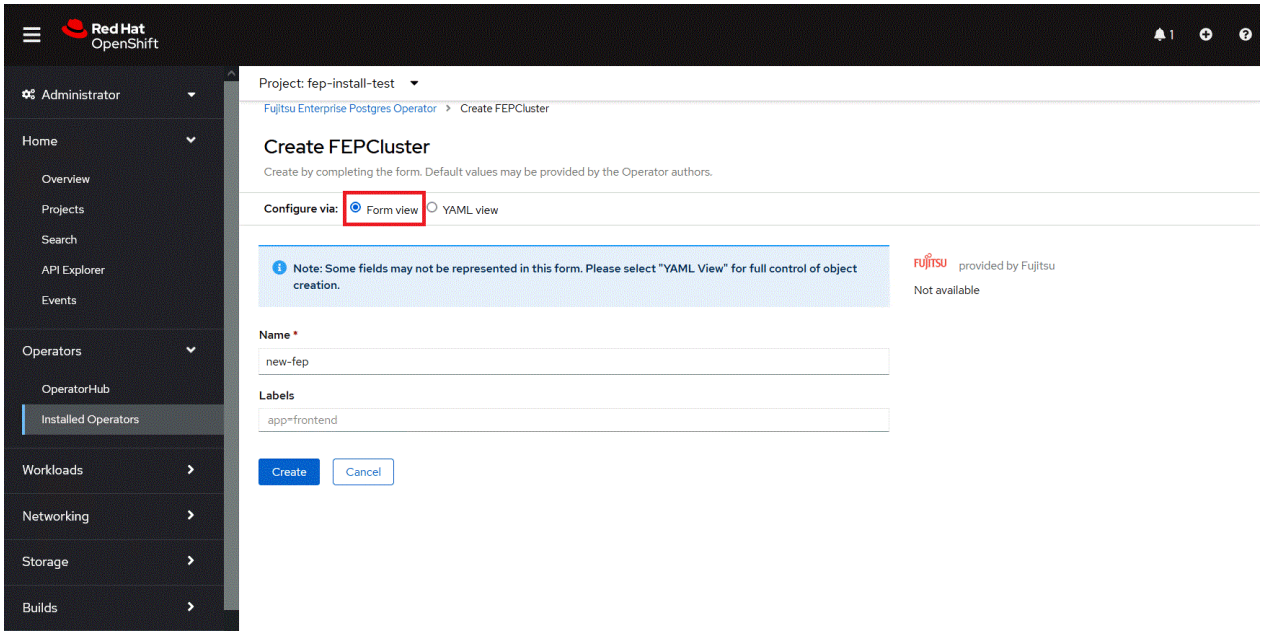
2. このオペレーターがサポートするすべてのカスタムリソースを含むページが表示されます。FEPClusterがメインのカスタムリソースで、その他はすべて子のカスタムリソースです。メインのカスタムリソースを作成すると、他のすべてのカスタムリソースはオペレーターによって自動的に作成されます。

   以下のどちらかの方法でFEPClusterのカスタムリソースを作成します。
   (1) FEPCluster配下の[Create Instance]をクリックします。

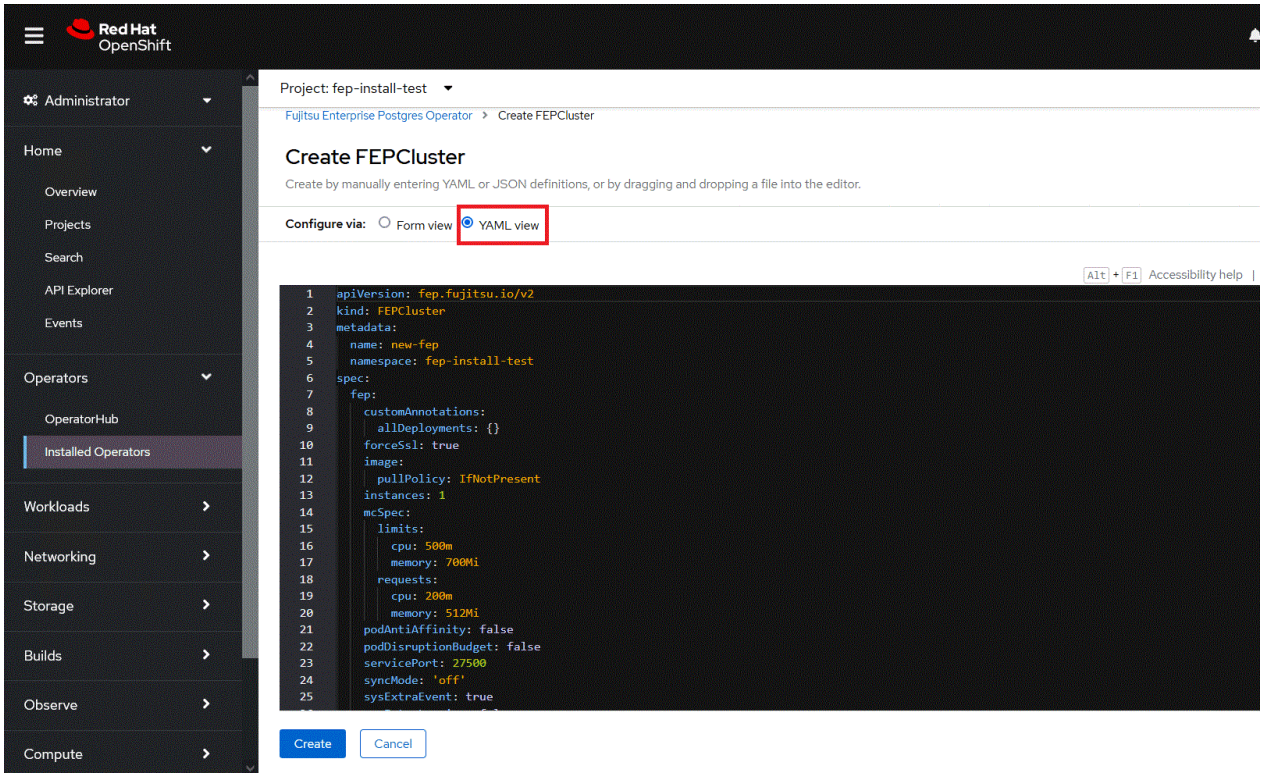
または
(2) 上部の[FEPCluster]をクリックしてから、次のページの[Create FEPCluster]をクリックします。



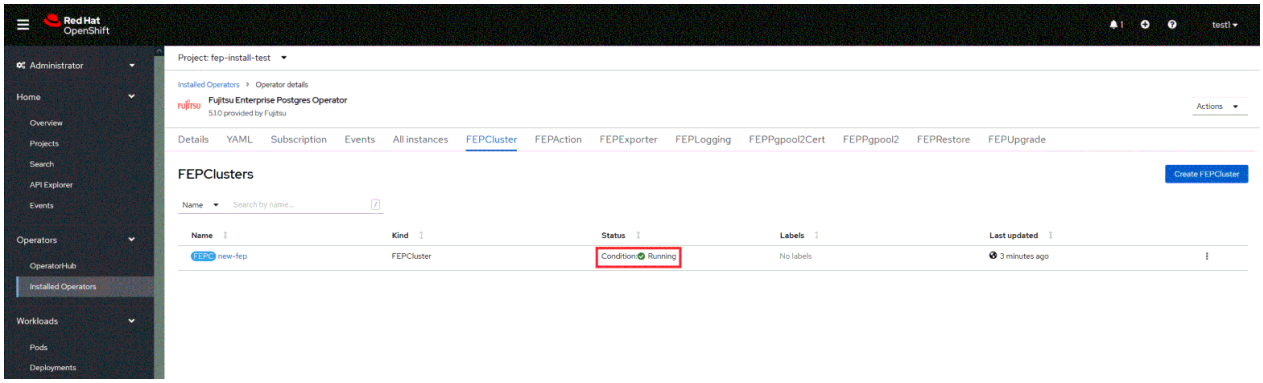
3. これにより、[Create FEPCluster]ページが表示されます。このページには、設定のための2つのオプションがあります。1つは[Form View]です。現時点では、[Form View]では、デプロイされているクラスタの名前のみが変更できます。デフォルト名は「new-fep」です。
この名前は、名前空間内で一意である必要があります。



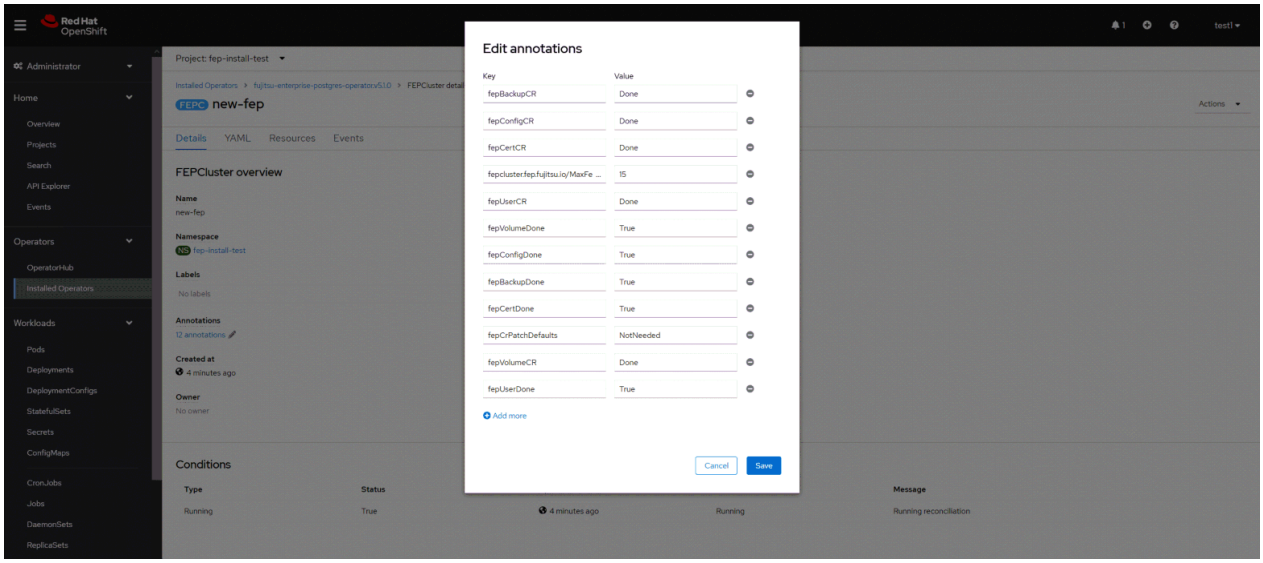
4. また、[YAML View]では、カスタムリソースの開始値が表示され、カスタムリソースを作成する前にパラメータを変更することができます。パラメータの詳細については、“リファレンス”を参照してください。



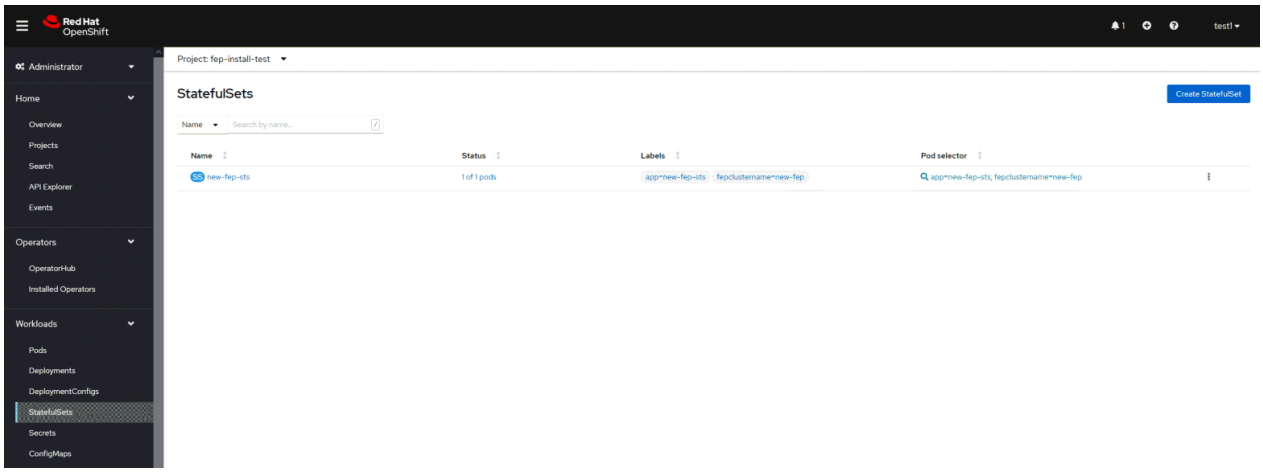
5. 上記の2つのページのいずれかで[Create]をクリックすると、オペレータはFEPClusterカスタムリソースを作成し、その後、FEPBackup、FEPConfig、FEPVolume、FEPUser、およびFEPCertの子カスタムリソースが1つずつ自動的に作成されます。
子カスタムリソースの開始値は、FEPCluster のYAMLファイルの「fepChildCrVal」セクションから取得されます。FEPClusterの「fepChildCrVal」セクションの値を変更します。オペレーターは、FEPClusterの親カスタムリソースからそれぞれの子カスタムリソースへの変更を反映します。許容される変更のみが子カスタムリソースに反映されます。子CRは内部オブジェクトとしてマークされているため、OCPコンソールには表示されません。ただし、コマンドラインツールを使用して子カスタムリソースを確認できます。



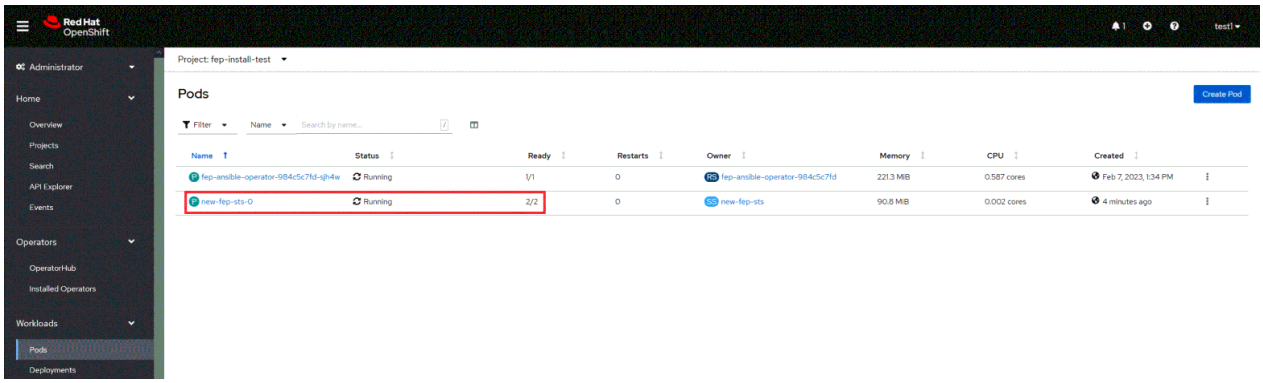
6. FEPClusterカスタムリソースには、子カスタムリソースが正常に作成され、適切に初期化されたことを示す注釈が追加されます。完了するまでに時間がかかる場合があります。



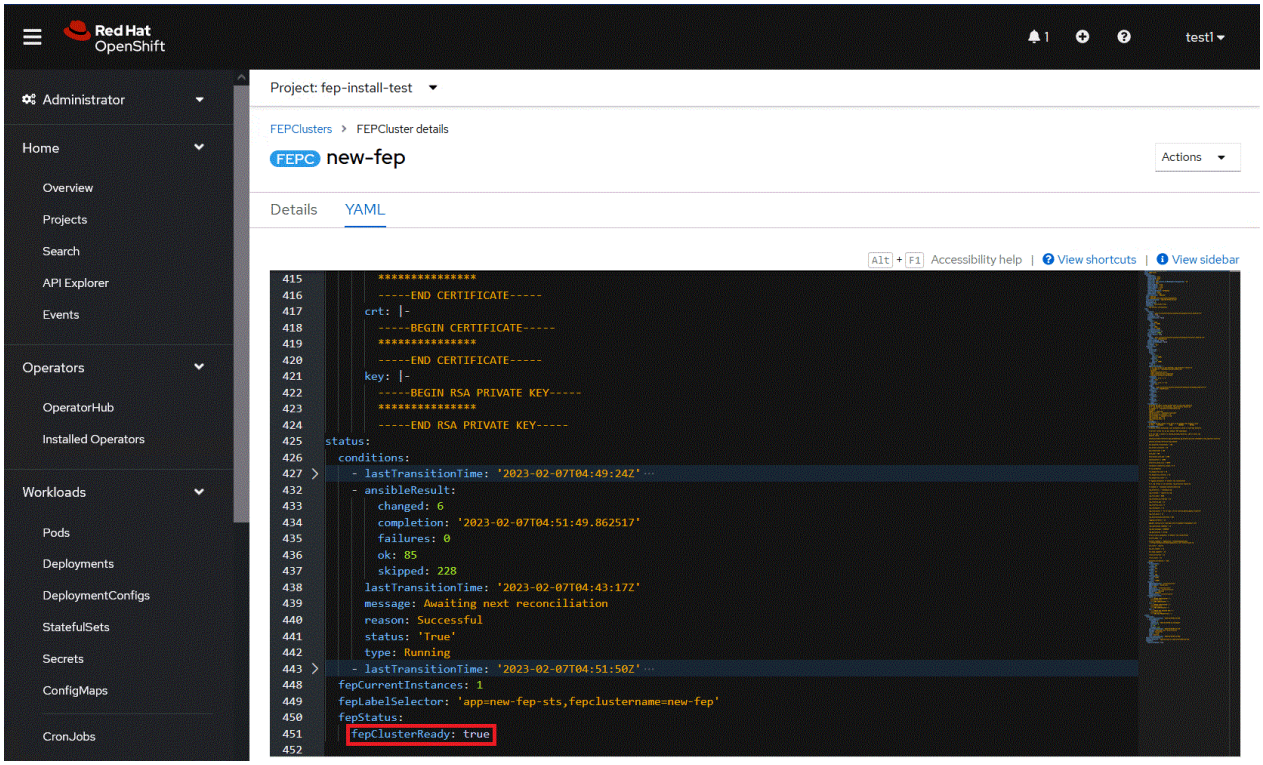
7. 子カスタムリソースの作成が完了すると、オペレーターはクラスタのStatefulSetを作成します。



8. StatefulSetは、1つのFEPインスタンスを開始し、準備が完了してから、次のインスタンスを開始します。



9. FEPサーバのすべてのインスタンスが開始されると、オペレーターはカスタムリソースの[status.fepStatus]セクションの下にあるフラグ[fepClusterReady]をtrueに設定し、FEPClusterを使用する準備が完了したことを示します。FEPClusterカスタムリソースのYAMLを見ると、次のようになります。



10. オペレーターは、FEPCluster fepChildCrValおよびそれぞれの子カスタムリソースのパスワード、パスフレーズ、証明書、キーなどの機密フィールドもマスクします。

4.2 高可用性FEPClusterのデプロイ

高可用性FEPクラスタでは、参照クエリをレプリカインスタンスに分散することで負荷分散が可能です。

さらに、マスターインスタンスに障害が発生した場合、ユーザーはすぐにレプリカインスタンスに切り替えて、業務の停止を局所的にすることができます。

高可用性構成では、レプリカインスタンスの同期モードを選択できます。マスターインスタンスに障害が発生した場合のデータ損失を許容できないシステムには、同期レプリケーションをお勧めします。

高可用性構成で複数のインスタンスが作成されるため、それぞれにライセンスが必要です。

指定された名前空間に可用性の高いFEPClusterをデプロイするには、次の手順に従います。

[前提条件]

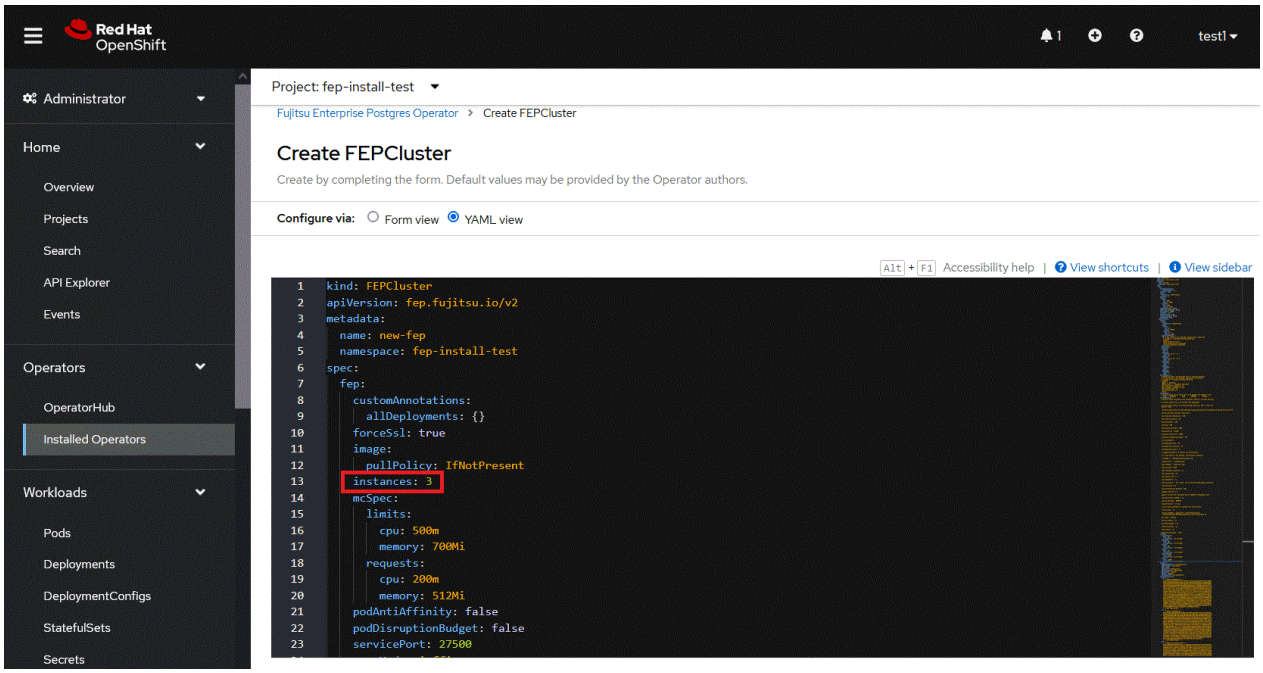
FEPクラスタがHAモードで実行されている場合、バックアップおよびアーカイブWALボリュームは、ReadWriteManyをサポートする共有ストレージ(NFSなど)で構成する必要があります。共有ストレージの設定手順については、Openshiftのドキュメントを参照してください。また、必要に応じて、「付録C 共有ストレージの使用」を参照してください。

共有ストレージがない場合は、バックアップセクションとバックアップおよびアーカイブボリュームセクションを削除して、バックアップ機能を無効にし、FEPクラスタをデプロイできます。

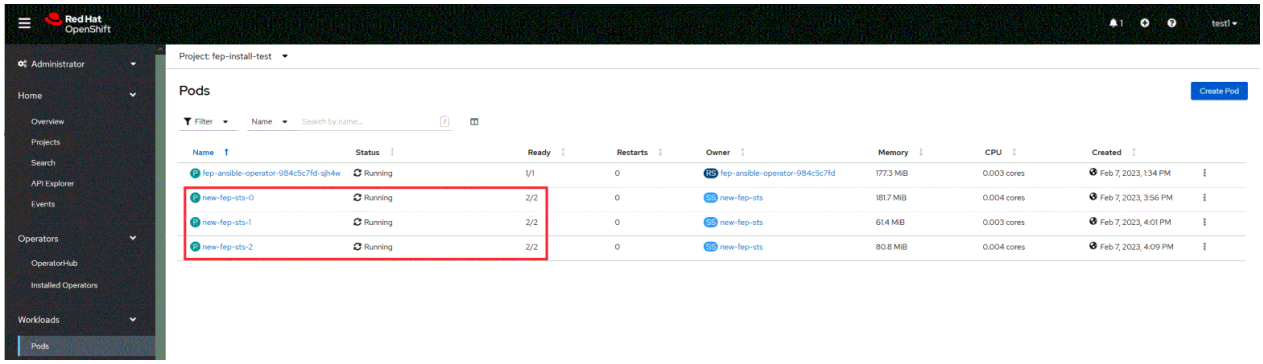
注意

Kubernetesクラスタ上へデプロイする場合は、“リファレンス”の“カスタムリソースパラメータ”を参照してyamlファイルを作成し、適用してください。

1. “4.1 オペレーターを使用したFEPClusterのデプロイ”の手順1から手順3までの手順と同じです。
2. “4.1 オペレーターを使用したFEPClusterのデプロイ”の手順4の代わりに、YAML Viewに変更し、「spec」の「fep」の「instances」パラメータに「3」を指定します。バックアップおよびアーカイブWALボリューム用に準備された共有ストレージのストレージクラスを指定します。



3. “4.1 オペレーターを使用したFEPClusterのデプロイ”の手順5から手順10までの手順と同じです。
4. 3つのPodがデプロイされ、高可用性FEPClusterの準備が整いました。



参考

以下のコマンドにより、マスタPodかレプリカPodかを確認できます。

```
$ oc get pod -L feprole
NAME                                  READY   STATUS    RESTARTS   AGE     FEPROLE
fep-ansible-operator-88f7fb4b-5jh85   1/1     Running   0          24m
new-fep-sts-0                         2/2     Running   0          17m     master
new-fep-sts-1                         2/2     Running   0          15m     replica
new-fep-sts-2                         2/2     Running   0          13m     replica
```

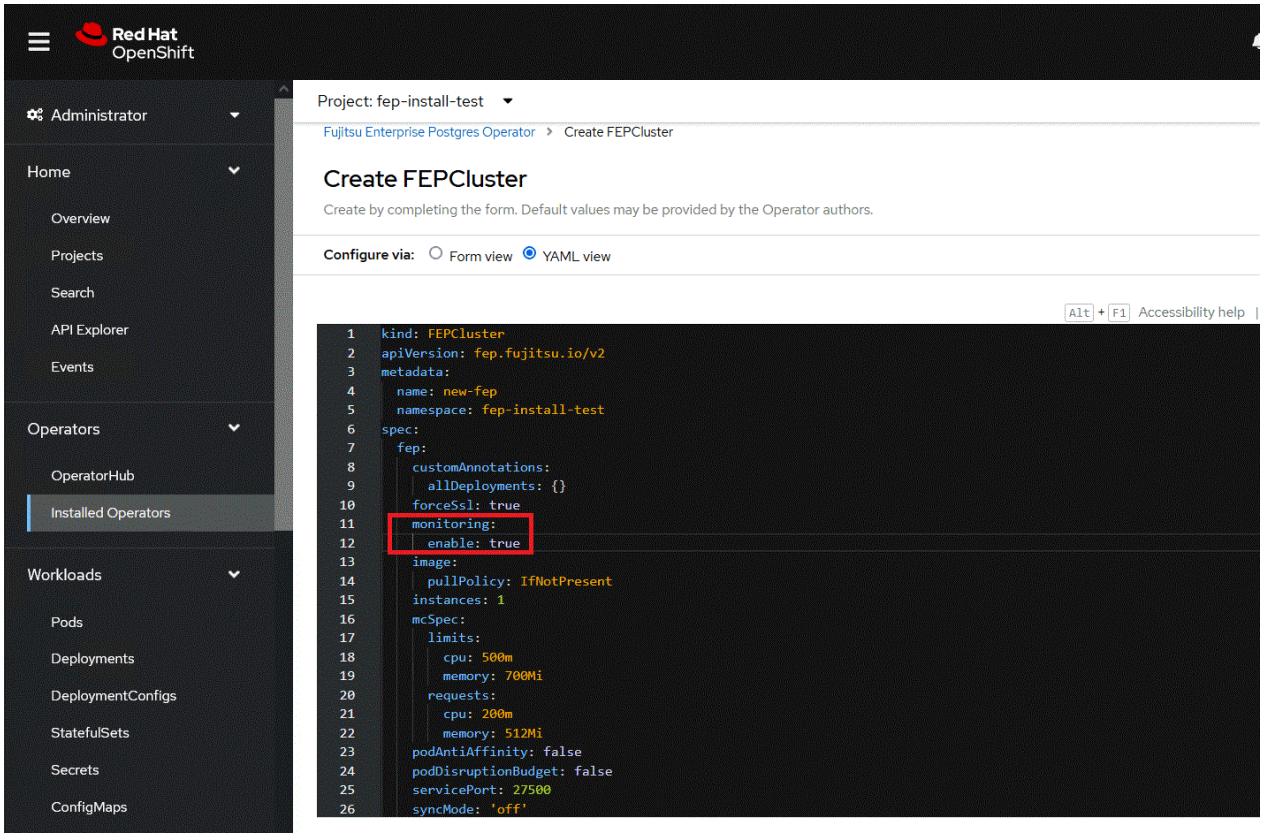
4.3 FEPExporterのデプロイ

以下の手順でFEPExporterをデプロイします。

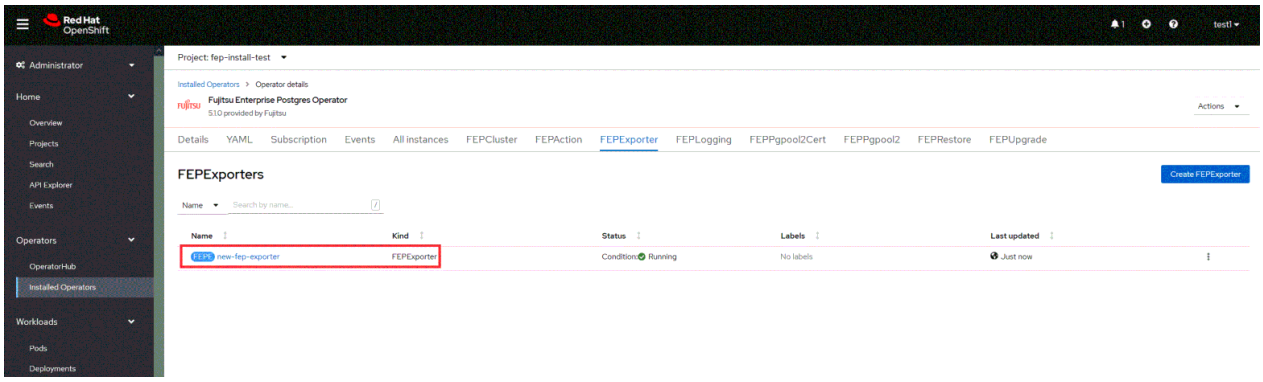
注意

Kubernetesクラスタ上へデプロイする場合は、“リファレンス”の“カスタムリソースパラメータ”を参照してyamlファイルを作成し、適用してください。

1. オペレーターが管理するFEPExporterをデプロイするには、デプロイ時にFEPClusterカスタムリソースで「fep.monitoring.enable」を「true」に設定するだけです。



2. FEPExporterは、<cluster-name> -fepexporterという名前で自動的に作成されます。また、指定したFEPclusterの統計を含むすべてのデータベースが一覧表示されます。



3. オペレーターによって生成されたFEPExporterは、デフォルトでマスターインスタンスとスタンバイインスタンスからメトリクスをスクレイピングし、Prometheusで使用できるようにします。

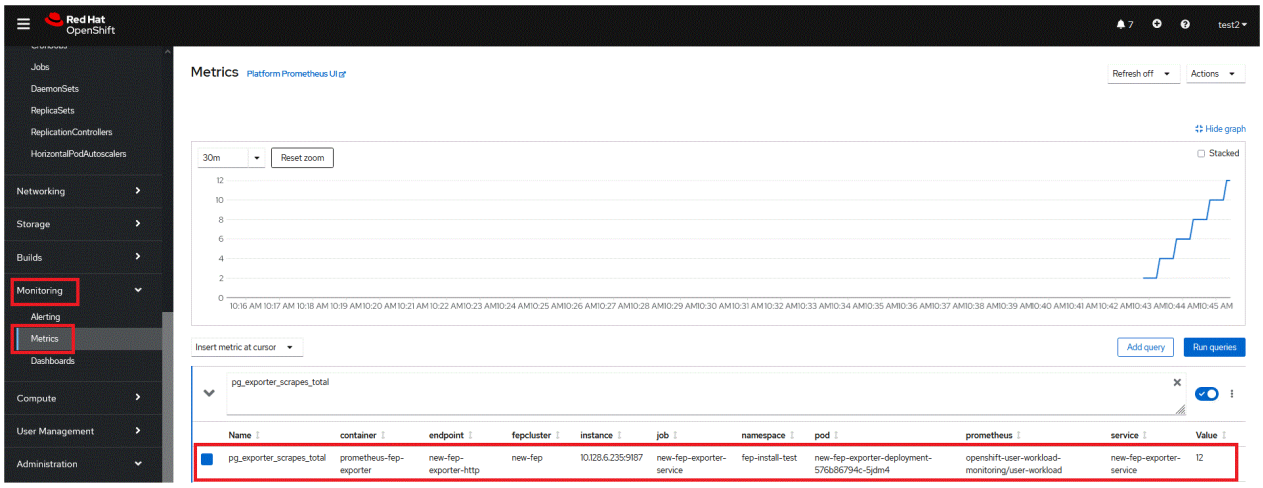
4. ユーザーは、Prometheusがメトリクススクレイピングおよび FEPExporterからデータベースへの接続に使用するHTTPエンドポイントに使用されるようにMTLSを構成できます。

   a. pgMetricsUser、pgMetricsPassword、およびpgMetricsUserTlsがFEPclusterで定義されている場合、FEPExporterは、FEPインスタンスへの接続を保護するためにこれらを使用します。これらのパラメータがない場合、FEPExporterはpgAdminUser（つまりスーパーユーザー）を使用します。

   b. MTLSでもFEPExporterによるメトリクスエンドポイント（/ metrics）が使用されるようにPrometheus.tlsとFEPExporter.tlsを設定できます（フィールドの詳細については、“リファレンス”の“FEPExporterカスタムリソース”を参照してください）。

5. ユーザー名とパスワードを含むシークレットを指定することにより、Basic認証を構成することもできます。（フィールドの詳細については、“リファレンス”の“FEPExporterカスタムリソース”を参照してください）

6. PROMQLを使用して対象のメトリクスを指定することにより、[Monitaring]領域でOpenshiftPlatformのスクレイプFEPExporter固有のメトリクスを確認できます。



注意

- すでにインスタンス化されているクラスタでもfep.monitoring.enableをtrueまたはfalseに設定して、目的の結果を得ることができます。
- pgMetricsUserは、モニタリングを有効にして実行中のFEPClusterで後で定義でき、FEPExporterにpgMetricsUserを再起動するだけで強制的に使用できます（restartRequiredを参照）。ただし、この場合、MTLSを構成することはできず、ユーザーは、情報のスクレイピング中に使用されることが予想されるすべてのデータベースオブジェクトに対してpgMetricsUserに特定のアクセス許可を付与する必要があります。
- MTLSを強制するには、usePodNameとpg_hba.confが適切に設定されていることを確認してください。
- FEPExporterのデフォルトのメトリクスは、postgresql.confで以下を設定する必要があります。
  - pg_stats_statementsライブラリがプリロードされている
  - track_activitiesとtrack_countsがon
  - モニタリングユーザーには、pg_stat_*ビューに対する権限が必要
- CPUメモリに関連するFEPExporterのPodの仕様を変更できます。リソースの指定を変更した後、restartRequiredフラグをtrueに設定します。FEPexporterは新しい仕様で再起動されます。
- FEPのモニタリングは、プラットフォームで利用可能なPrometheusと連携されています。ユーザーは、OpenShiftプラットフォームでユーザー定義プロジェクトのモニタリングが有効になっていることを確認する必要があります（https://docs.openshift.com/container-platform/4.11/monitoring/enabling-monitoring-for-user-defined-projects.htmlを参照）。OpenShift以外のプラットフォームの場合、オペレーターをデプロイする前にPrometheusがインストールされていることを確認してください。

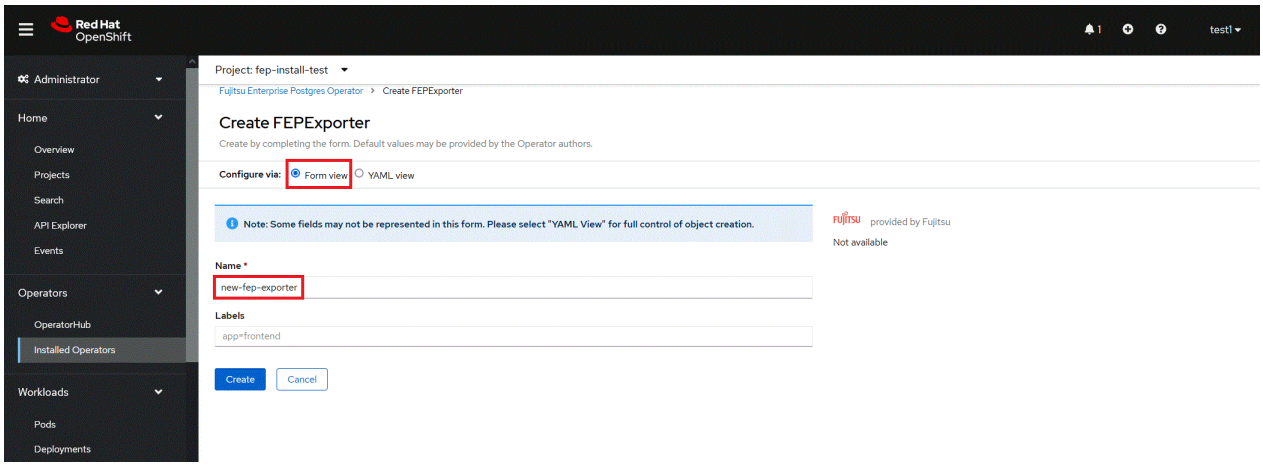
4.4 スタンドアロンモードでのFEPExporter

FEPExporterは独立したカスタムリソースです。したがって、必ずしもメインのFEPClusterに依存するわけではありません。指定された名前空間にFEPExporterをデプロイするには、以下の手順に従います。

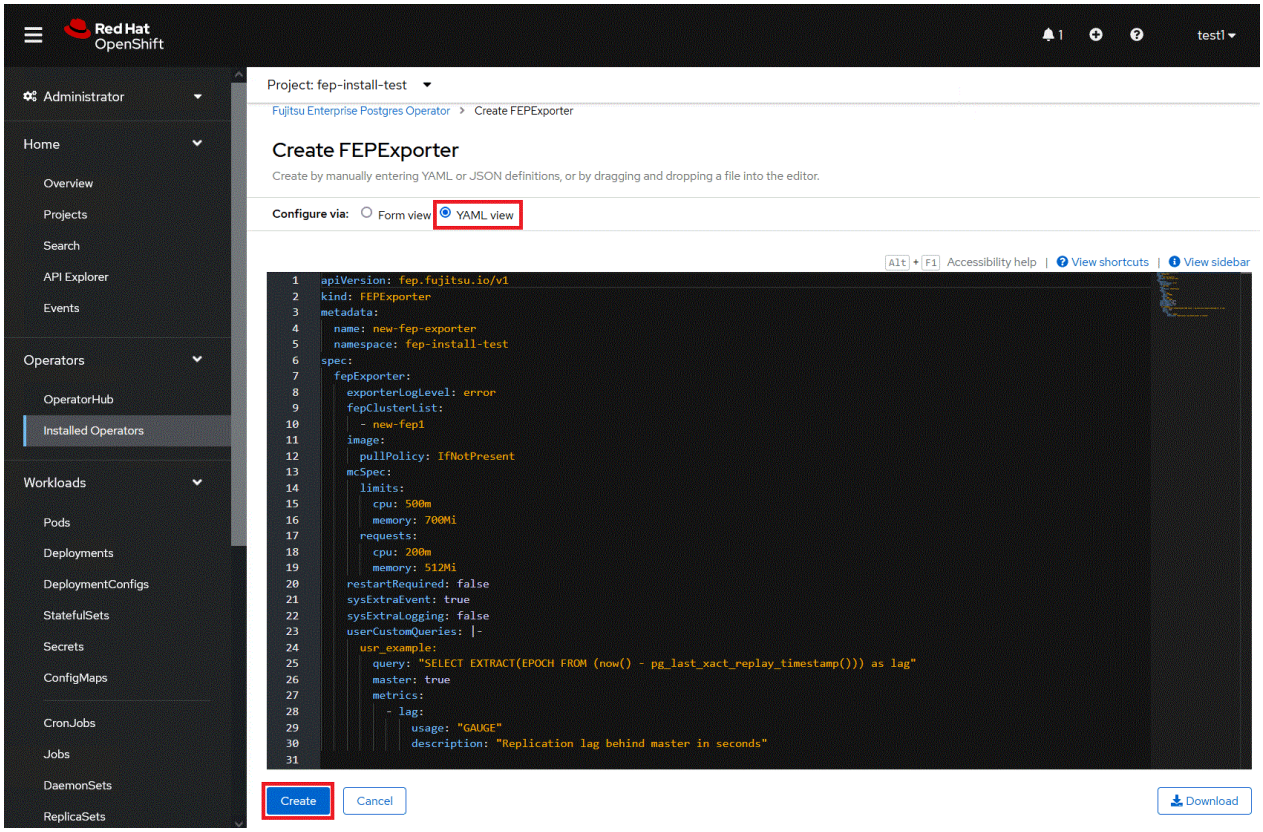
注意

Kubernetesクラスタ上へデプロイする場合は、“リファレンス”の“カスタムリソースパラメータ”を参照してyamlファイルを作成し、適用してください。

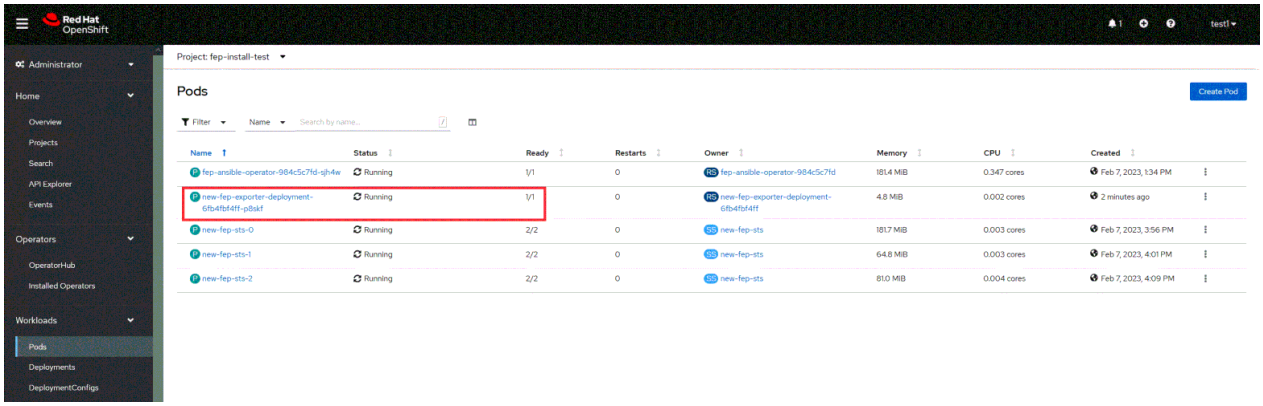
1. FEPExporter カスタムリソースを作成するには、次のいずれかを行います。
   (1) FEPExporter配下の[Create Instance]をクリックします。
   または
   (2) 上部の[FEPCluster]をクリックしてから、次のページの[Create FEPExporter]をクリックします。
2. [Form View]では、デプロイされているクラスタの名前のみが変更できます。デフォルト名は「new-fep-exporter」です。
   この名前は、名前空間内で一意である必要があります。
3. FEPExporterは、同じ名前空間内のFEPClusterのメトリクスをスクレイプします。



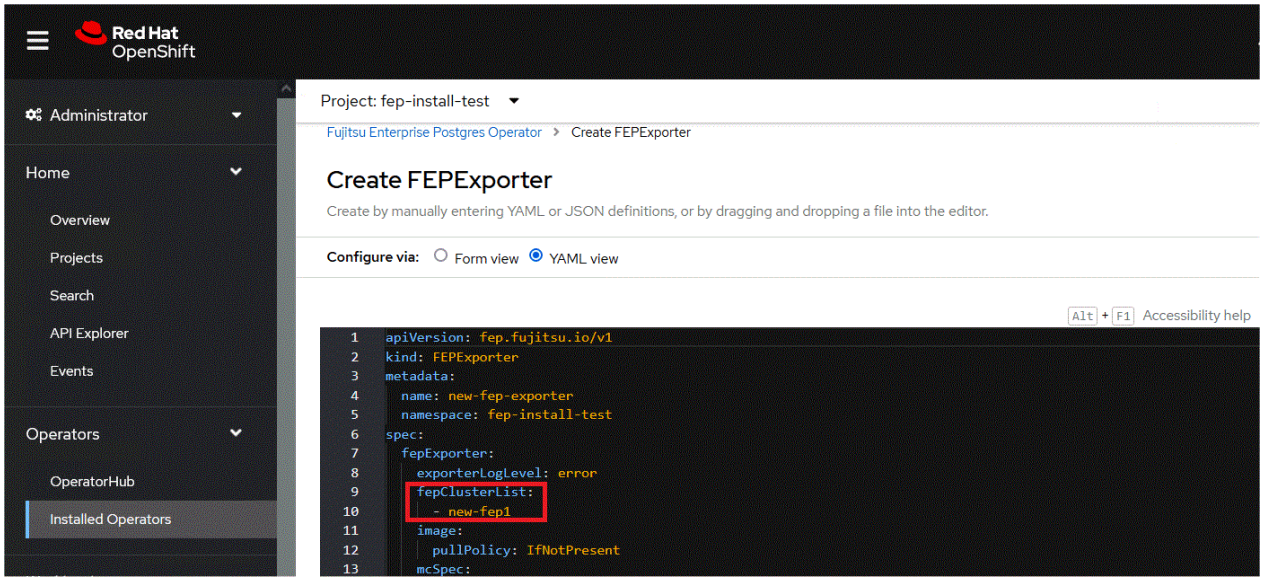
4. [YAML View]では、FEPExporterカスタムリソースの開始値が表示され、カスタムリソースを作成する前にパラメータを変更することができます。パラメータの詳細については、“リファレンス”を参照してください。



5. [Create]ボタンをクリックすると、シークレット、サービス、データソースクエリのconfigmapなどの他のリソースを使用してFEPExporter Podが作成されます。



6. FEPExporterのspec.fepExporter.fepClusterListでFEPClusterの名前を指定します。事前に、FEPClusterステータスとFEPStatefulSetが実行状態にあることを確認してください。



7. 新しいデータリソースシークレットを使用してFEPExporter Podを再作成します。監視セクションに、指定されたFEPclusterの統計を含むすべてのデータベースが一覧表示されます。
8. fepClusterListに複数のクラスタがリストされている場合、現在のエクスポータは、リストされているすべてのクラスタのメトリクスを収集します。
9. 複数のFEPExporterを、メトリクスを収集するための独自のクラスタリストを使用して1つの名前空間内にデプロイできます。

4.5 クラウドシークレット管理を用いたFEPクラスタのデプロイ

注意

クラウドシークレット管理機能は、下記パラメータとは併用できません。

- spec.fepChildCrVal.sysUsers.pgSecurityUser
- spec.fepChildCrVal.sysTde.tdek
- spec.fepChildCrVal.sysUsers.passwordValid.days

4.5.1 Helm Chartを利用したSecret Store CSI Driverのインストール

Helm ChartからSecret Store CSI Driverをインストールします。

Helm Chartのリポジトリを追加します。

```
helm repo add secrets-store-csi-driver https://kubernetes-sigs.github.io/secrets-store-csi-driver/charts
```

helmコマンドでインストールします。

```
helm install csi-secrets-store secrets-store-csi-driver/secrets-store-csi-driver --namespace kube-system --set
enableSecretRotation=true --set rotationPollInterval=30s
```

参考

- enableSecretRotation=trueは、シークレットの自動ローテーションを有効にします。外部のシークレットストア(Azure/AWS/GCP/HashiCorp vault)でシークレットの値が変更された場合、更新された値がFEPClusterに反映されます。

- rotationPollinterval = 30sは、すべてのpodに対して、マウントされたシークレットを最新の状態に変更する必要がある頻度をチェックするシークレットローテーションのポーリング間隔を30sに設定しています。
- OpenShiftクラスタでCSIタイプのボリュームをコンテナにマウントできるようにするには、システムの Security Context Constraintsにパッチを適用する必要があります。プロバイダーのCSIを含むボリュームセクションにパッチを適用します(nonroot,anyuid,hostmount-anyuid,machine-api-termination-handler,hostaccess,node-exporter,privileged,privileged-genevalogging,restricted)。
- 既存のOpenShiftをアップグレードする場合は、CSIが上記のシステムのSecurity Context Constraints に含まれていることを確認してください。

4.5.2 Azure Provider for Secret Store CSI Driverのインストールと設定

4.5.2.1 Helm Chartを利用したAzure Providerのインストール

```
helm repo add csi-secrets-store-provider-azure https://azure.github.io/secrets-store-csi-driver-provider-azure/charts
```

Azure Providerをインストールするときのデフォルトの設定では、secret-store-csi-driverのインストールは、true に設定されています。“4.5.1 Helm Chartを利用したSecret Store CSI Driverのインストール”の手順でsecret-store-csi-driverがすでにインストールされている場合は、以下のコマンドを実行します。

```
helm install csi csi-secrets-store-provider-azure/csi-secrets-store-provider-azure --namespace kube-system -set secrets-store-csi-driver.install=false
```

“4.5.1 Helm Chartを利用したSecret Store CSI Driverのインストール”の手順でsecret-store-csi-driverがインストールされていない場合は、以下のコマンドを実行して、secret-store-csi-driverとAzure Providerをインストールします。

```
helm install csi csi-secrets-store-provider-azure/csi-secrets-store-provider-azure --namespace kube-system --set secrets-store-csi-driver.enableSecretRotation=true --set secrets-store-csi-driver.rotationPollInterval=30s
```

4.5.2.2 Azure Key Vaultにアクセスするためのシークレットの作成

```
kind: Secret
apiVersion: v1
metadata:
 name: <Secret Name>
 namespace: <WHERE FEP CLUSTER TO BE INSTALLED>
 labels:
  secrets-store.csi.k8s.io/used: 'true'
data:
 clientid: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXx
 clientsecret: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX=
type: Opaque
```

Clientid: clientidは、SERVICE_PRINCIPAL_CLIENT_IDです。

Clientsecret: clientsecretは、SERVICE_PRINCIPAL_CLIENT_SECRETです。

4.5.2.3 Azure Key Vaultへのシークレットの保存

```
az keyvault secret set --vault-name <Vault Name> --name <Secret Name> --value <Secret value>
```

4.5.2.4 Azure Key Vaultへの証明書の保存

Azure Key Vault にアップロードする前に、証明書を以下の形式にします。1つの.pemファイル(1つのファイルにキー、crt、およびCAを含む)にする必要があります。

```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAx1rSsblocR8pROh5d2D3kuryTRRu6DA8axrSwrAaSDvdy1yU
KA7Q+Zg4IwaGwkt3cE2vK6oH4z3jwz+XOVjOxXo3hVh8tvfuXQOuNpFEWCRRX1xt
3S8xc8OCzbnHRWQAKdxRGWhfmPSdWdlpPe7uNcVe865TVOWLMAjYzZbMOJnFHmK3
```

```
5EoxRkcLs3sGi74YhwDsGa1sNzBhZpdR+iIheEZKJUc65dl13jKx9oDhc1c8lcwR
ecrVgfRo6NfZ86bkR2ImL5xROSWKnXP3KZqPOkL9DtCZK8iW2CgrfI8d2zcLbuUZ
UHEt4zzrwc9NV1yXe6nc8CrXbI6icwJYgVMZawIDAQABAoIBAF4kiNO/BpBtO8r7
OejLVP7/jr9Rx/JEXTPjLeaczTyRcPNJW/nyzUMhXFlGCruUceoJ9ZAOMpdgsb+R
t3s4aiUdyzxghjzNprYwtEM2pMTPGdJjzsomMD9P8+R9OBqP1/fswCuOe3i7A9fb
cPS7cajY9Tc0esvbvrhHZULpVLXhKl45SgDKgAWNaLJlm4u4gE56qpy+5kUKDzHg
yNOErpBSw2jlbtDE1Uta1hlR7BGWpK571UNvZ2AgLTbIgf1QFLq9IJdg9lll5pfm
DDn4AvcuFTHqJNj29DiMpsedvtPEnWceEkSScyzZnSvwJsADcdm2G8hyeeOsaQW+
/pVicfECgYEA7vADTIlWwOzcYH/CY+dOYAMaSOPO8IPi5PXFj5FJ44q8BwZUDHGI
gUZylxJfipBvca2zYbrNSJ1ynF6mup3OeeQDlVDSOdvcTg14OCuSZuvl/mG+1sBK
G5QiXE15D6IJj3Ngu3wu+RFK3CCQuveERAaWD1kZizRlOFiacV7lJBkCgYEA1Zcz
1YNlLybKXJbON3aFOhlz9RH1gNIx1PswJmDkM7qXlw5uxVpSPsvgngMsdAxMnSFQ
y5xxQY7fxUkv5ms6Po7c8BKyp2cLWRW2UH28ev8WT26yuml6OFXfv6XDhoF6CYeR
sGIlG9IUY2i4rkgajNYtyeE6r6O3LljoD7qNuiMCgYA55G94MOKTNhCjVPE9kYvx
426Qg/Op/tqPzTjD81jqx+eM8CyXIz8Gy5HiJrJ9eUd3TLXk3QT2Lifh2VEecDOW
93ciy4VUPYAgbUUzcwsy4r9EJly93bNXAUpeAOtvLTyRxEvQwWMEN/tiYIWQt34V
mV7scxMsVlKcF2O8SljMqQKBgBUgGV5a2pOpRwaVX55EuLSgY9mvZwrQv2EDXyXM
m4WKRQgJw2b9ofjYDWVThwgLV2CLNQSOepOzVmqa7IPrwxOA4FVWZBkule6/uQKJ
DSVVKY29syvA1vfPdovsBOS8daePoxdA/c6cnqueZfXG5+laHblD75wDo1CQNpOn
rfDlAoGBAJNI3q5XWGMciw8RcOOU2iWFSWWih9yHPpG3VGj2wUICDHdOoNvmYPik
PJMbemXI7fyUltthzx6TkY/8uvQpjNw1gLkKNSUQw/Fez8acA59jtvBnFy3ERDQD
+hsETWiHZ43QRo5fVOLjrUxurM9k/NTWzVBRov3yqc3XnVsgxujL
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIIFfjCCA2agAwIBAgIUCVqIwocAj7N/1NNCyLjporXLbE8wDQYJKoZIhvcNAQEL
BQAwVzEYMBYGA1UECgwPTXkgT3JnYW5pemFOaW9uMQswCQYDVQQLDAJDQTEuMCwG
A1UEAwwlTXkgT3JnYW5pemFOaW9uIENlcnRpZmljYXRlIEF1dGhvcmlOeTAeFw0y
MjEwMTMxMjE5MTBaFw0yMzEwMTMxMjE5MTBaMBMxETAPBgNVBAMMCHBvc3RncmVz
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAx1rSsblocR8pROh5d2D3
kuryTRRu6DA8axrSwrAaSDvdy1yUKA7Q+Zg4IwaGwkt3cE2vK6oH4z3jwz+XOVj0
xXo3hVh8tvfuXQOuNpFEWCRRX1xt3S8xc8OCzbnHRWQAKdxRGWhfmPSdWdlpPe7u
NcVe865TVOWLMAjYzZbMOJnFHmK35EoxRkcLs3sGi74YhwDsGa1sNzBhZpdR+iIh
eEZKJUc65dl13jKx9oDhc1c8lcwRecrVgfRo6NfZ86bkR2ImL5xROSWKnXP3KZqP
OkL9DtCZK8iW2CgrfI8d2zcLbuUZUHEt4zzrwc9NV1yXe6nc8CrXbI6icwJYgVMZ
awIDAQABo4IBhDCCAYAwggF8BgNVHREEggFzMIIBb4IKKi5ucy1hLnBvZIIYKi5u
cy1hLnBvZC5jbHVzdGVyLmxvY2FsghBuZjMzLXByaW1hcnktc3ZjghVuZjMzLXBy
aW1hcnktc3ZjLm5zLWGCGW5mMzMtcHJpbWFyeS1zdmMubnMtYS5zdmOCJ25mMzMt
cHJpbWFyeS1zdmMubnMtYS5zdmMuY2x1c3Rlci5sb2NhbIIQbmYzMy1yZXBsaWNh
LXN2Y4IVbmYzMy1yZXBsaWNhLXN2Yy5ucy1hghluZjMzLXJlcGxpY2Etc3ZjLm5z
LWEuc3ZjgiduZjMzLXJlcGxpY2Etc3ZjLm5zLWEuc3ZjLmNsdXNOZXIubG9jYWyC
HG5mMzMtc3RzLTAubmYzMy1oZWFkbGVzcy1zdmOCPHB1Ymxpc2hlci1ob3NOLW5h
bWUubmFtZXNwYWNlLW9mLXB1Ymxpc2hlci5zdmMuY2x1c3Rlci5sb2NhbIIRbmYz
My1oZWFkbGVzcy1zdmMwDQYJKoZIhvcNAQELBQADggIBACBWl1DVvZj6kO5SSGpv
jXCCRu6jhWBaXH9jTH9Awg6DxXU6BzOATpCFMEcMP4Bv+1lG/2Gkz8p7PSfznsr9
LWK2ACuQ9FettgPZyQaHtV8e5AHCtCNK9WeSKoZ2XGIAKPJu3DZ7LZODP7lqinPC
T/cxY+4Qbtuga+gHoLKfOiATlM7OsbRIpI5q4EosZtmp+dv8l1kHVZMLusDLhhV7
QYHhW1rJfpBEaUdrFaqUB+6Eo/MY3hbUzYMcGdae83KA1rW2/owL7E6pL8aJPhX9
igCT/XVwuIH3aaYkwDlOLZzU/ga8KOrs2cbEcHFBOtnNzs81hVebZmqV/GqmVTbD
ty8+IbU3miKa2/bDbmZBMWyvdVo52W1h62AZtGF93JvoaZVAAp53v3Gv6rs64lj2
7iP3CVLBs/OBFBG7y6q6/yOjlNEa4D9vOpPS3uBGSQDMpKG7mRIYksmOwULDBYI3
UjZpwVJjRuVY7N6ONgvZxOfC5HKb2Djb/u8RL8UrMmqqZlKNdh/O6OZIZEX63esb
yHzQbiYSnop6LgpK5STtizJlaTpxkVcrJ2tzHuWp1PcCpShRTuKU+LFlOmOUMYk9
6Oi5h9GDTURDSOOOO8RosiJd+locEBiKwZIA6dh98c+dd4eml9F+Pt3OlZA/wgcu
NwROKO5YLzFxBStiz2kiUOdZ
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
MIIFXzCCAOegAwIBAgIURO14D/Pjf9/VIxF+jYFV1MtKnpQwDQYJKoZIhvcNAQEL
BQAwVzEYMBYGA1UECgwPTXkgT3JnYW5pemFOaW9uMQswCQYDVQQLDAJDQTEuMCwG
A1UEAwwlTXkgT3JnYW5pemFOaW9uIENlcnRpZmljYXRlIEF1dGhvcmlOeTAeFw0y
MjEwMTMxMjE5MTBaFw0zMjEwMTAxMjE5MTBaMFcxGDAWBgNVBAoMD015IE9yZ2Fu
aXphdGlvbjELMAkGA1UECwwCQOExLjAsBgNVBAMMJU15IE9yZ2FuaXphdGlvbiBD
ZXJOaWZpY2FOZSBBdXRob3JpdHkwggIiMAOGCSqGSIb3DQEBAQUAA4ICDwAwggIK
AoICAQDOh1UQb6QLw/e1J2gar/eRf6W4PhkNpOKGmdS5RmOJ58sDEwb/BNaBRYzu
```

```
e05mLQ7R3YF3I83AZf19EOss36tfi9puRaCr5toC/XaBqK1zLPSZmZVtlxadZSFG
9+3WB8IXrDuSQw1cZi9oosOJeq962dPDqd56qicnEk7r8Vpd5ycYuadEclPDX7ne
zw6A6eHfIaAw9ETFOt1Ph88Yh3XhO+e937YOZOucpxJIXqxdGbK9yFgk4y4Pbjg7
yXWcFP1Cg2FKN/Odhr3k64WNDcqejpxbfJgxAtujg7lFjg/YuzbbMRjCzB1TZGPU
iM7TKPPw9PVoWKJ3siR5SoxJp5LgdkhvT83zx3zw87htjbcbnYPOy+F2PX88U5be
UpYzIcRjBPh59AYgfGJaBjTm5dy8ryWQ9diwAklxvnTwa7c443xG3IFHq5/Yt7ol
sbT1h5gp3hHfh/WvZxFagirX66Uz5TY2FDzWVsQHvoIGMHD8hcr7Reia8IPFnneW
zRE1lNPQNXhqgcOpflg/6u8FCMdEeR/QV1lsjavVEMXoJUOPEx+srhUg+4gVlzc1
7OPG/ThJOdzXCeEEaI8Z6Yq5I3PJiEUvbWhEGOQ/S9pJeIlBwCsADGlVaAOXy+gy
5Hh8dTrWg+TwI8lpWQSWXJGIpY684/jLVFu16U5aawgacrmExwIDAQABoyMwITAP
BgNVHRMBAf8EBTADAQH/MA4GA1UdDwEB/wQEAwIBhjANBgkqhkiG9w0BAQsFAAOC
AgEAujV7RkipqwNopqW4kwbIbF4mn3JPBzbzKSjr8uraCFpk3ZTiRsiHm3DO7/ox
N7KTqbk+DhSdbZ1NM+f1kZ7zDR6r4KGgBmKID51DOJ54jxNuCwRKndGUfePATuDO
yaLsOU1YAUO2/S6cWKki1wEHv+t+p9zlJORd75M4GIKdQOyOtyEsiMPEbP3OqfJt
PJ7R+WBGvedt3TPEi3REubzUOMhgsDHuqeKKVBuRdh3zvcSI1q59DKYUir7wY6Oy
3fwJtEkrpyBD57Tp/VsafOTxv9KTtbyiCYOnwmiN3RqyFx4lIEipT1dhVc2oBUFq
YWvTkUPubFbGOaLxcbi5aySCOmjZHYZvUCNLSAekTL2wH649/RD8xSkQf+Qs2N6a
jJOE1nUrapYRrKlwFRXj+5aj+fhhoZ1uU43jPRakdwinEWmw7JPRkOgjRQwQE6a6
bhBvBfStOZKmuOULuoHrL75BCyQMK5JaOgljmcsAQMbO/ERpPxoNzkXAS825wOTx
E+lnRRuOKfmILIHMteOpn+ffozT2Djl3mFMJhbbbnYEL1NEYxwI2si2oL8GjE26i
A5ojkdJO6kmFgOp2boa49ja61lWVZToirWhbnR6G9AKHPy8aXOyH25xStxbdojjO
eTP+zKBUH3E15zTOYOnb7NnIplHNNhq1kwi/OCBXPs9FWow=
-----END CERTIFICATE-----
```

mycert.pem

```
az keyvault secret set --vault-name <Key Vault Name> --name <Secret Name> --file "mycert.pem"
```

注意

Key Vaultに格納されるシークレットのキー値は1つだけです。

4.5.3 AWS Provider for Secret Store CSI Driverのインストールと設定

4.5.3.1 Helm Chartを利用したAWS Providerドライバのインストール

```
helm repo add aws-secrets-manager https://aws.github.io/secrets-store-csi-driver-provider-aws
```

```
helm install -n kube-system secrets-provider-aws aws-secrets-manager/secrets-store-csi-driver-provider-aws -namespace
kube-system
```

4.5.3.2 IAMロールとSecret Managerのアクセス権限を持つサービスアカウントでのEKSクラスタのセットアップ

以下を参照し、CSIのIAM ロールとEKS をセットアップします。

https://github.com/aws/secrets-store-csi-driver-provider-aws

Secret ManagerにアクセスするためのIAMロールの信頼ポリシーを作成します。

```
Create IAM role trust policy to access Secret Manager
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
```

```
                "Federated": "arn:aws:iam::123456789:oidc-provider/oidc.eks.ap-southeast-3.amazonaws.com/id/ABCD1234567"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    " oidc.eks.ap-southeast-3.amazonaws.com /id/ ABCD1234567:sub": "system:serviceaccount:myns:mysa",
                    " oidc.eks.ap-southeast-3.amazonaws.com /id/ ABCD1234567:aud": "sts.amazonaws.com"
                }
            }
        }
    ]
}
```

4.5.3.3 AWS Secret Managerへのシークレットの保存

```
aws secretsmanager create-secret --name <Secret Name> --secret-string <Secret Value>
```

4.5.3.4 AWS Secret Managerへの証明書の保存

AWS Secrets Manageに証明書をアップロードする前に、証明書を以下の形式にします。1つの.pemファイル(1つのファイルにキー、crt、および CAを含む)にする必要があります。

(証明書の形式のサンプルについては、“mycert.pem”を参照)

```
aws secretsmanager create-secret --name <Secret Name> --secret-binary fileb://<File Name>
```

注意

Secret Managerに格納されるシークレットのキー値は1つだけです。

4.5.4 GCP Provider for Secret Store CSI Driverのインストール

4.5.4.1 Kubernetesを利用したGCP Providerドライバのインストール

```
wget https://raw.githubusercontent.com/GoogleCloudPlatform/secrets-store-csi-driver-provider-gcp/main/deploy/provider-gcp-plugin.yaml
```

```
kubectl apply -f provider-gcp-plugin.yaml -namespace kube-system
```

4.5.4.2 GCP Secret ManagerとIAMの設定

サービスアカウントを作成します。

```
gcloud iam service-accounts create my-secret-acc;
```

SecretManagerAdminポリシーを新しいサービスアカウントにアタッチします。

```
gcloud projects add-iam-policy-binding $PROJECT_ID ¥
--member="serviceAccount: my-secret-acc @$PROJECT_ID.iam.gserviceaccount.com" ¥
--role="roles/secretmanager.admin" ¥
--condition="None";
```

新しいサービス アカウントのキーを生成します。

```
gcloud iam service-accounts keys create iam-key.json ¥
--iam-account=" my-secret-acc @$PROJECT_ID.iam.gserviceaccount.com";
```

4.5.4.3 GCP Secret Managerにアクセスするためのシークレットの作成

“4.5.4.2 GCP Secret ManagerとIAMの設定”で生成されたキー(iam-key.json file)を使用します。

```
kubectl create secret generic <secret-name> --from-file=<iam-key.json>
```

4.5.4.4 GCP Secret Managerへのシークレットの保存

```
gcloud secrets create <secret name> --data-file="/path/to/file"
```

4.5.4.5 GCP Secret Managerへの証明書の保存

GCP Secret Managerに証明書をアップロードする前に、証明書を以下の形式にします。1つの.pemファイル(1つのファイルにキー、crt、および CAを含む)にする必要があります。

(証明書の形式のサンプルについては、“mycert.pem”を参照)

```
gcloud secrets create <secret name> --data-file="/path/to/file"
```

注意

Secret Managerに格納されるシークレットのキー値は1つだけです。

4.5.5 HashiCorp Vault Provider for Secret Store CSI Driverのインストール

4.5.5.1 Helm Chartを利用したHashiCorp Providerドライバのインストール

```
helm repo add hashicorp https://helm.releases.hashicorp.com
```

```
helm install vault hashicorp/vault --set "server.enabled=false"  --set "injector.enabled=false" --set "csi.enabled=true"
```

4.5.5.2 HashiCorp VaultのKubernetesの認証の設定

```
vault auth enable kubernetes
```

```
vault write auth/kubernetes/config ¥
token_reviewer_jwt="$(cat /var/run/secrets/kubernetes.io/serviceaccount/token)" ¥
kubernetes_host="https://$KUBERNETES_PORT_443_TCP_ADDR:443" ¥
kubernetes_ca_cert=@/var/run/secrets/kubernetes.io/serviceaccount/ca.crt
```

4.5.5.3 HashiCorp Vaultへのシークレットの格納

```
vault enable secret
```

```
vault kv put secret/<path> <secret name>=<secret value>
```

4.5.5.4 HashiCorp Vaultへの証明書の格納

HashiCorp Vaultに証明書をアップロードする前に、証明書を以下の形式にします。1つの.pemファイル(1つのファイルにキー、crt、および CAを含む)にする必要があります。

(証明書の形式のサンプルについては、“mycert.pem”を参照)

```
Vault kv put secret/<path> <secret name>=@<path to cert.pem>
```

4.5.5.5 HashiCorp Vaultからシークレットにアクセスするためのポリシーとロールの作成

ポリシー:

```
vault policy write <policy name> - <<EOF
path "secret/database/credentials" {
capabilities = ["read", "write", "update","delete"]
}
EOF
```

ロール:

```
vault write auth/kubernetes/role/<role name> ¥
bound_service_account_names=* ¥
bound_service_account_namespaces=*¥
policies=<policy name> ¥
ttl=24h
```

注意: <fep-cluster>-sa サービス アカウントを bound_service_account_names に割り当てることでアクセスを制限でき、bound_service_account_namespaces に値を割り当てることで名前空間を制限することもできます。

注意

HashiCorp Vaultに格納されるシークレットのキー値は1つだけです。

4.5.6 Provider for Secret Store Driverを使用するためのFEPClusterの設定

Secret Store CSI Driverの使用を有効にするために、FEPCluster CR の spec.fepChildCrVal セクション配下にあるパラメータ“secretStore”を設定します。secretStore.csiの配下に、外部のシークレットストア (Azure、AWS、GCP、および HashiCorp Vault) に接続するための詳細と、そのシークレットストア内のシークレットのリストを定義する必要があります。spec.fepChildCrVal.secretStore パラメータの定義は、使用されるプロバイダのタイプによって異なります。

4.5.6.1 Azure Provider for Secret Store CSI Driver

```
spec:
  ….
  fepChildCrVal:
    secretStore:
      method: csi
      csi:
        providerName: azure
        azureProvider:
          keyvaultname:
          tenantid:
          credentials:
          fepSecrets:
          - pgadminpassword: pgadminpassword
          - tdepassphrase: passphrase
          - systemCertificates: systemCerts
          - pguser: pgusername
          - pgpassword: pgpwd
          - pgdb: pgdbsecret
          - pgrepluser: pgrepluser
          - pgreplpassword: pgreplpassword
          - pgRewinduser: pgRewinduser
          - pgRewindpassword: pgRewindpassword
          - pgMetricsUser: metricsuser
          - pgMetricsPassword: metricspwd
          - patronitls: patronicrt
          - patronitlscacrt: patronica
          - postgrestls: postgrescrt
```

```
        - postgrestlscacrt: postgresca
        - pgAdminTls: admincrt
        - pgAdminTlscacrt: adminca
        - pgAdminTls_privateKeyPassword: adminpvtkey
        - pgRewindUserTls: rewindcrt
        - pgRewindUserTlscacrt: rewindca
        - pgRewindUserTls_privateKeyPassword: rwndpvtkey
        - pgrepluserTls: replcrt
        - pgrepluserTlscacrt: replca
        - pgrepluserTls_privateKeyPassword: replpvtkey
        - pgMetricsUserTls: metricscrt
        - pgMetricsUserTlscacrt: metricsca
        - pgMetricsUserTls_privateKeyPassword: adminpvtkey
        fepCustomCerts:
          - userName:user1
            userCrt: user1crt
            userCa: user1ca
          - userName: mydbuser
            userCrt: mydbusercrt
            userCa: mydbuserca
```

注: fepSecretsで黒字の部分のパラメータは必須です。

4.5.6.2 AWS Provider for Secret Store CSI Driver

```
spec:
  ….
  fepChildCrVal:
    secretStore:
      method: csi
      csi:
        providerName: aws
        awsProvider:
          region:
          roleName:
          fepSecrets:
          - pgadminpassword: pgadminpassword
          - tdepassphrase: passphrase
          - systemCertificates: systemCerts
          - pguser: pgusername
          - pgpassword: pgpwd
          - pgdb: pgdbsecret
          - pgrepluser: pgrepluser
          - pgreplpassword: pgreplpassword
          - pgRewinduser: pgRewinduser
          - pgRewindpassword: pgRewindpassword
          - pgMetricsUser: metricsuser
          - pgMetricsPassword: metricspwd
          - patronitls: patronicrt
          - patronitlscacrt: patronica
          - postgrestls: postgrescrt
          - postgrestlscacrt: postgresca
          - pgAdminTls: admincrt
          - pgAdminTlscacrt: adminca
          - pgAdminTls_privateKeyPassword: adminpvtkey
          - pgRewindUserTls: rewindcrt
          - pgRewindUserTlscacrt: rewindca
          - pgRewindUserTls_privateKeyPassword: rwndpvtkey
          - pgrepluserTls: replcrt
          - pgrepluserTlscacrt: replca
          - pgrepluserTls_privateKeyPassword: replpvtkey
          - pgMetricsUserTls: metricscrt
          - pgMetricsUserTlscacrt: metricsca
```

```
        - pgMetricsUserTls_privateKeyPassword: adminpvtkey
        fepCustomCerts:
          - userName:user1
            userCrt: user1crt
            userCa: user1ca
          - userName: mydbuser
            userCrt: mydbusercrt
            userCa: mydbuserca
```

注: fepSecretsで黒字の部分のパラメータは必須です。

4.5.6.3 GCP Provider for Secret Store CSI Driver

```
spec:
  ……
  fepChildCrVal:
    secretStore:
      method: csi
      csi:
        providerName: gcp
        gcpProvider:
          credentials:
          fepSecrets:
          - pgadminpassword: pgadminpassword
          - tdepassphrase: passphrase
          - systemCertificates: systemCerts
          - pguser: pgusername
          - pgpassword: pgpwd
          - pgdb: pgdbsecret
          - pgrepluser: pgrepluser
          - pgreplpassword: pgreplpassword
          - pgRewinduser: pgRewinduser
          - pgRewindpassword: pgRewindpassword
          - pgMetricsUser: metricsuser
          - pgMetricsPassword: metricspwd
          - patronitls: patronicrt
          - patronitlscacrt: patronica
          - postgrestls: postgrescrt
          - postgrestlscacrt: postgresca
          - pgAdminTls: admincrt
          - pgAdminTlscacrt: adminca
          - pgAdminTls_privateKeyPassword: adminpvtkey
          - pgRewindUserTls: rewindcrt
          - pgRewindUserTlscacrt: rewindca
          - pgRewindUserTls_privateKeyPassword: rwndpvtkey
          - pgrepluserTls: replcrt
          - pgrepluserTlscacrt: replca
          - pgrepluserTls_privateKeyPassword: replpvtkey
          - pgMetricsUserTls: metricscrt
          - pgMetricsUserTlscacrt: metricsca
          - pgMetricsUserTls_privateKeyPassword: adminpvtkey
          fepCustomCerts:
            - userName:user1
              userCrt: user1crt
              userCa: user1ca
            - userName: mydbuser
              userCrt: mydbusercrt
              userCa: mydbuserca
```

注: fepSecretsで黒字の部分のパラメータは必須です。

4.5.6.4 HashiCorp Vault Provider for Secret Store CSI Driver

```
spec:
  ....
  fepChildCrVal:
    secretStore:
      method: csi
      csi:
        providerName: vault
        vaultProvider:
          roleName: "database"
          vaultAddress: "http://vault-url-addr:8765"
          fepSecrets:
          - pgadminpassword: pgadminpassword
          - tdepassphrase: passphrase
          - systemCertificates: systemCerts
          - pguser: pgusername
          - pgpassword: pgpwd
          - pgdb: pgdbsecret
          - pgrepluser: pgrepluser
          - pgreplpassword: pgreplpassword
          - pgRewinduser: pgRewinduser
          - pgRewindpassword: pgRewindpassword
          - pgMetricsUser: metricsuser
          - pgMetricsPassword: metricspwd
          - patronitls: patronicrt
          - patronitlscacrt: patronica
          - postgrestls: postgrescrt
          - postgrestlscacrt: postgresca
          - pgAdminTls: admincrt
          - pgAdminTlscacrt: adminca
          - pgAdminTls_privateKeyPassword: adminpvtkey
          - pgRewindUserTls: rewindcrt
          - pgRewindUserTlscacrt: rewindca
          - pgRewindUserTls_privateKeyPassword: rwndpvtkey
          - pgrepluserTls: replcrt
          - pgrepluserTlscacrt: replca
          - pgrepluserTls_privateKeyPassword: replpvtkey
          - pgMetricsUserTls: metricscrt
          - pgMetricsUserTlscacrt: metricsca
          - pgMetricsUserTls_privateKeyPassword: adminpvtkey
          fepCustomCerts:
            - userName:user1
              userCrt: user1crt
              userCa: user1ca
            - userName: mydbuser
              userCrt: mydbusercrt
              userCa: mydbuserca
```

注: fepSecretsで黒字の部分のパラメータは必須です。

4.6 カスタマイズしたFEPサーバコンテナイメージのデプロイ

4.6.1 前提条件

以下に記載されている手順では、dockerコマンドを使用してコンテナーイメージをビルドすることを前提としています。podmanなどの代替ツールを使用してコンテナイメージを構築することは、このドキュメントの範囲外です。

4.6.2 拡張機能によるカスタマイズしたFEPイメージのビルド

カスタマイズした新しいFEPサーバコンテナイメージをビルドする前に、イメージ固有のいくつかのビルド手順を理解することが重要です。

- FEPサーバコンテナイメージは、UBI8/UBI9 最小イメージであるubi-minimal の上に構築されます。
- USERのデフォルトは26です。

UBI8/UBI9 の最小イメージは、パッケージマネージャーとしてmicrodnfを使用します。Microdnfは、リモートURLまたはローカル ファイルからのRPMパッケージのインストールをサポートしていません。YUMリポジトリからのみインストールできます。YUMリポジトリにないRPMパッケージをインストールする場合は、パッケージをダウンロードしてrpmでインストールします。ただし、rpm には依存関係が解決されないという欠点があります。この問題を解決する唯一の方法は、最初にdnf をインストールし、dnfを使用してリモートURL またはローカルファイルからパッケージをインストールすることです。

USERはデフォルトで26であるため、RPMパッケージをインストールしたり、/usr/bin、/usr/local/bin などのシステムディレクトリにファイルを書き込んだりする権限がありません。この問題を回避するには、最初にUSERを root に設定して、カスタマイズを続行し、26に戻します。

```
FROM: quay.io/fujitsu/fujitsu-enterprise-postgres-15-server:ubi8-15-1.0

USER root

RUN …(customization)

USER 26
```

4.6.3 SQLite外部データラッパーをFEPサーバコンテナに追加

SQLite外部データラッパーモジュールをFEPサーバコンテナに追加し、カスタマイズする例を説明します。

1. Dockerfile を作成します。

```
#use FEP 15 image as a base to compile sqlite_fdw
FROM quay.io/fujitsu/fujitsu-enterprise-postgres-15-server:ubi8-15-1.0 as compile-sqlite_fdw

#change the user with root privilege
USER root

# install build tools
RUN microdnf -y install cmake gcc-c++ libtool clang which openssl-devel git llvm gettext

# Install sqlite_fdw build require
RUN microdnf install -y sqlite-devel

# Download sqlite_fdw source
RUN curl -sSL https://github.com/pgspider/sqlite_fdw/archive/refs/tags/v2.3.0.tar.gz | tar -zxf -

# Compile sqlite_fdw
RUN cd /sqlite_fdw-2.3.0 && ¥
    make install USE_PGXS=1

#Use base image is from FEPContainer to build the custom image
FROM quay.io/fujitsu/fujitsu-enterprise-postgres-15-server:ubi8-15-1.0

#change the user with root privilege
USER root

#copy the prepared OSS extension binaries to FEP server lib folder
COPY --from=compile-sqlite_fdw /opt/fsepv15server64/lib/sqlite_fdw.so /opt/fsepv15server64/lib/
COPY --from=compile-sqlite_fdw /opt/fsepv15server64/lib/bitcode/sqlite_fdw /opt/fsepv15server64/lib/bitcode/
COPY --from=compile-sqlite_fdw /opt/fsepv15server64/share/extension/sqlite_fdw* /opt/fsepv15server64/share/
extension/

# Install sqlite_fdw run time dependencies
RUN microdnf install -y sqlite-libs

#change the user to postgresql
USER 26
```

2. カスタムイメージをビルドします。

```
docker build -f Dockerfile -t my.registry/my-repo/fep-15-server-sqlite_fdw:ubi8-15-1.0
```

3. カスタムコンテナレジストリにイメージをpushします。

```
docker push my.registry/my-repo/fep-15-server-sqlite_fdw:ubi8-15-1.0
```

4.6.4 カスタムイメージでFEPクラスタを作成する

カスタムコンテナレジストリで認証が必要な場合は、quay-pull-secretという名前のプルシークレットを作成します。FEPオペレーターは、このプルシークレットを使用してコンテナイメージをダウンロードします。

```
kind:Secret
apiVersion:v1
metadata:
  name:quay-pull-secret
  namespace:fep-container-ct
data:
  .dockerconfigjson:~>-
    xxxxxxbCI6ICIiCiAgICB9CiAgfQp9
type:kubernetes.io/dockerconfigjson
```

FEPClusterカスタムリソースを作成します。

```
apiVersion: fep.fujitsu.io/v2
kind: FEPCluster
metadata:
  name: sqlite-fdw
  namespace: fep-container-ct
spec:
  fep:
    forceSsl: true
... ...
    image:
      image: 'my.registry/my-repo/fep-15-server-sqlite_fdw:ubi8-15-1.0'
... ...
    instances: 1
```

FEPClusterをデプロイします。

```
oc apply -f sqlite_fdw.yaml
```

拡張機能をインストールします。

```
postgres# CREATE EXTENSION sqlite_fdw;
CREATE EXTENSION
postgres=#
```

4.7 MTLSを実行するためのFEPの設定

以下の3つのトラフィックはすべて証明書で保護されたTLS接続を使用して保護できます。

- クライアントアプリケーションからFEPClusterへのPostgresトラフィック
- FEPCluster内のPatroniRESTAPI

- FEPCluster内のPostgresトラフィック（レプリケーション、巻き戻しなど）

ここでは、TLS接続を保護するための証明書を作成し、相互認証を提供する2つの方法を提供します。1つは、証明書を手動で作成および更新する方法です。もう1つは、cert-managerを使用して自動で更新される証明書を作成する方法です。

注意

MTLS構成のデータベースクラスタへのクライアント接続には、以下が適用されます。

- MTLSデータベースクラスタの作成時に指定したサーバ(検証) のルート証明書をクライアントマシンに配布します。
- 新しいクライアント証明書を作成して使用します。
- サーバールート証明書とクライアントルート証明書が異なる場合は、サーバー側の構成の更新が必要です。

4.7.1 証明書を手動で管理する方法

作業の概要

MTLS通信を有効にする手順を以下に示します。

1. CAとして自己署名証明書を作成します
2. CA証明書を保存するためのConfigmapを作成します
3. FEPサーバの秘密鍵を保護するためのパスワードを作成します（オプション）
4. FEPサーバの秘密鍵を作成します
5. FEPサーバ証明書署名要求を作成します
6. CAによって署名されたFEPサーバ証明書を作成します
7. FEPサーバの証明書とキーを保存するためのTLSシークレットを作成します
8. Patroniの秘密鍵を作成します
9. Patroniの証明書署名要求を作成します
10. Patroni用にCAによって署名された証明書を作成します
11. Patroni証明書とキーを保存するためのTLSシークレットを作成します
12. “postgres”ユーザーのクライアント証明書の秘密鍵を作成します
13. “postgres”ユーザーのクライアント証明書の証明書署名要求を作成します
14. “postgres”ユーザーのクライアント証明書を作成します
15. “postgres”証明書とキーを保存するためのTLSシークレットを作成します
16. “repluser”と“rewinduser”に対して手順12-15を繰り返します

注意

- マニュアルに記載されている情報は一例であり、動作時には、ユーザーが信頼できる認証局 (CA) によって署名された証明書を使用してください。
- Kubernetesクラスタ上で操作する場合は、ocコマンドをkubectlコマンドに読み替えてください。

CA証明書の作成

1. CAとして自己署名証明書を作成します

```
openssl genrsa -aes256 -out myca.key 4096
Generating RSA private key, 4096 bit long modulus (2 primes)
.....................++++
.....................................................++++
e is 65537 (0x010001)
Enter pass phrase for myca.key: 0okm9ijn8uhb7ygv
Verifying - Enter pass phrase for myca.key: 0okm9ijn8uhb7ygv

cat << EOF > ca.cnf
[req]
distinguished_name=req_distinguished_name
x509_extensions=v3_ca
[v3_ca]
basicConstraints = critical, CA:true
keyUsage=critical,keyCertSign,digitalSignature,cRLSign
[req_distinguished_name]
commonName=Common Name
EOF

openssl req -x509 -new -nodes -key myca.key -days 3650 -out myca.pem -subj "/O=My Organization/OU=CA /CN=My
Organization Certificate Authority" -config ca.cnf
Enter pass phrase for myca.key: 0okm9ijn8uhb7ygv
```

2. CA証明書を保存するためのConfigmapを作成します

```
oc create configmap cacert —from-file=ca.crt=myca.pem -n my-namespace
```

3. FEPサーバの秘密鍵を保護するためのパスワードを作成します(オプション)

```
oc create secret generic mydb-fep-private-key-password —from-literal=keypassword=abcdefghijk -n my-namespace
```

サーバ証明書の作成

4. FEPサーバの秘密鍵を作成します

```
openssl genrsa -aes256 -out fep.key 2048
Generating RSA private key, 2048 bit long modulus
.............................................+++
.......+++
e is 65537 (0x10001)
Enter pass phrase for fep.key: abcdefghijk
Verifying - Enter pass phrase for fep.key: abcdefghijk
```

5. FEPサーバ証明書署名要求を作成します

```
cat << EOF > san.cnf
[SAN]
subjectAltName = @alt_names
[alt_names]
DNS.1 = *.my-namespace.pod
DNS.2 =  *.my-namespace.pod.cluster.local
DNS.3 = mydb-primary-svc
DNS.4 = mydb-primary-svc.my-namespace
DNS.5 = mydb-primary-svc.my-namespace.svc
DNS.6 = mydb-primary-svc.my-namespace.svc.cluster.local
```

```
DNS.7 = mydb-replica-svc
DNS.8 = mydb-replica-svc.my-namespace
DNS.9 = mydb-replica-svc.my-namespace.svc
DNS.10 = mydb-replica-svc.my-namespace.svc.cluster.local
EOF

$ openssl req -new -key fep.key -out fep.csr -subj "/CN=mydb-headless-svc" -reqexts SAN -config <(cat /etc/pki/tls/
openssl.cnf <(cat san.cnf))
Enter pass phrase for fep.key: abcdefghijk
```

注意

- クラスタ名と名前空間を適切に変更する必要があります。
- OCPクラスタの外部から接続する場合は、その接続に使用するホスト名も含める必要があります。

6. CAによって署名されたFEPサーバ証明書を作成します

```
openssl x509 -req -in fep.csr -CA myca.pem -CAkey myca.key -out fep.pem -days 365 -extfile <(cat /etc/pki/tls/
openssl.cnf <(cat san.cnf)) -extensions SAN -CAcreateserial  # all in one line
Signature ok
subject=/CN=mydb-headless-svc
Getting CA Private Key
Enter pass phrase for myca.key: 0okm9ijn8uhb7ygv
```

7. FEPサーバの証明書とキーを保存するためのTLSシークレットを作成します

```
oc create secret generic mydb-fep-cert --from-file=tls.crt=fep.pem --from-file=tls.key=fep.key -n my-namespace
```

8. Patroniの秘密鍵を作成します

現時点では、FEPコンテナはPatroniのパスワードで保護された秘密鍵をサポートしていません。

```
openssl genrsa -out patroni.key 2048
Generating RSA private key, 2048 bit long modulus
.............................................+++
.......+++
e is 65537 (0x10001)
```

9. Patroniの証明書署名要求を作成します

```
cat << EOF > san.cnf
[SAN]
subjectAltName = @alt_names
[alt_names]
DNS.1 = *.my-namespace.pod
DNS.2 = *.my-namespace.pod.cluster.local
DNS.3 = mydb-primary-svc
DNS.4 = mydb-primary-svc.my-namespace
DNS.5 = mydb-replica-svc
DNS.6 = mydb-replica-svc.my-namespace
DNS.7 = mydb-headless-svc
DNS.8 = mydb-headless-svc.my-namespace
EOF

openssl req -new -key patroni.key -out patroni.csr -subj "/CN=mydb-headless-svc" -reqexts SAN -config
<(cat /etc/pki/tls/openssl.cnf <(cat san.cnf))  # all in one line
```

10. Patroni用にCAによって署名された証明書を作成します

```
openssl x509 -req -in patroni.csr -CA myca.pem -CAkey myca.key -out patroni.pem -days 365 -extfile <(cat /etc/pki/tls/
openssl.cnf <(cat san.cnf)) -extensions SAN -CAcreateserial  # all in one line
Signature ok
subject=/CN=mydb-headless-svc
Getting CA Private Key
Enter pass phrase for myca.key: 0okm9ijn8uhb7ygv
```

11. Patroni証明書とキーを保存するためのTLSシークレットを作成します

```
oc create secret tls mydb-patroni-cert --cert=patroni.pem --key=patroni.key -n my-namespace
```

ユーザー証明書の作成

12. postgresユーザーのクライアント証明書の秘密鍵を作成します

    現時点では、FEPサーバコンテナ内のSQLクライアントはパスワードで保護された証明書をサポートしていません。

```
openssl genrsa -out postgres.key 2048
Generating RSA private key, 2048 bit long modulus
..............................................+++
.......+++
e is 65537 (0x10001)
```

13. postgresユーザーのクライアント証明書の証明書署名要求を作成します

```
openssl req -new -key postgres.key -out postgres.csr -subj "/CN=postgres"
```

14. postgresユーザーのクライアント証明書を作成します

```
$ openssl x509 -req -in postgres.csr -CA myca.pem -CAkey myca.key -out postgres.pem -days 365
Signature ok
subject=CN = postgres
Getting CA Private Key
Enter pass phrase for myca.key: 0okm9ijn8uhb7ygv
```

15. postgres証明書とキーを保存するためのTLSシークレットを作成します

```
oc create secret tls mydb-postgres-cert --cert=postgres.pem --key=postgres.key -n my-namespace
```

16. repluserとrewinduserに対して手順12-15を繰り返します

4.7.2 証明書を自動で管理する方法

証明書管理ツールは、多く公開されています。この例では、cert-managerを使用します。

注意

- この例で作成された証明書はパスワードで保護されていないことに注意してください。
- Kubernetesクラスタ上で操作する場合は、ocコマンドをkubectlコマンドに読み替えてください。

cert-managerをインストールします

```
oc create namespace cert-manager

oc apply -f https://github.com/jetstack/cert-manager/releases/download/v1.3.0/cert-manager.yaml
```

自己署名発行者を作成します（これは名前空間固有またはクラスタ単位にすることができます）

この例では、名前空間my-namespaceに自己署名証明書を作成できる発行者を作成します。

```
cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Issuer
metadata:
  name: selfsigned-issuer
  namespace: my-namespace
spec:
  selfSigned: {}
EOF
```

selfsigned-issuerを使用して自己署名CA証明書を作成します

```
cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: cacert
  namespace: my-namespace
spec:
  subject:
    organizations:
      - My Organization
    organizationalUnits:
      - CA
  commonName: "My Organization Certificate Authority"
  duration: 87600h
  isCA: true
  secretName: cacert
  issuerRef:
    name: selfsigned-issuer
EOF
```

上記のコマンドは、名前空間my-namespaceのKubernetesシークレット「cacert」に格納されている自己署名ルート証明書と秘密鍵を作成します。

上記の証明書を使用してCA発行者を作成します

```
cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Issuer
metadata:
  name: ca-issuer
  namespace: my-namespace
spec:
  ca:
    secretName: cacert
EOF
```

上記のCA発行者を使用してFEPサーバ証明書を作成します

FEPCluster名が名前空間my-namespaceのmydbであると仮定します。

```
cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: mydb-fep-cert
  namespace: my-namespace
spec:
  subject:
  commonName: "mydb-headless-svc"
  dnsNames:
  - "*.my-namespace.pod"
  - "*.my-namespace.pod.cluster.local"
  - "mydb-primary-svc"
  - "mydb-primary-svc.my-namespace"
  - "mydb-primary-svc.my-namespace.svc"
  - "mydb-primary-svc.my-namespace.svc.cluster.local"
  - "mydb-replica-svc"
  - "mydb-replica-svc.my-namespace"
  - "mydb-replica-svc.my-namespace.svc"
  - "mydb-replica-svc.my-namespace.svc.cluster.local"
  duration: 8760h
  usages:
  - server auth
  secretName: mydb-fep-cert
  issuerRef:
    name: ca-issuer
EOF
```

上記のCA発行者を使用してPatroni証明書を作成します

FEPCluster名が名前空間my-namespaceのmydbであると仮定します。

```
cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: mydb-patroni-cert
  namespace: my-namespace
spec:
  subject:
  commonName: "mydb-headless-svc"
  dnsNames:
  - "*.my-namespace.pod"
  - "*.my-namespace.pod.cluster.local"
  - "*.mydb-primary-svc"
  - "*.mydb-primary-svc.my-namespace"
  - "*.mydb-replica-svc "
  - "*.mydb-replica-svc.my-namespace"
  duration: 8760h
  usages:
  - server auth
  secretName: mydb-patroni-cert
  issuerRef:
    name: ca-issuer
EOF
```

postgresユーザークライアント証明書を作成します

```
cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: mydb-postgres-cert
  namespace: my-namespace
spec:
  subject:
  commonName: "postgres"
  duration: 8760h
  usages:
  - client auth
  secretName: mydb-postgres-cert
  issuerRef:
    name: ca-issuer
EOF
```

repluserユーザークライアント証明書を作成します

```
cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: mydb-repluser-cert
  namespace: my-namespace
spec:
  subject:
  commonName: "repluser"
  duration: 8760h
  usages:
  - client auth
  secretName: mydb-repluser-cert
  issuerRef:
    name: ca-issuer
EOF
```

上記のCA発行者を使用してFEPLogging(Fluentd)サーバ証明書を作成します

FEPLogging名が名前空間feplogging-devのnflであると仮定します。

```
cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: fluentd-cert
  namespace: feplogging-dev
spec:
  subject:
  commonName: "nfl-fluentd-headless-service"
  dnsNames:
  - 'nfl-fluentd-headless-service'
  - 'nfl-fluentd-headless-service.feplogging-dev'
  - 'nfl-fluentd-headless-service.feplogging-dev.svc'
  - 'nfl-fluentd-headless-service.feplogging-dev.svc.cluster.local'
  duration: 8760h
  usages:
  - server auth
  secretName: fluentd-cert
  issuerRef:
```

```
    name: ca-issuer
EOF
```

FEPLoggingクライアント(prometheus)証明書を作成します

```
cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: prometheus-cert
  namespace: feplogging-dev
spec:
  subject:
  commonName: "prometheus"
  duration: 8760h
  usages:
  - client auth
  secretName: prometheus-cert
  issuerRef:
    name: ca-issuer
EOF
```

FEPLoggingクライアント(fluentbit)証明書を作成します

```
cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: fluentbit-cert
  namespace: feplogging-dev
spec:
  subject:
  commonName: "fluentbit"
  duration: 8760h
  usages:
  - client auth
  secretName: fluentbit-cert
  issuerRef:
    name: ca-issuer
EOF
```

上記のCA発行者を使用してFEPExporter証明書を作成します

FEPExporter名が名前空間my-namespaceのexp1であると仮定します。

```
cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: fepexporter-cert
  namespace: my-namespace
spec:
  subject:
  commonName: "exp1-service"
  dnsNames:
  - 'exp1-service'
  - 'exp1-service.fepexporter-dev'
  - 'exp1-service.fepexporter-dev.svc'
  - 'exp1-service.fepexporter-dev.svc.cluster.local'
  duration: 8760h
```

```
  usages:
  - server auth
  secretName: fepexporter-cert
  issuerRef:
    name: ca-issuer
EOF
```

FEPExporterユーザークライアント(prometheus)証明書を作成します

```
cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: prometheus-cert
  namespace: my-namespace
spec:
  subject:
  commonName: "prometheus"
  duration: 8760h
  usages:
  - client auth
  secretName: prometheus-cert
  issuerRef:
    name: ca-issuer
EOF
```

4.7.3 MTLSをサポートするFEPクラスタのデプロイ

手動の証明書管理を使用してFEPClusterをデプロイする

手動の証明書管理を使用してFEPClusterをデプロイする例として、以下にyamlを示します。MTLS関連のパラメータは赤字で強調表示されています。

```
apiVersion: fep.fujitsu.io/v2
kind: FEPCluster
metadata:
  name: mydb
  namespace: my-namespace
spec:
  fep:
    usePodName: true
    patroni:
      tls:
        certificateName: mydb-patroni-cert
        caName: cacert
    postgres:
      tls:
        certificateName: mydb-fep-cert
        caName: cacert
        privateKeyPassword: mydb-fep-private-key-password
    forceSsl: true
    podAntiAffinity: false
    mcSpec:
      limits:
        cpu: 500m
        memory: 700Mi
      requests:
        cpu: 200m
        memory: 512Mi
    customAnnotations:
      allDeployments: {}
```

```
  servicePort: 27500
  image:
    image: 'quay.io/fujitsu/fujitsu-enterprise-postgres-15-server:ubi8-15-1.0'
    pullPolicy: IfNotPresent
  sysExtraLogging: false
  podDisruptionBudget: false
  instances: 3
  syncMode: 'on'
fepChildCrVal:
  customPgAudit: |
    # define pg audit custom params here to override defaults.
    # if log volume is not defined, log_directory should be
    # changed to '/database/userdata/data/log'
    [output]
    logger = 'auditlog'
    log_directory = '/database/log/audit'
    [rule]
  customPgHba: |
    # define pg_hba custom rules here to be merged with default rules.
    # TYPE     DATABASE        USER        ADDRESS        METHOD
    hostssl    all             all         0.0.0.0/0      cert
    hostssl    replication     all         0.0.0.0/0      cert
  customPgParams: >+
    # define custom postgresql.conf parameters below to override defaults.
    # Current values are as per default FEP deployment
    shared_preload_libraries='pgx_datamasking,pgaudit,pg_prewarm'
    session_preload_libraries='pg_prewarm'
    max_prepared_transactions = 100
    max_worker_processes = 30
    max_connections = 100
    work_mem = 1MB
    maintenance_work_mem = 12MB
    shared_buffers = 128MB
    effective_cache_size = 384MB
    checkpoint_completion_target = 0.8

    # tcp parameters
    tcp_keepalives_idle = 30
    tcp_keepalives_interval = 10
    tcp_keepalives_count = 3

    # logging parameters in default fep installation
    # if log volume is not defined, log_directory should be
    # changed to '/database/userdata/data/log'
    log_directory = '/database/log'
    log_filename = 'logfile-%a.log'
    log_file_mode = 0600
    log_truncate_on_rotation = on
    log_rotation_age = 1d
    log_rotation_size = 0
    log_checkpoints = on
    log_line_prefix = '%e %t [%p]: [%l-1] user=%u,db=%d,app=%a,client=%h'
    log_lock_waits = on
    log_autovacuum_min_duration = 60s
    logging_collector = on
    pgaudit.config_file='/opt/app-root/src/pgaudit-cfg/pgaudit.conf'
    log_replication_commands = on
    log_min_messages = WARNING
    log_destination = stderr

    # wal_archive parameters in default fep installation
    archive_mode = on
    archive_command = '/bin/true'
```

```
    wal_level = replica
    max_wal_senders = 12
    wal_keep_segments = 64

  storage:
    dataVol:
      size: 2Gi
      storageClass: nfs-client
    walVol:
      size: 1200Mi
      storageClass: nfs-client
    logVol:
      size: 1Gi
      storageClass: nfs-client
  sysUsers:
    pgAdminPassword: admin-password
    pgdb: mydb
    pgpassword: mydbpassword
    pguser: mydbuser
    pgrepluser: repluser
    pgreplpassword: repluserpwd
    pgRewindUser: rewinduser
    pgRewindPassword: rewinduserpwd
    pgAdminTls:
      certificateName: mydb-postgres-cert
      caName: cacert
      sslMode: prefer

    pgrepluserTls:
      certificateName: mydb-repluser-cert
      caName: cacert
      sslMode: prefer

    pgRewindUserTls:
      certificateName: mydb-rewinduser-cert
      caName: cacert
      sslMode: prefer

    tdepassphrase: tde-passphrase
  systemCertificates:
    key: |-
      -----BEGIN RSA PRIVATE KEY-----
      MIIEowIBAAKCAQEAODFkImha8CIJiVcwXbBPlL+/DmS9/ipRhQQHxfO5x7jSOnse
      IHdFd6+Qx2GX8KAiAhVykf6kfacwBYTATU1xDgwWTm82KVRPh+kZDIj2wPcJr14m
      mTP6I6a2mavUgDhezHc9F8/dchYj3cw81XOkU6xamqrKQYlxQH48NkIOqcwhO6sK
      AHF4eWfCr8Ot44xADIA1JcU2CS1RKSZEtURZ+3OPy+j9O7Enjp1YR33ZKUHw3OpU
      9dpIneyfXBN/pT6cX3MetYwtgmpV/pHqY8pbxqGfoyRhgQDsSRCl4dtlecaZeZ4j
      uTOotcPkZELHP6eu8gaLtycG9lpbAMQl5wOr8QIDAQABAoIBACq213qPuoimExrQ
      fqXaNJmqNYK4fJqXCB6oUwfOF1u4ubkx5V532hLSPHwLs+aOlAWlbNozSoBVOu8G
      64VwrA9bv3/cJVqZZ6/UzUTbHPU+Ogh24qhwF5QU8kXZEUI1To3YsPoftalgjX9G
      FfOfLcLVC8nL3K9RiaDXxXbEYpWrYu39M3FCpAXAzV2PrNxsP9PKyNWHnBPc08z5
      tFj45/bHn+j31AVVvgWtqzOpLks57hc4Q7yW/2RoRYq2md1KI7O9OLNwtkWEOVqb
      qnraorh2TwGnNaOB5oX5/lJvKtlq778fw96jGqykBrO+DKozj9rlr1OGgYOKDwlD
      nsZJPAECgYEA+Oqf/fxtPdsNGiaL2Z/heewvtaxjw/WoEVBFECb6/y4Ro7aux9nB
      16FcVi79CwfpOUTJ7cnZvYSmBk5GWEObEIAeo61lvm/QeltM5+usAPd5/TcHXLYe
      92OnXmq7h3F4UXEkMayak8Lpu/TdmR5uOaL+m4aEu+XMY5tlxqDCnyECgYEA1h4X
      jCPi7Ja5CHK7a2Ud4TL2DNpIBE6GSK9iQ+OxFL6TsiK2Sfu6n8mx2sh+JmOKHTiE
      /gWHdHQZSSWiuULfHoYEq3Rq8S6Av3GsGtRSpOO3j7BE8C2OVptOFnNTjZmdzf2/
      YZxc5KuYLh9qeY7Y7ceOsWA8JckDgMHPYzyLAtECgYBALDOTPgDr8Y1vMIDdmlqH
      FFO4eTk/TBYIYKltgJ81KqthibeFzp4q+W7UyUhzj5a4XQOySlfYhFpJReTc3JEd
      r+o2SH3ymuEkqmUpZZjyptrMbWN4g3t4TDjaHqo6QQbD+GdcZyNy9M1Np9N5pl7E
      fUEm14dg6d3HOEhs7QVAAQKBgQDRUx3mLXc9oKRINBIyDerGLJILQqLBQxtYl81T
      ZuFizGWL8w+PCIAMkpxDrVpWqqcGpiiuRi2ElbPapOaOg2epaY/LJscd/j5z6uc8
```

```
    W3JoNIjpKoRa4fO578Pv5tM6TYHOzIF5Veoiy/a8sI3hRNuiqkM/+TsUHY5FJDRh
    aeDk4QKBgCOHIevvR+MWuwakzD6INCbb8H6fvZ3WRAT8BYYz3wW9YfnV4J4uh/BI
    moWYgIK2UpkrhA8scMUC79OFoybQeParQ35x7JI91bmTKkCqsX63fyqqYhx3SXRI
    JSktmH4E2cGmosZisjB7COKHR32wOJ5JCgaGInQxjldbGrwhZQpn
    -----END RSA PRIVATE KEY-----
  crt: |-
    -----BEGIN CERTIFICATE-----
    MIID2DCCAsCgAwIBAgIQDfFYteD4kZj4Sko2iy1IJTANBgkqhkiG9w0BAQsFADBX
    MRgwFgYDVQQKEw9NeSBPcmdhbmI6YXRpb24xCzAJBgNVBAsTAkNBMS4wLAYDVQQD
    EyVNeSBPcmdhbmI6YXRpb24gQ2VydGlmaWNhdGUgQXV0aG9yaXR5MB4XDTIxMDQy
    MDAwMDQ1OVoXDTIxMDQyMDAxMDQ1OVowGDEWMBQGA1UEAwwNKi5jaGctcHRjLnBv
    ZDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBANAxZCJoWvAiCYIXMF2w
    T5S/vw5kvf4qUYUEB8Xzuce4Ojp7HiB3RXevkMdhl/CgIgIVcpH+pH2nMAWEwE1N
    cQ4MFk5vNiIUT4fpGQyI9sD3Ca9eJpkz+iOmtpmr1IA4Xsx3PRfP3XIWI93MPNV9
    JFOsWpqqykGJcUB+PDZCNKnMITurCgBxeHInwq/DreOMQAyANSXFNgktUSkmRLVE
    Wft9D8vo/dOxJ46dWEd92SIB8N9KVPXaSJ3sn1wTf6U+nF9zHrWMLYJqVf6R6mPK
    W8ahn6MkYYEA7EkQpeHbZXnGmXmeI7kzqLXD5GRCxz+nrvIGi7cnBvZaWwDEJecN
    K/ECAwEAAaOB3jCB2zATBgNVHSUEDDAKBggrBgEFBQcDATAMBgNVHRMBAf8EAjAA
    MIG1BgNVHREEgaOwgaqCCWxvY2FsaG9zdIIbKi5jaGctcHRjLnBvZC5jbHVzdGVy
    LmxvY2FsghMqLm15ZGItaGVhZGxlc3Mtc3ZjghsqLm15ZGItaGVhZGxlc3Mtc3Zj
    LmNoZy1wdGOCHyoubXlkYi1oZWFkbGVzcy1zdmMuY2hnLXBOYy5zdmOCLSoubXlk
    Yi1oZWFkbGVzcy1zdmMuY2hnLXBOYy5zdmMuY2x1c3Rlci5sb2NhbDANBgkqhkiG
    9w0BAQsFAAOCAQEALnhliDflu+BHp5conq4dXBwD/Ti2YR5TWQixM/Oa6OD4KecZ
    MmaLlOT+OJJvA/j2IufZpc7dzEx5mZDkR2CRmoq1OqZXqCRTrBZSXm6ARQWoYpeg
    9cOl4f8roxrkMGUKVPTKUwAvbnNYhD2l6PlBPwMpkMUfqFaSEXMaPyQKhrtQxdpH
    WjuS54OPOlmOpeYu/yiaD98LtrTXnb6jch84SKf6Vii4HAVQyMeJaW+dpkqcI2+V
    Q4fkwWYSJy8BNcmXCwvHDLDy+s4EXWvHafhusuUhcp4HyMb1A6hd5hJhgFSnEvLy
    kLAOL9LaScxee6V756Vt9TN1NGjwmwyQDOhnQQ==
    -----END CERTIFICATE-----
  cacrt: |-
    -----BEGIN CERTIFICATE-----
    MIIDXDCCAkSgAwIBAgIRAMPzF3BNFxT9HWE+NXlFQjQwDQYJKoZIhvcNAQELBQAw
    VzEYMBYGA1UEChMPTXkgT3JnYW5pemF0aW9uMQswCQYDVQQLEwJDQTEuMCwGA1UE
    AxMlTXkgT3JnYW5pemF0aW9uIENlcnRpZmljYXRlIEF1dGhvcmlOeTAeFw0yMTA0
    MTkwNDQ0MjNaFw0zMTA0MTcwNDQ0MjNaMFcxGDAWBgNVBAoTD015IE9yZ2FuaXph
    dGlvbjELMAkGA1UECxMCQ0ExLjAsBgNVBAMTJU15IE9yZ2FuaXphdGlvbiBDZXJO
    aWZpY2FOZSBBdXRob3JpdHkwggEiMAOGCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIB
    AQC5t6CS23Glk65YMw5e4i4xHldyxkCZS67w/6LWqeIlYKmFAaEl83WWy8MHUpOb
    4mahtUafEzDEOX6+URf72J8mOvoldQ5FYr1AyUOyX8U9OwGFqhbEgKRqt7vZEwIe
    2961fwqHh6917zI4xmt5W6ZJ5dBQVtkhzB+Pf7O6KBYjHoCnBBkfNVzsfZQ/1hnR
    OUzimfAc7Ze+UNwhXJhinFRJ3YuR+xiOTpPkl1GXPhLgFSQhekz4KepcbQEQKejb
    jgOdum1oBYIXZTSSbiO9rNmfUVLB5DcVOvZbSrGxLjWLBt5U8N2xf2d1bvkQW+bw
    Kklf9OG26bAi27tujurzN3r3AgMBAAGjIzAhMA4GA1UdDwEB/wQEAwICpDAPBgNV
    HRMBAf8EBTADAQH/MAOGCSqGSIb3DQEBCwUAA4IBAQAMOCN3n5C/KOT4uZ4ewwKK
    rHmANBPVM9u6MJBO8U62HcqLeoCuDFeU8zmUjLHjsQaPX64mJZlR7T5y52gEKO5A
    OqsBz3pg/vJ5DJTvO698+1Q1hB9k3smQdksAim19FZqysB7J4zK/+8aJ/q2kIFvs
    Jk3ekwQdQ3xfggklBQVuf76gr1vOuYlPtfPfPlfcGZO6Im6mqbajenXoR1PxPBO
    +zyCS8DkgPtDulplruwvXCFMYw9TPbzXKlt7tlsqRXogYLnxWJDzM1nOYCnD+rDm
    qxenV9Ir8RqZOXSYuUyzRka5N4dhIhrzTAiNdeU5gzynXOz67u/Iefz1iK9ZcdE3
    -----END CERTIFICATE-----
```

自動の証明書管理を使用してFEPClusterをデプロイする

自動の証明書管理を使用してFEPClusterをデプロイする例として、以下にyamlを示します。MTLS関連のパラメータは赤字で強調表示されています。

```
apiVersion: fep.fujitsu.io/v2
kind: FEPCluster
metadata:
  name: mydb
  namespace: my-namespace
spec:
  fep:
```

```
  usePodName: true
  patroni:
    tls:
      certificateName: mydb-patroni-cert
  postgres:
    tls:
      certificateName: mydb-fep-cert
  forceSsl: true
  podAntiAffinity: false
  mcSpec:
    limits:
      cpu: 500m
      memory: 700Mi
    requests:
      cpu: 200m
      memory: 512Mi
  customAnnotations:
    allDeployments: {}
  servicePort: 27500
  image:
    image: 'quay.io/fujitsu/fujitsu-enterprise-postgres-15-server:ubi8-15-1.0'
    pullPolicy: IfNotPresent
  sysExtraLogging: false
  podDisruptionBudget: false
  instances: 3
  syncMode: 'on'
fepChildCrVal:
  customPgAudit: |
    # define pg audit custom params here to override defaults.
    # if log volume is not defined, log_directory should be
    # changed to '/database/userdata/data/log'
    [output]
    logger = 'auditlog'
    log_directory = '/database/log/audit'
    [rule]
  customPgHba: |
    # define pg_hba custom rules here to be merged with default rules.
    # TYPE     DATABASE        USER        ADDRESS        METHOD
    hostssl    all             all         0.0.0.0/0      cert
    hostssl    replication     all         0.0.0.0/0      cert
  customPgParams: >+
    # define custom postgresql.conf parameters below to override defaults.
    # Current values are as per default FEP deployment
    shared_preload_libraries='pgx_datamasking,pgaudit,pg_prewarm'
    session_preload_libraries='pg_prewarm'
    max_prepared_transactions = 100
    max_worker_processes = 30
    max_connections = 100
    work_mem = 1MB
    maintenance_work_mem = 12MB
    shared_buffers = 128MB
    effective_cache_size = 384MB
    checkpoint_completion_target = 0.8

    # tcp parameters
    tcp_keepalives_idle = 30
    tcp_keepalives_interval = 10
    tcp_keepalives_count = 3

    # logging parameters in default fep installation
    # if log volume is not defined, log_directory should be
    # changed to '/database/userdata/data/log'
```

```
    log_directory = '/database/log'
    log_filename = 'logfile-%a.log'
    log_file_mode = 0600
    log_truncate_on_rotation = on
    log_rotation_age = 1d
    log_rotation_size = 0
    log_checkpoints = on
    log_line_prefix = '%e %t [%p]: [%l-1] user=%u,db=%d,app=%a,client=%h'
    log_lock_waits = on
    log_autovacuum_min_duration = 60s
    logging_collector = on
    pgaudit.config_file='/opt/app-root/src/pgaudit-cfg/pgaudit.conf'
    log_replication_commands = on
    log_min_messages = WARNING
    log_destination = stderr

    # wal_archive parameters in default fep installation
    archive_mode = on
    archive_command = '/bin/true'
    wal_level = replica
    max_wal_senders = 12
    wal_keep_segments = 64

  storage:
    dataVol:
      size: 2Gi
      storageClass: nfs-client
    walVol:
      size: 1200Mi
      storageClass: nfs-client
    logVol:
      size: 1Gi
      storageClass: nfs-client
  sysUsers:
    pgAdminPassword: admin-password
    pgdb: mydb
    pgpassword: mydbpassword
    pguser: mydbuser
    pgrepluser: repluser
    pgreplpassword: repluserpwd
    pgRewindUser: rewinduser
    pgRewindPassword: rewinduserpwd
    pgAdminTls:
      certificateName: mydb-postgres-cert
      sslMode: verify-full

    pgrepluserTls:
      certificateName: mydb-repluser-cert
      sslMode: verify-full

    pgRewindUserTls:
      certificateName: mydb-rewinduser-cert
      sslMode: verify-full

    tdepassphrase: tde-passphrase
  systemCertificates:
    key: |-
      -----BEGIN RSA PRIVATE KEY-----
      MIIEowIBAAKCAQEA0DFkImha8CIJiVcwXbBPlL+/DmS9/ipRhQQHxfO5x7jSOnse
      IHdFd6+Qx2GX8KAiAhVykf6kfacwBYTATU1xDgwWTm82KVRPh+kZDIj2wPcJr14m
      mTP6I6a2mavUgDhezHc9F8/dchYj3cw81X0kU6xamqrKQYlxQH48NkIOqcwhO6sK
      AHF4eWfCr8Ot44xADIA1JcU2CS1RKSZEtURZ+3OPy+j9O7Enjp1YR33ZKUHw3OpU
      9dpIneyfXBN/pT6cX3MetYwtgmpV/pHqY8pbxqGfoyRhgQDsSRCl4dtlecaZeZ4j
```

```
    uTOotcPkZELHP6eu8gaLtycG9lpbAMQl5wOr8QIDAQABAoIBACq213qPuoimExrQ
    fqXaNJmqNYK4fJqXCB6oUwfOF1u4ubkx5V532hLSPHwLs+aOlAWlbNozSoBVOu8G
    64VwrA9bv3/cJVqZZ6/UzUTbHPU+Ogh24qhwF5QU8kXZEUI1To3YsPoftalgjX9G
    FfOfLcLVC8nL3K9RiaDXxXbEYpWrYu39M3FCpAXAzV2PrNxsP9PKyNWHnBPcO8z5
    tFj45/bHn+j31AVVvgWtqzOpLks57hc4Q7yW/2RoRYq2md1KI7O9OLNwtkWEOVqb
    qnraorh2TwGnNaOB5oX5/lJvKtlq778fw96jGqykBrO+DKozj9rlr1OGgYOKDwlD
    nsZJPAECgYEA+Oqf/fxtPdsNGiaL2Z/heewvtaxjw/WoEVBFECb6/y4Ro7aux9nB
    16FcVi79CwfpOUTJ7cnZvYSmBk5GWEObEIAeo61lvm/QeltM5+usAPd5/TcHXLYe
    92OnXmq7h3F4UXEkMayak8Lpu/TdmR5uOaL+m4aEu+XMY5tlxqDCnyECgYEA1h4X
    jCPi7Ja5CHK7a2Ud4TL2DNpIBE6GSK9iQ+OxFL6TsiK2Sfu6n8mx2sh+JmOKHTiE
    /gWHdHQZSSWiuULfHoYEq3Rq8S6Av3GsGtRSpO03j7BE8C2OVptOFnNTjZmdzf2/
    YZxc5KuYLh9qeY7Y7ceOsWA8JckDgMHPYzyLAtECgYBALDOTPgDr8Y1vMIDdmlqH
    FFO4eTk/TBYIYKltgJ81KqthibeFzp4q+W7UyUhzj5a4XQOySlfYhFpJReTc3JEd
    r+o2SH3ymuEkqmUpZZjyptrMbWN4g3t4TDjaHqo6QQbD+GdcZyNy9M1Np9N5pl7E
    fUEm14dg6d3HOEhs7QVAAQKBgQDRUx3mLXc9oKRINBIyDerGLJILQqLBQxtYl81T
    ZuFizGWL8w+PCIAMkpxDrVpWqqcGpiiuRi2ElbPapOaOg2epaY/LJscd/j5z6uc8
    W3JoNljpKoRa4fO578Pv5tM6TYHOzlF5Veoiy/a8sI3hRNuiqkM/+TsUHY5FJDRh
    aeDk4QKBgCOHIevvR+MWuwakzD6lNCbb8H6fvZ3WRAT8BYYz3wW9YfnV4J4uh/Bl
    moWYgIK2UpkrhA8scMUC79OFoybQeParQ35x7Jl91bmTKkCqsX63fyqqYhx3SXRl
    JSktmH4E2cGmosZisjB7COKHR32wOJ5JCgaGInQxjldbGrwhZQpn
    -----END RSA PRIVATE KEY-----
  crt: |-
    -----BEGIN CERTIFICATE-----
    MIID2DCCAsCgAwIBAgIQDfFYteD4kZj4Sko2iy1IJTANBgkqhkiG9w0BAQsFADBX
    MRgwFgYDVQQKEw9NeSBPcmdhbml6YXRpb24xCzAJBgNVBAsTAkNBMS4wLAYDVQQD
    EyVNeSBPcmdhbml6YXRpb24gQ2VydGlmaWNhdGUgQXV0aG9yaXR5MB4XDTIxMDQy
    MDAwMDQ1OVoXDTIxMDQyMDAxMDQ1OVowGDEWMBQGA1UEAwwNKi5jaGctcHRjLnBv
    ZDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBANAxZCJoWvAiCYlXMF2w
    T5S/vw5kvf4qUYUEB8Xzuce4Ojp7HiB3RXevkMdhl/CgIgIVcpH+pH2nMAWEwE1N
    cQ4MFk5vNilUT4fpGQyI9sD3Ca9eJpkz+iOmtpmr1IA4Xsx3PRfP3XIWI93MPNV9
    JFOsWpqqykGJcUB+PDZCNKnMITurCgBxeHlnwq/DreOMQAyANSXFNgktUSkmRLVE
    Wft9D8vo/dOxJ46dWEd92SlB8N9KVPXaSJ3sn1wTf6U+nF9zHrWMLYJqVf6R6mPK
    W8ahn6MkYYEA7EkQpeHbZXnGmXmeI7kzqLXD5GRCxz+nrvIGi7cnBvZaWwDEJecN
    K/ECAwEAAaOB3jCB2zATBgNVHSUEDDAKBggrBgEFBQcDATAMBgNVHRMBAf8EAjAA
    MIG1BgNVHREEgaOwgaqCCWxvY2FsaG9zdIIbKi5jaGctcHRjLnBvZC5jbHVzdGVy
    LmxvY2FsghMqLm15ZGItaGVhZGxlc3Mtc3ZjghsqLm15ZGItaGVhZGxlc3Mtc3Zj
    LmNoZy1wdGOCHyoubXlkYi1oZWFkbGVzcy1zdmMuY2hnLXBOYy5zdmOCLSoubXlk
    Yi1oZWFkbGVzcy1zdmMuY2hnLXBOYy5zdmMuY2x1c3Rlci5sb2NhbDANBgkqhkiG
    9w0BAQsFAAOCAQEALnhliDflu+BHp5conq4dXBwD/Ti2YR5TWQixM/Oa6OD4KecZ
    MmaLlOT+OJJvA/j2IufZpc7dzEx5mZDkR2CRmoq1OqZXqCRTrBZSXm6ARQWoYpeg
    9cOl4f8roxrkMGUKVPTKUwAvbnNYhD2l6PlBPwMpkMUfqFaSEXMaPyQKhrtQxdpH
    WjuS54OPOlmOpeYu/yiaD98LtrTXnb6jch84SKf6Vii4HAVQyMeJaW+dpkqcI2+V
    Q4fkwWYSJy8BNcmXCwvHDLDy+s4EXWvHafhusuUhcp4HyMb1A6hd5hJhgFSnEvLy
    kLAOL9LaScxee6V756Vt9TN1NGjwmwyQDOhnQQ==
    -----END CERTIFICATE-----
  cacrt: |-
    -----BEGIN CERTIFICATE-----
    MIIDXDCCAkSgAwIBAgIRAMPzF3BNFxT9HWE+NXlFQjQwDQYJKoZIhvcNAQELBQAw
    VzEYMBYGA1UEChMPTXkgT3JnYW5pemFOaW9uMQswCQYDVQQLEwJDQTEuMCwGA1UE
    AxMlTXkgT3JnYW5pemFOaW9uIENlcnRpZmljYXRlIEF1dGhvcmlOeTAeFw0yMTA0
    MTkwNDQOMjNaFw0zMTA0MTcwNDQOMjNaMFcxGDAWBgNVBAoTD015IE9yZ2FuaXph
    dGlvbjELMAkGA1UECxMCQOExLjAsBgNVBAMTJU15IE9yZ2FuaXphdGlvbiBDZXJO
    aWZpY2FOZSBBdXRob3JpdHkwggEiMAOGCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIB
    AQC5t6CS23Glk65YMw5e4i4xHldyxkCZS67w/6LWqeIlYKmFAaEl83WWy8MHUpOb
    4mahtUafEzDEOX6+URf72J8mOvoldQ5FYr1AyUOyX8U9OwGFqhbEgKRqt7vZEwIe
    2961fwqHh6917zI4xmt5W6ZJ5dBQVtkhzB+Pf7O6KBYjHoCnBBkfNVzsfZQ/1hnR
    OUzimfAc7Ze+UNwhXJhinFRJ3YuR+xiOTpPkl1GXPhLgFSQhekz4KepcbQEQKejb
    jgOdum1oBYIXZTSSbiO9rNmfUVLB5DcVOvZbSrGxLjWLBt5U8N2xf2d1bvkQW+bw
    Kklf9OG26bAi27tujurzN3r3AgMBAAGjIzAhMA4GA1UdDwEB/wQEAwICpDAPBgNV
    HRMBAf8EBTADAQH/MAOGCSqGSIb3DQEBCwUAA4IBAQAMOCN3n5C/KOT4uZ4ewwKK
    rHmANBPVM9u6MJBO8U62HcqLeoCuDFeU8zmUjLHjsQaPX64mJZlR7T5y52gEKO5A
    OqsBz3pg/vJ5DJTvO698+1Q1hB9k3smQdksAim19FZqysB7J4zK/+8aJ/q2klFvs
    Jk3ekwQdQ3xfggklBQVuf76gr1vOuYlPtfPffPlfcGZO6Im6mqbajenXoR1PxPBO
```

```
+zyCS8DkgPtDu|p|ruwvXCFMYw9TPbzXK|t7t|sqRXogYLnxWJDzM1nOYCnD+rDm
qxenV9Ir8RqZOXSYuUyzRka5N4dhIhrzTAiNdeU5gzynXOz67u/Iefz1iK9ZcdE3
-----END CERTIFICATE-----
```

4.7.4 設定可能なパラメータ

MTLSを有効にするには、次のパラメータを変更します

Key	Value	Details
spec.fep.usePodName	True	MTLSの場合、このキーを定義してtrueに設定する必要があります。MTLSを使用しないTLS接続の場合、省略できます。ただし、これもtrueに設定することをお勧めします。
spec.fep.patroni.tls.certificateName	<secret-name>	Patroni RESTAPIのtls.crtの証明書とtls.keyの秘密鍵を含むKubernetesシークレットの名前。MTLS Patroni REST API通信の場合、このキーを定義する必要があります。秘密鍵をパスワードで保護することはできません。cert-managerを使用する場合、シークレットにはca.crtのCAバンドルも含まれます。
spec.fep.patroni.tls.caName	<configmap-name>	CAバンドルを含むKubernetesconfigmapの名前。cert-managerを使用している場合、ca.crtはすでに上記のシークレットに含まれています。この場合、このキーは省略できます。
spec.fep.postgres.tls.certificateName	<secret-name>	Postgresサーバのtls.crtの証明書とtls.keyの秘密鍵を含むKubernetesシークレットの名前。MTLS Postgres通信の場合、このキーを定義する必要があります。秘密鍵はパスワードで保護できます。cert-managerを使用する場合、シークレットにはca.crtのCAバンドルも含まれます。
spec.fep.postgres.tls.caName	<configmap-name>	CAバンドルを含むKubernetesconfigmapの名前。cert-managerを使用している場合、ca.crtはすでに上記のシークレットに含まれています。この場合、このキーは省略できます。
spec.fep.postgres.tls.privateKeyPassword	<secret-name>	Postgresサーバの秘密鍵のパスワードを含むKubernetesシークレットの名前。.
spec.fepChildCrVal.sysUsers.pgAdminTls.certificateName	<secret-name>	「postgres」ユーザーのtls.crtの証明書とtls.keyの秘密鍵を含むKubernetesシークレットの名前。MTLS Postgres通信の場合、このキーを定義する必要があります。秘密鍵をパスワードで保護すること

Key	Value	Details
		はできません。cert-managerを使用する場合、シークレットにはca.crtのCAバンドルも含まれます。
spec.fepChildCrVal.sysUsers.pgAdminTls.caName	<configmap-name>	CAバンドルを含むKubernetesconfigmapの名前。cert-managerを使用している場合、ca.crtはすでに上記のシークレットに含まれています。この場合、このキーは省略できます。
spec.fepChildCrVal.sysUsers.pgAdminTls.sslMode	verify-full	MTLSの場合、この値をverify-fullに設定する必要があります。TLSのみが必要な場合は、verify-caまたはpreferに設定できます。
spec.fepChildCrVal.sysUsers.pgrepluserTls.certificateName	<secret-name>	「repluser」ユーザーのtls.crtの証明書とtls.keyの秘密鍵を含むKubernetesシークレットの名前。MTLS Postgres通信の場合、このキーを定義する必要があります。秘密鍵をパスワードで保護することはできません。cert-managerを使用する場合、シークレットにはca.crtのCAバンドルも含まれます。
spec.fepChildCrVal.sysUsers.pgrepluserTls.caName	<configmap-name>	CAバンドルを含むKubernetesconfigmapの名前。cert-managerを使用している場合、ca.crtはすでに上記のシークレットに含まれています。この場合、このキーは省略できます。
spec.fepChildCrVal.sysUsers.pgrepluserTls.sslMode	verify-full	MTLSの場合、この値をverify-fullに設定する必要があります。TLSのみが必要な場合は、verify-caまたはpreferに設定できます。
spec.fepChildCrVal.sysUsers.pgRewindUserTls.certificateName	<secret-name>	「rewinduser」ユーザーのtls.crtの証明書とtls.keyの秘密鍵を含むKubernetesシークレットの名前。MTLS Postgres通信の場合、このキーを定義する必要があります。秘密鍵をパスワードで保護することはできません。cert-managerを使用する場合、シークレットにはca.crtのCAバンドルも含まれます。
spec.fepChildCrVal.sysUsers.pgRewindUserTls.caName	<configmap-name>	CAバンドルを含むKubernetesconfigmapの名前。cert-managerを使用している場合、ca.crtはすでに上記のシークレットに含まれています。この場合、このキーは省略できます。
spec.fepChildCrVal.sysUsers.pgRewindUserTls.sslMode	verify-full	MTLSの場合、この値をverify-fullに設定する必要があります。TLSのみが必要な場合は、verify-caまたはpreferに設定できます。

MTLSを実行するには、pg_hba.confをカスタマイズする必要もあります。以下は2つの可能な設定です。

spec.fep.customPgHba	hostssl all all 0.0.0.0/0 cert hostssl replication all 0.0.0.0/0 cert

上記の設定により、FEPサーバは認証を実行します。同時に、クライアント証明書の信頼性を検証します。

spec.fep.customPgHba	hostssl all all 0.0.0.0/0 md5 clientcert=verify-full hostssl replication repluser 0.0.0.0/0 md5 clientcert=verify-full

上記の設定により、FEPサーバはmd5認証を実行し、クライアント証明書の信頼性を検証します。

4.8 レプリケーションスロット

4.8.1 MTLSを使用したロジカルレプリケーションの設定

ロジカルレプリケーションの設定について説明します。

以下の手順で、MTLSを使用してロジカルレプリケーションを設定します。

1. 2つのFEPClusterを作成し（パブリッシャーとサブスクライバー）、それらが相互に通信できることを確認します。FEPClusterの作成については、“4.1 オペレーターを使用したFEPClusterのデプロイ”を参照してください。
2. パブリッシャーをセットアップするには、パブリッシャーとして使用するクラスタのFEPCluster yamlに次の変更を加えます。
   a. spec.fepの下にセクションreplicationSlotsを追加して、レプリケーションスロットを作成します。
   「database」は、ロジカルレプリケーションを設定するデータベースの名前である必要があります。

   b. 以下に示すように、spec.fepの下にセクションpostgresを追加します。

   caName = CA用に作成されたconfigmapの名前

certificateName = ユーザーによって作成されたサーバ証明書を含むシークレット

```
79    customAnnotations:
80      allDeployments: {}
81    servicePort: 27500
82    postgres:
83      tls:
84        caName: cacert
85        certificateName: my-fep-cert
86    image:
```

c. spec.fepChildCrVal.customPgParamsの下のwal_levelパラメータの値を「replica」から「logical」論理に変更します。

```
301
302     archive_mode = on
303
304     archive_command = 'pgbackrest --stanza=backupstanza
305     --config=/database/userdata/pgbackrest.conf archive-push %p'
306
307     wal_level = logical
308
309     max_wal_senders = 12
310
311     wal_keep_size = 401
```

d. 以下に示すように、spec.fepChildCrVal.customPgHbaの下にエントリを追加します。

これには、クライアントが証明書を提示する必要があり、証明書認証のみが許可されます。

「SubClusterName」と「SubNamespace」を、サブスクライバーFEPClusterに従って適切な値に置き換えます。

```
  [rule]
customPgHba: |
  # define pg_hba custom rules here to be merged with default rules.
  # TYPE      DATABASE        USER        ADDRESS         METHOD
  hostssl all all  <SubClusterName>-primary-svc.<SubNamespace>.svc.cluster.local cert
customPgParams: >
```

3. サブスクライバーをセットアップするには、サブスクライバーとして使用するクラスタのFEPCluster yamlに次の変更を加えます。

a. 以下に示すように、spec.fepChildCrValの下にcustomCertificatesを追加します。

caName = CA用に作成されたconfigmapの名前（つまり、サーバ/クライアント証明書の署名/認証に使用されるCA証明書は、「cacert」と呼ばれるconfigMapとしてマウントされます）

certificateName = ユーザーによって作成されたサーバ証明書を含むシークレット

username = ロジカルレプリケーションのためにパブリッシャークラスタで作成されたロールの名前

```
74  fepChildCrVal:
75    customCertificates:
76      - caName: cacert
77        certificateName: my-logicalrepl-cert
78        userName: logicalrepluser
79    customPgAudit: |
80      # define pg audit custom params here to override defaults.
81      # if log volume is not defined, log_directory should be
```

4. パブリッシャーFEPClusterのPodに接続してから、以下に示すようにpostgresデータベースに接続します。

```
sh-4.4$ psql -h /tmp -p 27500 -U postgres
Password for user postgres:
psql (13.1)
Type "help" for help.

postgres=#
```

5. 次に、パブリッシャー側で、複製するテーブルを含むデータベースに接続し、logicalrepluserなどのロールを作成して、このロールに必要な権限を付与します。

   以下に例を示します。付与する権限は、要件によって異なる場合があります。

```
db1=# CREATE ROLE logicalrepluser WITH REPLICATION LOGIN PASSWORD 'my_password';
CREATE ROLE
db1=# GRANT ALL PRIVILEGES ON DATABASE db1 TO logicalrepluser;
GRANT
db1=# GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO logicalrepluser;
GRANT
db1=#
```

6. パブリッシャー側で、パブリケーションを作成し、パブリケーションを変更して、複製する必要のあるテーブルを追加します。

```
db1=# create publication my_publication;
CREATE PUBLICATION
db1=# alter publication my_publication add table my_table;
ALTER PUBLICATION
db1=#
```

7. サブスクライバー側では、上記の手順3のaで追加されたカスタム証明書が、次のようにパス/ tmp / custom_certs /にマウントされます。

```
sh-4.4$ ls -rlt /tmp/custom_certs
total 0
drwxr-xr-t. 3 1001190000 root 103 Aug 10 10:08 logicalrepluser
sh-4.4$ ls -rlt /tmp/custom_certs/logicalrepluser
total 0
lrwxrwxrwx. 1 1001190000 root 14 Aug 10 10:08 tls.key -> ..data/tls.key
lrwxrwxrwx. 1 1001190000 root 14 Aug 10 10:08 tls.crt -> ..data/tls.crt
lrwxrwxrwx. 1 1001190000 root 13 Aug 10 10:08 ca.crt -> ..data/ca.crt
sh-4.4$
```

8. ロジカルレプリケーションはデータのみを複製し、テーブル構造は複製しないため、複製されるテーブルの構造はサブスクライバークラスタに存在する必要があります。

   以下に示すようにサブスクリプションを作成します。

```
db1=# CREATE SUBSCRIPTION my_subscription CONNECTION 'host=fepcluster-publisher-primary-svc.ns-a.svc.cluster.local port=27500 sslcert=/tmp/custom
_certs/logicalrepluser/tls.crt sslkey=/tmp/custom_certs/logicalrepluser/tls.key sslrootcert=/tmp/custom_certs/logicalrepluser/ca.crt sslmode=veri
fy-full dbname=db1 user=logicalrepluser' PUBLICATION my_publication WITH (slot_name=myslot1, create_slot=false);
CREATE SUBSCRIPTION
```

   上記の例のコマンドは次のとおりです。

```
CREATE SUBSCRIPTION my_subscription CONNECTION 'host=fepcluster-publisher-primary-svc.ns-a.svc.cluster.local
port=27500 sslcert=/tmp/custom_certs/logicalrepluser/tls.crt sslkey=/tmp/custom_certs/logicalrepluser/tls.key
sslrootcert=/tmp/custom_certs/logicalrepluser/ca.crt sslmode=verify-full password=my_password user=logicalrepluser
dbname=db1' PUBLICATION my_publication WITH (slot_name=myslot1, create_slot=false);

Host = primary service of the publisher FEP Cluster
sslcert, sslkey, sslrootcert = path to certificates mounted on the Subscriber FEP Cluster
user= Role created on the Publisher side
password= password for the role
dbname= database which contains the tables to be replicated
```

指定内容

Host = パブリッシャーFEPクラスタのプライマリサービス
sslcert, sslkey, sslrootcert = サブスクライバーFEPクラスタにマウントされた証明書へのパス
user= パブリッシャー側で作成され、サブスクライバーからパブリッシャーへの論理レプリケーション接続を確立するために使用されるロール
dbname= 複製されるテーブルを含むデータベース

4.9 FEPログ機能

FEPClusterはログファイルおよび監査ログファイルを生成します。これらのログファイルは、クラスタの状態とデバッグの目的を理解するのに役立ちます。デフォルトでは、ログファイルはコンテナの永続ボリュームに保存されます。ユーザーは、これらのログファイルおよび監査ログファイルをElasticsearchなどの分析プラットフォームに転送することでログ監視機能を有効にできます。

監視と転送を有効にするには、2つの手順があります。

1. FEPLogging構成-FEPLoggingインスタンスの作成
2. FEPCluster構成-FEPClusterでのロギングの有効化

FEP Loggingインスタンスは、Fluentdを実行するスタンドアロンコンテナです。FEPClusterから転送されたログを受け入れ、ログエントリの重大度に応じてデータを集約し、監視とアラートの目的でPrometheusに提示します。オプションで、詳細分析のためにこれらのログをElasticsearchインスタンスに転送するように構成できます。

FEPClusterでロギングが有効になっている場合、fluentbitを含むサイドカーがFEPサーバコンテナと一緒に展開されます。このfluentbitサイドカーは、永続ボリューム上のFEPサーバログファイルと監査ログファイルを監視し、FEPLoggingインスタンスに転送します。

複数のFEPClusterは、ログを単一のFEPLoggingインスタンスに転送できます。

ユーザーは、FEPClusterとFEPLoggingの間で2種類の接続を持つことができます。

- セキュアでない接続:TLS/MTLS証明書なし
- セキュアな接続:TLS/MTLS証明書を使用

コンポーネント間の安全な接続のために2つのオプションがあります。

- ユーザーが独自の証明書を使用する
- ユーザーが自己署名証明書を作成する(“4.7.2 証明書を自動で管理する方法”を参照)

FEP Loggingインスタンスは、追加のコンポーネントなしでスタンドアロンで実行できます。詳細なログ分析のために、ユーザーはログをElasticStackまたはElasticCloudに転送するようにFEPLoggingインスタンスを構成できます。Elastic Stackをデプロイする方法、またはElastic Cloudにサインアップする方法については、ElasticDocumentを参照してください。

4.9.1 FEPLoggingの設定

FEPLoggingカスタムリソースを介してFEPLoggingインスタンスをデプロイおよび設定する方法について説明します。FEPLoggingは、FEPClusterから送信されたログを受け入れ、アラームを発生させるためにそれらをElasticsearchまたはPrometheusに転送する別のカスタムリソースです。ユーザーは、FEPClusterのログ機能を有効にする前に、FEPLoggingカスタムリソースを作成する必要があります。

4.9.1.1 FEPLoggingカスタムリソース - spec

FEPLoggingの設定で必要なパラメータを定義するには、specの下にfepLoggingセクションを追加する必要があります。

以下にテンプレートの例を示します。

```
spec:
  fepLogging:
    elastic:
      authSecret:
        secretName: elastic-auth
        passwordKey: password
        userKey: username
      host: elastic-passthrough.apps.openshift.com
      logstashPrefix: postgres
      port: 443
      scheme: https
      sslVerify: true
      tls:
        certificateName: elastic-cert
        caName: elastic-cacert
    image:
      pullPolicy: IfNotPresent
    mcSpec:
      limits:
        cpu: 500m
        memory: 700Mi
      requests:
        cpu: 200m
        memory: 512Mi
    restartRequired: false
    sysExtraLogging: false
    scrapeInterval: 30s
    scrapeTimeout: 30s
    tls:
      certificateName: fluentd-cert
      caName: cacert
  prometheus:
    …
```

fepLoggingセクションで定義されているすべてのパラメータと設定内容について、以下に示します。

フィールド	必須/オプション	説明	変更可否
spec.fepLogging.image.image	オプション	FEPLoggingのFluentdイメージ	可
spec.fepLogging.image.pullPolicy	必須	FEPLoggingのFluentdイメージプルポリシー	可
spec.fepLogging.mcSpec.limits.cpu	必須	Fluentdコンテナに割り当てられた最大CPU	可
spec.fepLogging.mcSpec.limits.memory	必須	Fluentdコンテナに割り当てられた最大メモリ	可
spec.fepLogging.mcSpec.requests.cpu	必須	Fluentdコンテナーの開始時のCPU割り当て	可
spec.fepLogging.mcSpec.requests.memory	必須	Fluentdコンテナの開始時のメモリ割り当て	可
spec.fepLogging.sysExtraLogging	必須	オペレーターの追加のデバッグメッセージをオンにするには、値をtrueに設定します。いつでもオン/オフを切り替えることができます。	可
spec.fepLogging.restartRequired	必須	証明書のローテーション後など、新しい構成を適用するためにFEPLoggingインスタンスを再起動します	可
spec.fepLogging.scrapeInterval	オプション	PrometheusがFEPLoggingインスタンスからメトリックをフェッチするためのスクレイピング間隔	可

フィールド	必須/オプション	説明	変更可否
spec.fepLogging.scrapeTimeout	オプション	PrometheusがFEPLoggingインスタンスからメトリックをフェッチするためのスクレイピングタイムアウト	可
spec.fepLogging.elastic.host	オプション	ターゲットElasticsearchホスト名	可
spec.fepLogging.elastic.port	オプション	ターゲットElasticsearchポート番号	可
spec.fepLogging.elastic.authSecret.secretName	オプション	Elasticsearch認証のユーザー名とパスワードを含むシークレット名	可
spec.fepLogging.elastic.authSecret.userKey	オプション	Elasticsearch認証シークレットで指定されたユーザー名キー	可
spec.fepLogging.elastic.authSecret.passwordKey	オプション	Elasticsearch認証シークレットで指定されたパスワードキー	可
spec.fepLogging.elastic.logstashPrefix	オプション	Elasticsearchでインデックスパターンを区別するためのLogstashプレフィックス。デフォルト値はpostgresです。	可
spec.fepLogging.elastic.auditLogstashPrefix	オプション	監査ログのElasticsearchでインデックスパターンを区別するためのLogstashプレフィックス。デフォルト値は、logstashPrefixと同じ値になります。	可
spec.fepLogging.elastic.scheme	オプション	FEPLoggingとElasticsearch間の接続スキーム。可能なオプションはhttpとhttps	可
spec.fepLogging.elastic.sslVerify	オプション	ssl証明書を検証する場合は、trueに設定します。falseに設定すると、TLSの重要性は考慮されません。	可
spec.fepLogging.elastic.tls.certificateName	オプション	Fluentd証明書を保持するKubernetesのシークレット名	可
spec.fepLogging.elastic.tls.caName	オプション	Elasticsearch TLS接続を検証するElasticsearchのcacertを保持するKubernetesコンフィグマップ	可
spec.fepLogging.tls.certificateName	オプション	Fluentd証明書を保持するKubernetesのシークレット名	可
spec.fepLogging.tls.caName	オプション	FEPLoggingとPrometheusの間のMTLSを設定するためのFluentdのcacertを保持するKubernetesのコンフィグマップ	可
spec.prometheus.tls.certificateName	オプション	Prometheus証明書を保持するKubernetesのシークレット名	可
spec.prometheus.tls.caName	オプション	FEPLoggingとPrometheusの間のMTLSを設定するためのFluentdのcacertを保持するKubernetesのコンフィグマップ	可

4.9.1.1.1 fepLogging imageの定義

imageプロパティは、デフォルトのFluentdイメージとFEPLogging CRからのpullPolicy以外を指定するために使用されます。

指定しない場合は、オペレーターが提供するデフォルトイメージが使用されます。

例)

```
spec:
  fepLogging:
    image:
      image: 'quay.io/fujitsu/fujitsu-enterprise-postgres-fluentbit:ubi8-15-1.0'
      pullPolicy: IfNotPresent
```

4.9.1.1.2 fepLogging mcSpecの定義

FEPLoggingコンテナのメモリとCPUの設定は、mcSpecプロパティで指定できます。

例)

```
spec:
  fepLogging:
    mcSpec:
      limits:
        cpu: 500m
        memory: 700Mi
      requests:
        cpu: 200m
        memory: 512Mi
```

4.9.1.1.3 fepLogging restartRequiredの定義

新しい変更を適用するためにFEPLoggingを再起動する必要がある場合 (証明書のローテーション後など) 、restartRequiredフラグをtrueに設定すると、FEPLoggingコンテナを再起動できます。このフラグの既定値はfalseです。Podが再起動されると、このフラグはfalseに戻ります。

例)

```
spec:
  fepLogging:
    restartRequired: true
```

4.9.1.1.4 fepLogging scrapeIntervalおよびscrapeTimeoutの定義

FEPLoggingのscapeIntervalおよびscapeTimeoutプロパティはオプションです。これらのプロパティは、メトリクスのフェッチ間隔 (scrapeInterval) とリクエストのタイムアウトを設定するためにPrometheus Servicemonitorによって使用されます。

例)

```
spec:
  fepLogging:
    scrapeInterval: 30s
    scrapeTimeout: 30s
```

4.9.1.1.5 fepLogging elasticの定義

FEPLogging (Fluentd) からElasticsearchにログを転送するには、elasticプロパティを設定する必要があります。これはオプションのプロパティです。Elasticsearchサーバと証明書は、ユーザーによって設定されます。

Elasticsearchへのログ転送を設定するには、次のプロパティが必要です。

- authSecret
- host
- port
- logstashPrefix
- auditLogstashPrefix
- scheme
- sslVerify
- tls(sslVerifyがtrueに設定されている場合)

Elasticsearchサーバを設定し、そのホスト名とポートを使用します。

ここでtlsプロパティはオプションで、sslVerifyフラグと一緒に動作します。セキュア接続とtls検証を有効にするには、sslVerifyにtrueを設定し、有効なcertificateNameとcaNameを指定します。

Elastic SearchサーバのCA証明書を保持するElasticsearch caNameは必須です。

例)

```
  spec:
    fepLogging:
      elastic:
        authSecret:
          passwordKey: password
          secretName: elastic-auth
          userKey: username
        host: elastic-passthrough.apps.openshift.com
        logstashPrefix: postgres
        auditLogstashPrefix: postgres
        port: 443
        scheme: https
        sslVerify: false
        tls:
          certificateName: fluentd-cert
          caName: elastic-cacert
```

4.9.1.1.6 ElasticsearchのためのauthSecretの定義

authSecretは、Elasticsearch認証用のbase64形式のユーザー名とパスワードを含むシークレットです。

例)

```
kind: Secret
apiVersion: v1
metadata:
  name: elastic-auth
  namespace: my-namespace
data:
  password: OFBobzlyRUJWOGg1MkOxcXdaMUQ5bzQO
  username: ZWxhc3RpYw==
type: Opaque
```

4.9.1.1.7 fepLogging TLSの定義

FEPLoggingには、オプションのTLSプロパティがあります。セキュアな接続を介してFEPClusterからFEPLoggingインスタンスにログを転送する場合は、FEPClusterのTLS設定 (remoteLoggingセクション) 、およびFEPLoggingとPrometheusのTLS設定が必須です。fepLoggingまたはPrometheusだけでTLS設定を構成することはできません。

自己署名証明書を使用する場合、caNameはスキップできます。

例)

```
  spec:
    fepLogging:
      tls:
        certificateName: fluentd-cert
        caName: cacert
```

4.9.1.1.8 Prometheus TLSの定義

FEPLoggingとFEPCluster間のセキュアな接続が必要な場合、FEPLoggingとPrometheusのTLS設定は必須です。fepLoggingまたはPrometheusだけでTLSを設定することはできません。

自己署名証明書を使用する場合、caNameはスキップできます。

例)

```
spec:
  fepLogging:
   …
  prometheus:
    tls:
      certificateName: prometheus-cert
      caName: cacert
```

4.9.2 FEPClusterの設定

FEPClusterでログを有効にする方法について説明します。FEPClusterはログをリモートFluentd（FEPLogging）に転送する機能を提供し、FEPLoggingインスタンスは同じログをElasticsearch（オプション）とPrometheusに転送します。

4.9.2.1 FEP カスタムリソース - spec.fep.remoteLogging

remoteLoggingの設定で必要なパラメータを定義するには、fepの下にremoteLoggingセクションを追加する必要があります。

以下にテンプレートの例を示します。

```
spec:
  fep
    …
    remoteLogging:
      enable: true
      fluentdName: new-fep-logging
      tls:
        certificateName: fluentbit-cert
        caName: cacert
    …
```

remoteLoggingセクションで定義されているすべてのパラメータと設定内容について、以下に示します。

フィールド	必須/オプション	説明	変更可否
remoteLogging.enable	必須	ログ機能を有効にするには、trueを設定します。	不可
remoteLogging.fluentdName	必須	ログが転送されるFEPLogging CRの名前です。	可
remoteLogging.tls.certificateName	オプション	FluentbitのMTLS証明書を含むシークレット名	可
remoteLogging.tls.caName	オプション	SSL検証のためのFluentdのCacert	可
remoteLogging.image	オプション	remoteLogging用の Fluentbitイメージ	可
remoteLogging.pullPolicy	オプション	Fluenbitイメージのプルポリシー	可
remoteLogging.mcSpec.limits.cpu	オプション	FluentbitのCPU割り当て制限	可
remoteLogging.mcSpec.limits.memory	オプション	Fluentbitのメモリ割り当て制限	可
remoteLogging.mcSpec.requests.cpu	オプション	FluentbitのCPU割り当て要求	可

フィールド	必須/オプション	説明	変更可否
remoteLogging.mcSpec.requests.memory	オプション	Fluentbitのメモリ割り当て要求	可
remoteLogging.fluentbitParams.memBufLimit	オプション	FluentbitでMem_Buf_Limitを定義します。これは、このパラメータを使用するすべてのセクションに影響します。	可

4.9.2.1.1 remoteLogging enableおよびfluentdNameの定義

enableがtrueに設定されている場合、FEPClusterのログ監視機能が有効になります。

enableがtrueに設定されている場合、fluentdNameは必須フィールドです。ここには、FEPClusterがログを転送するFEPLogging CRの名前が記述されます。

enableがfalseに設定されている場合、FEPClusterのログ機能が有効になりません。

例)

```
fep:
  remoteLogging:
    enable: true
    fluentdName: new-fep-logging
```

ユーザーがログ監視機能を使用するために既存のFEPClusterを更新する場合は、FEPClusterのlog_destinationにcsvlogを設定する必要があります。新しいクラスタの場合は、すでに設定されています。

例)

```
fep:
  …
  remoteLogging:
    enable: true
    fluentdName: new-fep-logging
  …

fepChildCrVal:
  customPgParams:
     …
     log_destination = csvlog
     …
```

4.9.2.1.2 remoteLogging tlsの定義

FEPClusterがremoteLoggingにセキュア接続を使用する場合、TLSセクションは必須です。

TLSセクションで、SSL検証に使用される証明書と秘密鍵を含むシークレット名を指定します。

MTLS接続の場合、証明書を相互に検証するにはcaNameが必要です。

例)

```
fep:
  remoteLogging:
    enable: true
    fluentdName: new-fep-logging
    tls:
      certificateName: fluentbit-cert-secret
      caName: ca-cert
```

注意

Elasticsearchサーバはユーザーによって設定されます。オペレータによるFEPLoggingデプロイの一部ではありません。

4.9.2.1.3 remoteLogging imageの定義

imageプロパティは、デフォルトのFluentbitイメージとpullPolicy以外を指定するために使用されます。指定しない場合、オペレータが提供するデフォルトのイメージが使用されます。

例)

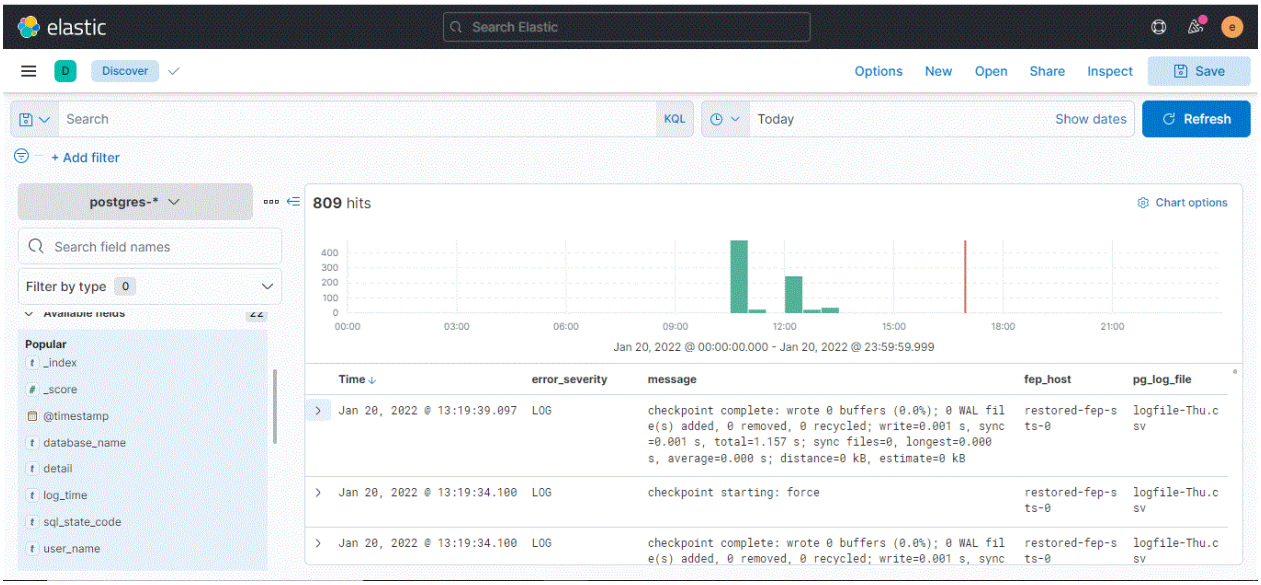
```
spec:
  fep:
    remoteLogging:
      image: 'quay.io/fujitsu/fujitsu-enterprise-postgres-fluentbit:ubi8-15-1.0'
      pullPolicy: IfNotPresent
```

4.9.3 FEPLoggingの操作

4.9.3.1 Elasticsearchへのログ転送

ユーザーがFEPLoggingカスタムリソースでElasticsearchを指定し、FEPClusterがそのFEPLoggingインスタンスにサーバログファイルと監査ログファイルを送信するように設定されている場合、それらのログはElasticsearchスタックまたはElastic Cloudに表示されます。ElasticsearchがKibanaで設定されている場合、ログはKibanaダッシュボードに表示されます。fep log csvフィールドを使用して、Kiabanaでさまざまなダッシュボードを作成することもできます。LogstashPrefixとauditLogstashPrefixは、特定のFEPLoggingインスタンスのログをフィルタするために使用されます。

ユーザーは、FEPログがリアルタイムに出力先に存在するかを確認することにより、FEPLogging機能が適切に構成されているかどうかを確認できます。



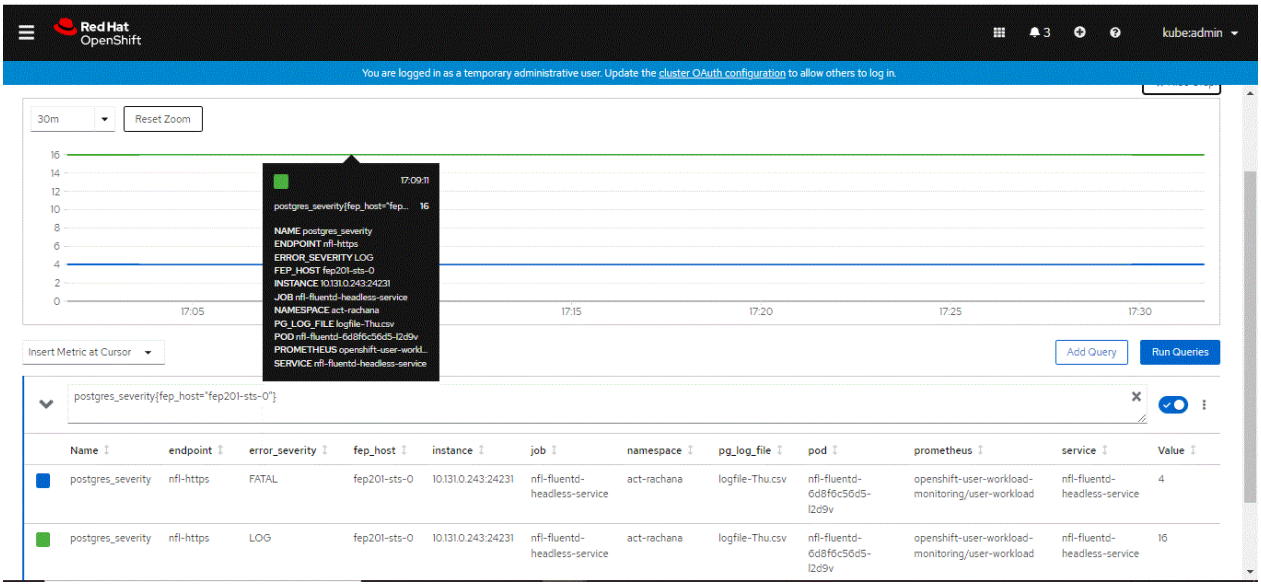
4.9.3.2 ログの重要度ベースのアラーム/メトリクス

FEPLogging機能は、postgresのログ重要度に基づいてアラーム/アラートを生成するためにも使用されます。ユーザがFEPLoggingカスタムリソースを作成している間、オペレーターは様々なpostgresのサービスメトリクスのリアルタイムカウントをOpenshiftで管理されているPrometheusに転送します。Openshiftで管理されているAlertmanagerは、このメトリクスカウンターにアクセスし、ユーザーはそれらを使用してアラート/アラームを作成できます。次の4つのデフォルトのアラートルールが、FEPLoggingの一部としてすでに作成されています。

- FEPLogErrorMessage
- FEPLogFatalMessage

- FEPLogPanicMessage
- FEPLogWarningMessage

デフォルトのスクレイピング間隔は30秒であるため、Prometheusは30秒ごとにpostgres_severityカウンターをスクレイピングします。ユーザーは、FEPLoggingカスタムリソースからこのスクレイピング間隔を変更できます。各スクレイピング間隔の後、postgres_severityカウンターで変更/増分が見つかった場合、アラートルールが実行されます。ユーザーは、Prometheusダッシュボードからいつでもpostgres_severityメトリックのカウントを確認できます。



4.9.3.3 監査ログの Elasticsearch への転送

監査ログをElasticsearch に転送するには、FEPClusterを更新して監査ログの作成を有効にします。

例)

```
spec:
  fep:
    fepChildCrVal:
      customPgAudit: |
        [output]
        logger = 'auditlog'
        log_directory = '/database/log/audit'
      customPgParams: |
        shared_preload_libraries='...,pgaudit'
        session_preload_libraries='...,pgaudit'
```

参考

監査ログの有効化については、“4.11 監査ログの自動運用”も参照してください。spec.fep.pgAuditLog.enableパラメータを設定することでも監査ログ機能を有効化できます。

4.9.4 制限事項

- ERROR、PANIC、FATAL、WARNINGを含むPostgreSQLのログレベルのみが監視されます。
- 外部fluentdは、ログの監視とログの転送には使用できません。
- ログ転送には外部Elasticsearchが必要です。

- ユーザーは、デプロイ時にFEPClusterとFEPLogging間のセキュアな接続が必要かどうかを決定する必要があります。展開後、接続をセキュアでない接続からセキュアな接続に切り替えることはできますが、セキュアな接続からセキュアでない接続に切り替えることはできません。
- FEPLoggingカスタムリソースを設定してから、FEPClusterのみが特定のFEPLoggingにログを転送できるようにしてください。それ以外の場合は、ログ機能が動作しません。
- ユーザーは、FEPClusterカスタムリソースでlog_destinationを設定する必要があります。

4.10 pgBadgerの設定

pgBadgerの設定方法について説明します。FEPClusterでは、定義されたスケジュールでpgbadgerレポートを作成し、そのレポートを外部のWebサーバにアップロードすることができます。

4.10.1 FEPカスタムリソース - spec.fep.pgBadger

フィールド	説明
pgBadger.schedules.create	レポートを作成してアップロードするための作成スケジュール
pgBadger.schedules.cleanup	コンテナに残されたレポートを削除するための クリーンアップスケジュール
pgBadger.options.incremental	デフォルトはfalseです。trueを設定すると、pgbadgerで増分レポートを作成します。
pgBadger.endpoint.authentication	エンドポイントにアクセスするための認証情報を含むシークレット Basic認証のみをサポートします。
pgBadger.endpoint.customCertificateName	customCertificate CRでのクライアント証明書の参照
pgBadger.endpoint.fileUploadParameter	Webサーバによって定義されたファイルアップロードパラメータ デフォルトはfile
pgBadger.endpoint.insecure	curl-insecureオプションと同じです。 デフォルトはfalse
pgBadger.endpoint.url	レポートファイルをアップロードするWebサーバのURL

4.10.2 pgBdager schedulesの定義

スケジュールは、Cron形式で記述されたジョブを定期的に作成および実行するために使用されます。

スケジュール形式が無効な場合、cronjobは作成されないため、pgBdagerレポートは作成されず、アップロードされません。

例)

```
pgBadger:
  schedules:
    cleanup: '10 * * * *'
    create: '50 * * * *'
```

4.10.3 pgBdager optionsの定義

incrementalオプションがfalseに設定されている場合、pgbadgerは通常のhtmlレポートを作成し、そのhtmlファイルをWebサーバにアップロードします。

incrementalオプションがtrueに設定されている場合、pgbadgerは増分レポートを作成し、zipファイルをWebサーバにアップロードします。

例)

```
  pgBadger:
    options:
      incremental: true
```

4.10.4 レポートをアップロードするための定義

Webサーバのurl

httpとhttpsの両方がサポートされています。

例)

```
  pgBadger:
    endpoint:
      url: 'https://webserver-svc:4443/cgi-bin/upload.php'
```

Webサーバ認証

Basic認証のみがサポートされています

Webサーバ認証を設定するには、以下を行います。

username:passwordからbase64でエンコードされたテキストを作成します

例)

```
$ echo -ne "myuser:mypass" | base64

amFzb253Omphc29udw==
```

シークレットを作成するために出力をbase64でラップします

例)

```
$ echo -ne "amFzb253Omphc29udw==" | base64

YW1GemIyNTNPbXBoYzI5dWR3PT0=
```

ラップされたテキストを使用してシークレットを作成します。キーは「basic_auth」である必要があります

例)

```
kind: Secret
apiVersion: v1
metadata:
  name: pgbadger-endpoint-auth
  namespace: fep-container-ct
data:
  basic_auth: YW1GemIyNTNPbXBoYzI5dWR3PT0=
type: Opaque
```

endpoint定義にシークレット名を追加します。

例)

```
  pgBadger:
    endpoint:
      authentication: pgbadger-endpoint-auth
```

Webサーバ証明書

Webサーバで証明書ファイルが必要な場合、FEPClusterクラスタはcustomCertificateカスタムリソースを提供して、証明書ファイルをコンテナにマウントします。

Webサーバで証明書を使用するには、以下を行います。

証明書ファイルと鍵ファイルに基づいてシークレットを作成します。

例)

```
oc create secret tls  webserver-cert --cert=webserver.pem --key=webserver.key
```

webserver.pemおよびwebserver.keyは、Webサーバにアクセスするための証明書ファイルです。

CA証明書に基づいてコンフィグマップを作成します。

例)

```
oc create configmap webserver-cacert --from-file=ca.crt=webca.pem
```

webca.pemは、WebサーバにアクセスするためのCA証明書ファイルです。

FEPClusterカスタムリソースでカスタム証明書を定義します。

例)

```
spec:
  fepChildCrVal:
    customCertificates:
      - userName: pgbadger-custom
       certificateName: webserver-cert
       caName: webserver-cacert
```

userNameはpgBadgerエントリ内の参照です。

certificateNameは、上記で作成したシークレットです。

caNameは、上記で作成したコンフィグマップです。

pgbadgerのendpointでカスタム証明書名を参照します。

例)

```
    pgBadger:
      endpoint:
        customCertificateName: pgbadger-custom
```

Webサーバへのセキュアでないアクセス

pgbadgerカスタムリソースは、セキュアな接続が必要でない場合に、Webサーバのendpointにオプションを提供します。

例)

```
    pgBadger:
      endpoint:
        insecure: true
```

ファイルアップロードのパラメータ

ファイルをWebサーバにアップロードするためのパラメータを指定します。このパラメータの値は、Webサーバの実装によって異なります。

例)

```
  pgBadger:
    endpoint:
      fileUploadParameter: uploadfile
```

curlコマンドおよびパラメータ

FEPClusterはcurlコマンドを使用して、生成されたレポートをWebサーバのエンドポイントにアップロードします。endpointセクションのCRは、curlコマンドパラメータに変換されます。次の表に、マッピングを示します。

curlコマンドのパラメータ	ユーザーの設定
[URL]	Endpointのurl
--cert	webserver.pem customCertificateNameで参照されるシークレットに含まれる
--key	webserver.key customCertificateNameで参照されるシークレットに含まれる
--cacert	webca.pem customCertificateNameで参照されるコンフィグマップに含まれる
--form “uploadfile=@/path/to/report”	EndpointのfileUploadParameter
--header "Authorization: Basic passxxx”	Endpointのauthentication configmap
--insecure	endpoint.insecure が「true」に設定されている場合

4.10.5 Webサーバへのファイルのアップデート

FEPClusterは、増分モードに従ってpgbadgerレポートをアップロードします。

増分モード	アップロードされたファイル名	例
True	[fep cluster name]-sts-[pod index].zip	pgbadger-test3-sts-0.zip pgbadger-test3-sts-1.zip
False	[fep cluster name]-sts-[pod index].html	pgbadger-test3-sts-0.html pgbadger-test3-sts-1.html

zipファイルには、pgbadgerインクリメンタルレポートのフォルダが含まれています。

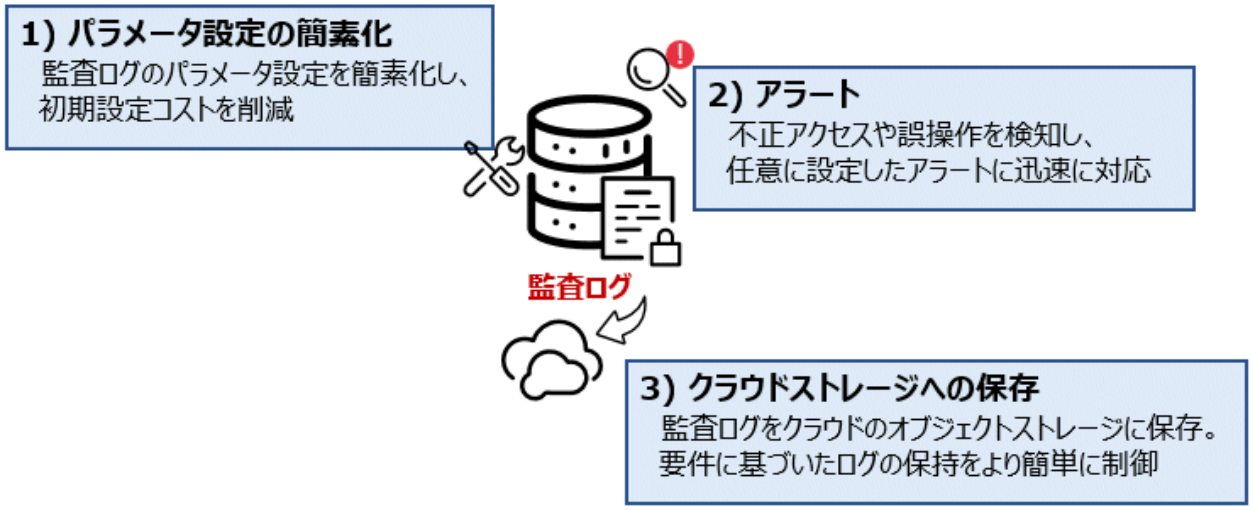
例)

```
¥database
  ¥log
    ¥pgbadger-report
      ¥[years]
        ¥[months]
        ¥[weeks]
```

- WebサーバはFEPClusterソリューションに含まれていません。
- Webサーバはビジネスロジックに従って、アップロードされたファイルを処理します。

4.11 監査ログの自動運用

監査などのセキュリティ要件を満たす運用を実現するための監査ログの運用が簡素化できます。



4.11.1 パラメータ設定の簡素化

監査ログのパラメータ設定を簡素化し、初期設定コストを削減します。必要な設定は、enableパラメータの有効化のみです。FEPサーバの起動時にpgaduitモジュールがロードされ、監査ログはログディレクトリに格納されます。

必要に応じて監査ログ設定ファイルをカスタマイズすることで、運用要件に応じた設定も可能です。

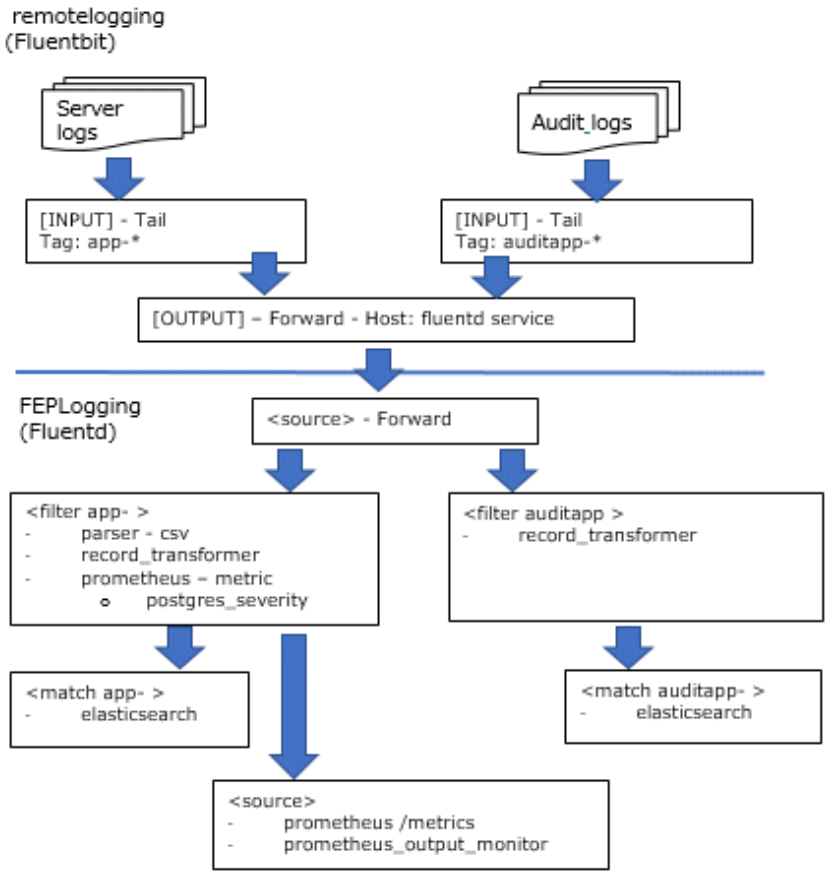
```
spec:
  fep:
    pgAuditLog:
      enable: true
```

4.11.2 アラート

あらかじめ設定されたアラート条件に従い、不正アクセスや誤操作を早期に検出し、迅速な対応が可能になります。

監査ログをremotelogging機能を用いてFluentdに送信し、Prometheusにはsqlstate条件に従ったアラート設定をすることで監査ログの監視を実現します。

アラートは、1分間のsqlstate(28P01) (無効なパスワード)の平均が50を超えたときにトリガーされます。



4.11.3 クラウドストレージへの保存

監査ログは、システムや業界標準のセキュリティポリシー要件に従い、長期間の保存が求められることがあります。しかし、ログの長期間保存はディスク管理やローテーション管理など複雑な運用の継続が必要になります。

そこで本機能では、監査ログをクラウドのオブジェクトストレージに保存することで、要件に基づいたログの保存を容易に制御することができます。

```
spec:
  fep:
    pgAuditLog:
      enable: true
      endpoint:
        protocol: s3
        url: s3://pgaudit/cluster1
        authentication: s3-secret
      schedules:
        upload: '30 * * * *'
```

4.12 鍵管理システムを利用した透過的データ暗号化

鍵管理システムを利用した透過的データ暗号化を構成する方法について説明します。

鍵管理システムを利用した透過的データ暗号化は、FEPClusterを最初に作成したときのみ構成できます。ユーザーは鍵管理システムを利用した透過的データ暗号化を既存のFEPClusterに設定することはできません。

4.12.1 認証情報の登録

4.12.1.1 KMIPサーバを使用する場合

KMIPサーバとの間のTLS通信に利用する証明書をSecretまたはConfigMapに保存します。

作成したSecretまたはConfigMapはFEPClusterカスタムリソースにリソース名を指定し、FEPコンテナにマウントします。

KMIPサーバに接続するためのクライアント証明書と秘密鍵を保存するSecretを作成します。

また、任意でルート証明書を保存するConfigMapを作成します。

下記の資格情報ファイルを利用して資格情報を登録する例を説明します。

```
kmip.pem   # KMIPサーバに接続するためのクライアント証明書
kmip.key   # 秘密鍵
myca.pem   # ルート証明書
```

クライアント証明書と秘密鍵を保存するためのSecret を作成します。

クライアント証明書と秘密鍵をマウントするときのファイル名は、それぞれtls.crtとtls.keyを指定します。

```
$ oc create secret generic kmip-cert --from-file=tls.crt=kmip.pem --from-file=tls.key=kmip.key -n kmip-demo
```

必要に応じて、ルート証明書を保存するためのConfigMapを作成します。

マウントするファイル名はca.crtを指定します。

```
$ oc create configmap kmip-cacert --from-file=ca.crt=myca.pem -n my-namespace
```

4.12.1.2 AWSの鍵管理サービスを使用する場合

AWSの鍵管理サービスに接続するために必要な認証情報、およびその他の設定をSecretとConfigMapに保存します。

認証情報とその他の設定をAWSのクライアントインターフェースが定める形式に従って記載したcredentialsとconfigの2つのファイルを準備します。credentialsファイルへのaccess_key_idとsecret_access_keyの指定は必須です。

下記の設定ファイルを利用して認証情報を登録する例を説明します。

```
credentials    # credentialsファイル
config         # configファイル
```

configファイルを保存するためのConfigMapを作成します。キー名にはconfigを指定します。ConfigMapの名前は任意です(ここではaws-kms-config)。

```
$ oc create configmap aws-kms-config --from-file=config=config -n my-namespace
```

credentialsファイルを保存するためのSecretを作成します。キー名にはcredentialsを指定します。Secretの名前は任意です(ここではaws-kms-credentials)。

```
$ oc create secret generic aws-kms-credentials --from-file=credentials=credentials -n my-namespace
```

参照

AWSのクライアントインターフェースの設定ファイルについては以下を参照してください。

https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-files.html

4.12.1.3 Azureの鍵管理サービスを使用する場合

Azureの鍵管理サービスに接続するために必要な認証情報をSecretに保存します。

利用できる認証方式はパスワードを使用した認証とクライアント証明書を使用した認証のどちらかです。

パスワードを使用した認証の場合、以下のようなSecretを定義するYAML形式のファイルを作成します。Secretの名前は任意です(ここではazure-key-vault-passphrase)。data.clientsecretにはbase64でエンコードされたパスワードを記載します。

```
kind: Secret
apiVersion: v1
metadata:
  name: azure-key-vault-passphrase
  namespace: my-namespace
data:
  clientsecret: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX=
type: Opaque
```

作成したYAMLファイルをもとにSecretを作成します。ここではazure-client-secret.yamlという名前のYAMLファイルを使用しています。

```
$ kubectl apply -f azure-client-secret.yaml -n my-namespace
```

クライアント証明書を使用した認証の場合、クライアント証明書ファイルと秘密鍵をSecretに保存します。

下記の証明書ファイルを利用してSecretを作成する例を説明します。

```
azuremycert.pem    # クライアント証明書と秘密鍵を含むPEMファイル
```

クライアント証明書を保存するためのSecretを作成します。キー名にはazure-key-vault.crtを指定します。Secretの名前は任意です(ここではazure-key-vault-secret)。

```
$ oc create secret generic azure-key-vault-secret --from-file=azure-key-vault.crt=azuremycert.pem -n my-namespace
```

4.12.2 FEPClusterカスタムリソースの設定

鍵管理システムを利用したTDEを有効化するためには、“spec.fepChildCrVal.customPgParams”と“spec.fepChildCrVal.sysTde”の設定が必要です。

4.12.2.1 spec.fepChildCrVal.customPgParamsの定義

fepChildCrVal.customPgParamsセクションでは、以下のパラメータを定義する必要があります。

shared_preload_libraries

「tde_kms」ライブラリをshared_preload_librariesのライブラリのリストに追加します。

例)

```
spec:
  fep:
   …
    fepChildCrVal:
      …
      customPgParams:
      shared_preload_libraries='pgx_datamasking,pg_prewarm,pg_stat_statements,tde_kms'
```

クラスタ作成後に「tde_kms」ライブラリを「shared_preload_libraries」リストから削除しないでください。

4.12.2.2 spec.fepChildCrVal.sysTdeの定義

sysTdeセクションをspec.fepChildCrValの下に追加して、鍵管理システムへの接続に必要なパラメータを定義します。sysTdeの下には、下記の2つのパラメータが定義されています。

- tdeType
- tdek

spec.fepChildCrVal.sysTde.tdeTypeの定義

sysTde自体はオプションのパラメータです(sysTdeが定義されていない場合、ファイルベースのキーストアを使用します)。ただし、sysTdeがユーザーによって定義されている場合は、sysTde.tdeTypeも定義する必要があります。

鍵管理システムを利用したTDEを構成する場合は、sysTde.tdeTypeを「tdek」に設定します。

例)

```
    sysTde:
      tdeType: tdek
```

spec.fepChildCrVal.sysTde.tdek.kmsDefinitionの定義

sysTde.tdeTypeを「tdek」に設定した場合、sysTde.tdekも定義する必要があります。

sysTde.tdek.kmsDefinitionには鍵管理システムの接続情報を定義します。ここに定義された情報を基にオペレーターがFujitsu Enterprise Postgresで利用される鍵管理システム接続情報ファイルを作成します。

kmsDefinitionには複数の鍵管理システムの情報を定義できます。typeにはその鍵管理システムの種別(kmip,awskms,azurekeyvaultのいずれか)を指定します。

例)

```
    sysTde:
      tdeType: tdek
      tdek:
        targetKmsName: kms_conninfo1
        kmsDefinition:
          - name: kms_conninfo1
            type: kmip
...
```

各パラメータの詳細は、“リファレンス”を参照してください。

“4.12.1 認証情報の登録”で作成したSecretまたはConfigMapの名前を、kmsDefinition配下の対応するパラメタに指定します。typeがawskmsの場合、profileには、AWS クライアントインターフェイスの設定ファイル内にあるプロファイルのうち、使用するプロファイルの名前を指定します。

例)

```
spec:
  fep:
   …
    fepChildCrVal:
      …
      sysTde:
      tdeType: tdek
      tdek:
        targetKmsName: kms_conninfo1
        targetKeyId: xxxyyyzzz
        kmsDefinition:
          -  name: kms_conninfo1
             type: kmip
             address: xxx.xxx.xxx.xxx
             port: 100
```

```
    authMethod: cert
    sslpassphrase: ssl-password
    cert:
      certificateName: kmip-cert
      caName: kmip-cacert
      sslcrlName: kmip-crl
```

spec.fepChildCrVal.sysTde.tdek.targetKeyId, spec.fepChildCrVal.sysTde.tdek.targetKmsNameの定義

kmsDefinitionで定義した鍵管理システムの中から、キーストアとして使用する鍵管理システムの名前（name）をsysTde.tdek.targetKmsNameに指定します。sysTde.tdek.targetKeyIdには、マスタ暗号化キーとして使用するその鍵管理システム内にある暗号化キーの鍵IDを指定します。

4.13 ホットスタンバイ構成でのディザスタリカバリ

ホットスタンバイ構成でのディザスタリカバリを実現することで、災害発生時により迅速に業務システムを復旧することができます。本機能には、以下の2つの方式があります。

継続的リカバリ方式

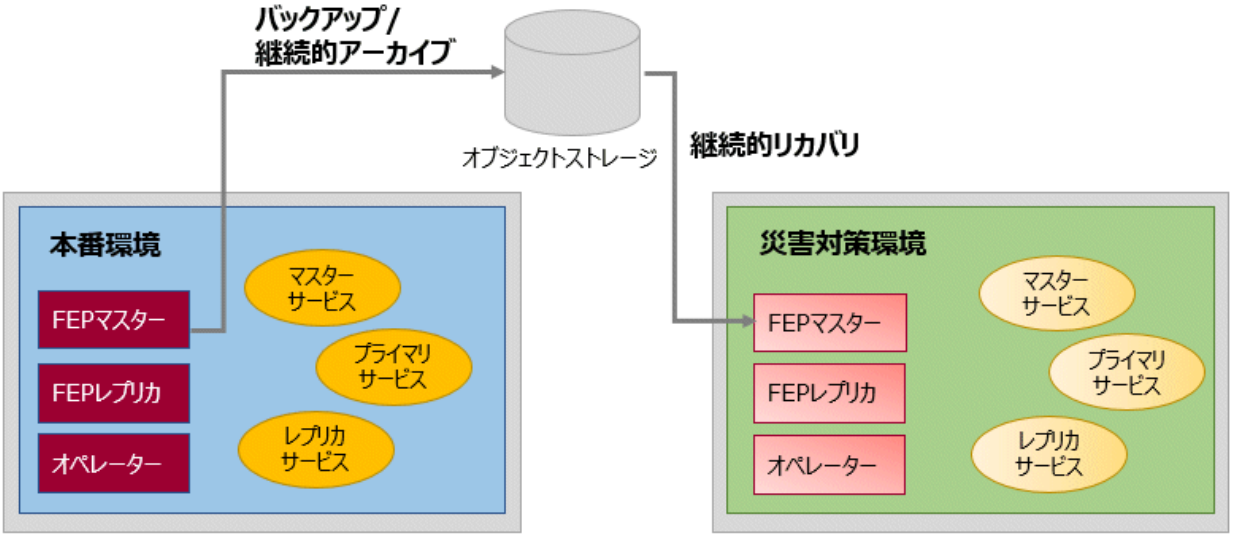
本番環境のコンテナ環境と災害対策環境のコンテナ環境を作成します。本番環境のデータをオブジェクトストレージに格納し、災害対策環境へ継続的にリストアします。この方式ではバックアップ/リストア方式と比較して迅速な復旧が可能ですが、定期的に実行されるバックアップタイミングによっては、RPO(Recovery Point Objective, 目標復旧時間)が大きくなってしまいます。

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に、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、データベース本体のストリーミングレプリケーション方式を使用します。この方式ではバックアップ/リストア方式と比較して迅速な復旧が可能になりつつ、継続的リカバリ方式と比較してRPOが小さくなり、リアルタイムでのデータの同期が可能になります。しかし、ストリーミングレプリケーション方式のためのネットワーク設定が必要であるため管理コストが大きく、本番環境のデータベースの性能にわずかですが影響が生じます。

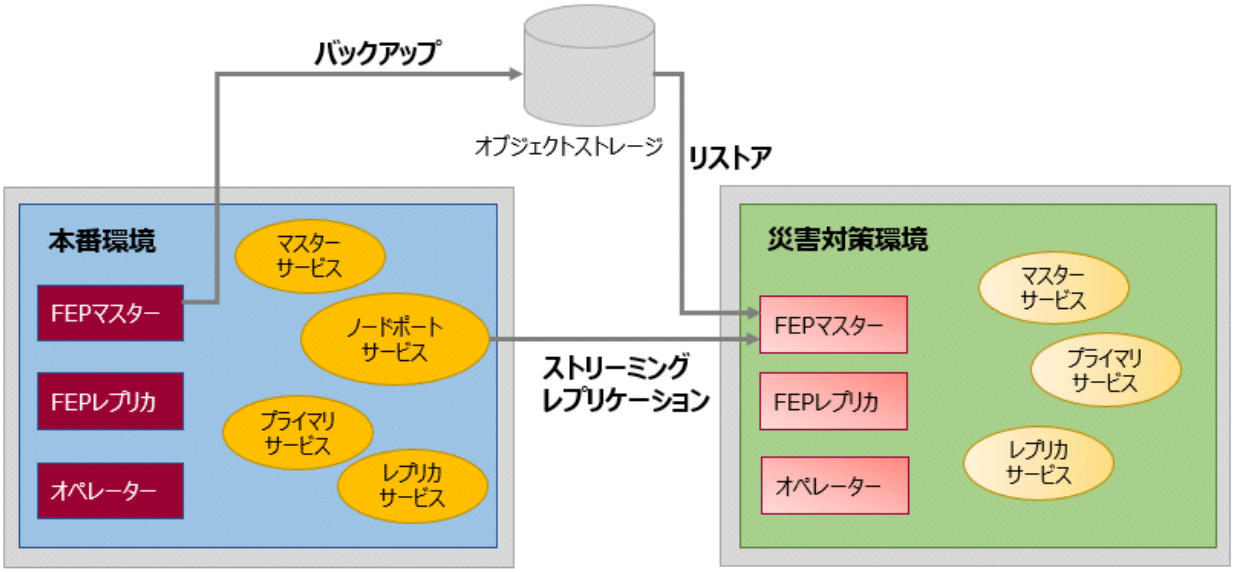
4.13.1 継続的リカバリ方式

継続的リカバリ方式では、本番環境と災害対策環境との同期にオブジェクトストレージを使用します。想定する災害の範囲に対して安全と思われる地域にあるオブジェクトストレージを指定します。



4.13.2 ストリーミングレプリケーション方式

ストリーミングレプリケーション方式では、本番環境のデータベースと災害対策環境のデータベース間において、直接的なデータの同期を実現します。



4.13.3 ホットスタンバイ構成の定義

ホットスタンバイ構成の継続的リカバリ方式およびストリーミングレプリケーション方式の配備手順を説明します。

4.13.3.1 継続的リカバリ方式の定義

本番環境のFEPClusterのカスタムリソース定義には、ホットスタンバイ構成のみで使用するパラメタなどは存在しません。継続的リカバリ方式を利用する場合には、災害対策環境でのFEPClusterのカスタムリソースを下記のように定義します。

```
apiVersion: fep.fujitsu.io/v2
kind: FEPCluster
metadata:
 …
spec:
  fep:
    standby:
      enable: true
      method: archive-recovery
      pgBackrestConf: |
        [global]
        log-path=/database/log/backup
        repo1-type=azure
        repo1-path=< Backup path of primary/ cluster from which data is to be restored>
        repo1-azure-account=<my storage account>
        repo1-azure-container=fepbackups
        repo1-azure-key=<my storage account key >
…
```

4.13.3.2 ストリーミングレプリケーション方式の定義

ストリーミングレプリケーション方式を利用する場合は、下記のように定義します。

```
apiVersion: fep.fujitsu.io/v2
kind: FEPCluster
metadata:
...
spec:
  fep:
    standby:
      enable: true
      method: streaming
```

```
      streaming:
        host: <LoadBalancer IP>
        port: 27500
      pgBackrestConf: |
        [global]
        log-path=/database/log/backup
        repo1-type=azure
        repo1-path=< Backup path of primary/ cluster from which data is to be restored>
        repo1-azure-account=<my storage account>
        repo1-azure-container=fepbackups
        repo1-azure-key=<my storage account key >
...
```

ストリーミングレプリケーションでは、FEPClusterCRを上記のように定義しますが、別途LoadBalancerを配備する必要があります。

```
kind: Service
apiVersion: v1
metadata:
  name: my-fep-internal-svc
  namespace: sample-namespace
  annotations:
    service.beta.kubernetes.io/azure-load-balancer-internal: 'true'
spec:
  ports:
    - protocol: TCP
      port: 27500
  type: LoadBalancer
  selector:
    app: <my-fep-cluster>-sts
    feprole: master
```

4.13.4 FEPClusterカスタムリソースの定義

ホットスタンバイ構成を実現するために必要な、災害対策環境のFEPClusterCRのパラメータを以下に示します。パラメータの詳細については、“リファレンス”の“FEPClusterパラメータ”を参照してください。

- spec.fep.standby.enable
- spec.fep.standby.method
- spec.fep.standby.pgBackrestConf
- spec.fep.standby.streaming.host
- spec.fep.standby.streaming.port

4.14 scram-sha-256認証を使用したクライアント認証の有効化

scram-sha-256認証を使用したクライアント認証を有効化する手順を説明します。

4.14.1 FEPサーバコンテナにおけるscram-sha-256認証を使用したクライアント認証の有効化

新規に構築したFEPサーバコンテナでscram-sha-256認証を有効化するには、以下を設定したFEPClusterカスタムリソースを配置してください。

4.14.1.1 spec.fepChildCrVal.customPgParamsの定義

password_encryptionにscram-sha-256を指定します。

```
password_encryption = 'scram-sha-256'
```

設定を有効にするには、FEPサーバコンテナ内のデータベースを再起動する必要があります。

4.14.1.2 spec.fepChildCrVal.customPgHbaの定義

scram-sha-256認証を許可するエントリを追加してください。

scram-sha-256認証を許可するためには、METHODにscram-sha-256を指定します。

設定を有効にするには、FEPサーバコンテナ内のデータベースを再起動する必要があります。

4.14.2 FEPpgpool2コンテナにおけるscram-sha-256認証を使用したクライアント認証の有効化

新規に構築したFEPpgpool2コンテナでscram-sha-256認証を有効化するための手順を説明します。

4.14.2.1 scram-sha-256認証の有効化に必要なリソースの作成

scram-sha-256認証の有効化に必要なリソースを作成します。

1. 暗号化キー用シークレットを作成

   scram-sha-256認証に使用するパスワードを記載した暗号化キー用シークレットpgpoolkeySecretを作成します。
   このとき、キーはpgpoolkeyとし、バリューには暗号化に使用するパスワードをbase64で暗号化した値を指定します。

```
apiVersion: v1
kind: Secret
metadata:
  name: scrampgpoolkey-secret
type: Opaque
data:
  pgpoolkey: cGdwb29sa2V5cGFzc3dvcmQ=
```

   シークレット作成後、FEPPgpool2 カスタムリソースに作成したシークレット名を記載します。これにより、作成されたシークレットがFEPpgpool2コンテナにボリューム内のファイルとしてマウントされ、暗号化に使用するパスワードがFEPpgpool2コンテナに引き渡されます。

   scram-sha-256認証を使用し、かつ、シークレットを作成しない場合、オペレーターがFEPPgpool2カスタムリソースの情報を基に、以下のシークレットを自動で作成します。

```
apiVersion: v1
kind: Secret
metadata:
  name: "{{spec.name}}-feppgpool2-pgpoolkey"
type: Opaque
data:
  pgpoolkey: K1kxOVZxKzRrdWluT3A2UHNQMzcwcUJuOUZ2UUoxUklNMms2cktIY1NkekFOemZBYkhjZDFadG5VR3ZtTVR6Uw==
```

   バリューにはオペレーターがランダムで生成したパスワードをbase64で暗号化した値が設定されます。この場合のシークレット名は、“{{spec.name}}-feppgpool2-pgpoolkey”となります。

   参考

   scram-sha-256暗号化では暗号化キーを使用します。この暗号化キーはpool_passwdファイルにAES暗号化パスワードが保存されていた場合、パスワードを復号化するためにFEPpgpool2コンテナで使用します。そのため、暗号化キーと復号キーは同じものである必要があります。

2. データベースユーザー情報用シークレットを作成

   FEPpgpool2コンテナのクライアント認証に使用するユーザー/パスワードは、FEPサーバコンテナのデータベースで使用するデータベースユーザーと同様のものである必要があります。

FEPpgpool2コンテナに、FEPサーバコンテナのデータベースユーザーのユーザー名とパスワードの情報を引き渡すためのデータベースユーザー情報用シークレットuserinfoSecretを作成します。
ユーザー名とパスワードは下記のような形式でシークレットに記載します。

```
apiVersion: v1
kind: Secret
metadata:
  name: scramuserinfo-secret #任意の名前を指定
type: Opaque
data:
  user1: dXNlcjFwYXNzd2Q=
  user2: dXNlcjJwYXNzd2Q=
```

オペレーターが自動で作成するデータベースユーザーについては、オペレーターがFEPCluster カスタムリソースおよびFEPサーバコンテナからデータベースユーザー名とパスワードの情報を取得するため、シークレットの作成は不要です。

scram-sha-256認証を使用し、かつ、シークレットを作成しない場合、オペレーターが以下のシークレットを自動で作成します。

```
apiVersion: v1
kind: Secret
metadata:
  name: ” {{spec.name}}-feppgpool2-userinfo”
type: Opaque
data:
```

この場合のシークレット名は、“{{spec.name}}-feppgpool2-userinfo”となります。

4.14.2.2 FEPPgpool2カスタムリソースの編集

scram-sha-256認証を有効化するには、FEPPgpool2カスタムリソースを以下の手順で編集し、配置してください。

1. シークレットの情報を設定

   FEPPgppol2カスタムリソースのspec.clientAuthMethodにscramを指定します。
   また、暗号化キー用シークレット(spec.scram.pgpoolkeySecret)に“4.14.2.1 scram-sha-256認証の有効化に必要なリソースの作成”の手順1で作成したシークレット名を記載します。同様に、データベースユーザー情報用シークレット(spec.scram.userinfoSecret)に“4.14.2.1 scram-sha-256認証の有効化に必要なリソースの作成”の手順2で作成したシークレット名を記載します。

   ```
   spec:
     clientAuthMethod: scram
     scram:
       pgpoolkeySecret: scrampgpoolkey-secret
       userinfoSecret: scramuserinfo-secret
   ```

   これにより、作成されたシークレットがFEPpgpool2コンテナにボリューム内のファイルとしてマウントされ、ユーザー名とパスワードの情報がFEPpgpool2コンテナに通知されます。

2. spec.customhbaを編集

   FEPPgpool2カスタムリソースのspec.customhbaフィールドを編集し、scram-sha-256認証用のエントリを追加します。

3. pgpool.confを編集

   FEPPgpool2カスタムリソースのspec.customparamsフィールドを編集し、認証設定に関連するパラメータを編集します。

   ```
   enable_pool_hba=true
   ```

4.14.3 既存のFEPサーバコンテナおよびFEPpgpool2コンテナにおけるscram-sha-256認証を使用したクライアント認証の有効化

既存のFEPサーバコンテナおよびFEPpgpool2コンテナで、scram-sha-256認証を使用したクライアント認証を有効化する方法を説明します。

FEPサーバコンテナについては、“4.14.1 FEPサーバコンテナにおけるscram-sha-256認証を使用したクライアント認証の有効化”を参照してください。このとき、ユーザーが作成したデータベースユーザーについて、すでにmd5で暗号化されたパスワードが設定されている場合は、scram-sha-256で暗号化されたパスワードに変更する必要があるため、パスワードの再設定を行ってください。

FEPpgpool2コンテナについては、“4.14.2 FEPpgpool2コンテナにおけるscram-sha-256認証を使用したクライアント認証の有効化”を参照してください。このとき、既存のFEPpgpool2カスタムリソースが配備されている場合は、FEPpgpool2カスタムリソースを削除してください。FEPpgpool2カスタムリソースで指定したシークレット、または、オペレーターが自動で作成したシークレットの内容を変更する場合、FEPpgpool2カスタムリソースを削除する必要はありません。

第5章 デプロイ後の運用

本章ではデプロイ後の運用について説明します。

5.1 FEPクラスタへの接続方法

OpenShiftシステムの同一プロジェクト内から接続する場合

FEPClusterおよびFEPPgpool2に同一プロジェクト内から接続するにはサービスリソースを利用します。

サービスリソースはコンテナと通信するための単一のエンドポイントを提供します。

サービスリソースは以下の命名規則で作成されます。

FEPClusterサービス

- <FEPCluster名>-primary-svc
- <FEPCluster名>-replica-svc
- <FEPCluster名>-headless-svc

FEPPGPool2サービス

- <FEPPgpool2名>-feppgpool2-svc

FEPClusterコンテナおよびFEPpgpool2コンテナのサービスリソースを確認する例

```
$ oc get all
```

リソースタイプがサービスである場所を確認します(「svc /」で始まります)。

oc get svcコマンドでも確認できます。以下はその例です。

```
$ oc get svc
NAME                              TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)                       AGE
<FEPCluster名>-headless-svc       ClusterIP   None             <none>        27500/TCP,25001/TCP           24h
<FEPCluster名>-primary-svc        ClusterIP   xxx.xxx.xxx.xxx  <none>        27500/TCP,25001/TCP           24h
<FEPCluster名>-replica-svc        ClusterIP   yyy.yyy.yyy.yyy  <none>        27500/TCP,25001/TCP           24h
<FEPPgpool2名>-feppgpool2-svc     NodePort    zzz.zzz.zzz.zzz  <none>        9999:31707/TCP,9998:31906/TCP 24h
```

FEPpgpool2コンテナにアクセスする例

```
$ psql -h <FEPPgpool2名>-feppgpool2-svc  -p 9999 -c "select version();"
```

OpenShiftシステムの外部から接続する場合

ClusterIPを使用してサービスを自動的に作成し、デプロイされたコンテナに接続します。OpenShiftシステムの内部ネットワークからFEPまたはFEPpgpool2サービスに接続できます。OpenShiftシステムの外部からアクセスするには、OpenShiftノードのアドレスを知っている必要があります。

例えば、OpenShiftシステムの外部で実行されているが内部ネットワークの一部であるアプリケーションサーバからFEPpgpool2コンテナにアクセスできます。

本手順を実施するためには、OpenShiftまたはKubernetesクラスタのノードに対する“get, list”操作が可能な権限が必要です。

OpenShiftでノードIPをチェックする方法の例

```
$ oc get nodes
NAME                                    STATUS   ROLES    AGE    VERSION
openshiftcluster1-cmfv8-master-0        Ready    master   370d   v1.19.0+4c3480d
openshiftcluster1-cmfv8-master-1        Ready    master   370d   v1.19.0+4c3480d
openshiftcluster1-cmfv8-master-2        Ready    master   370d   v1.19.0+4c3480d
$ oc describe nodes openshiftcluster1-cmfv8-master-0 | grep IP
 InternalIP: 10.0.2.8
```

FEPpgpool2コンテナのサービスリソースを確認する例

```
$ oc get all
```

リソースタイプがサービスである場所を確認します(「svc /」で始まります)。

oc get svcコマンドでも確認できます。以下はその例です。

```
$ oc get svc
NAME                          TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)                         AGE
<FEPPgpool2名>-feppgpool2-svc  NodePort    172.30.248.12    <none>        9999: 30537/TCP, 9998: 30489/TCP 2m5s
```

FEPpgpool2コンテナにアクセスする例

```
$ psql -h 10.0.2.8 -p 30537 -c "show pool_nodes"
```

5.2 構成の変更

FEPClusterの構成の変更について説明します。

FEPClusterの一覧表示

Kubernetesコマンド: kubectl get FEPClusters (-A)

この操作では、名前空間内のすべてのFEPClusterが一覧表示されます。または、-Aオプションが指定されている場合は、すべての名前空間内のすべてのFEPClusterが一覧表示されます。

デフォルトの出力形式:

フィールド	値	説明
NAME	.metadata.name	FEPClusterの名前
AGE	経過時間	クラスタが作成されてからの経過時間を示します

例)

```
# kubectl get fepclusters -A

NAMESPACE       NAME            AGE
namespace1      ns1fep1         21h
namespace2      ns2fep2         22h
```

FEPClusterの更新

Kubernetesコマンド: kubectl apply -f <new_spec>

以下を更新することができます。

カスタムリソースの仕様	変更内容
.spec.fep.instances: *n*	クラスタ内のノードの数を*n*に増やします。
.spec.fep.image.image: 'quay.io/fujitsu/fujitsu-enterprise-postgres-15-server:ubi8-15-1.1'	FEPイメージを「ubi8-15-1.1」にマイナーアップグレードします。
spec.fepChildCrVal.backup.image.image: 'quay.io/fujitsu/fujitsu-enterprise-postgres-15-backup:ubi8-15-1.1'	バックアップイメージを「ubi8-15-1.1」にマイナーアップグレードします。

これは、fepセクションの値の動作にのみ影響します。

すべてのパラメータはFEPClusterカスタムリソースから更新することができます。

FEPClusterの削除

Kubernetesコマンド: kubectl delete FEPCluster <cluster_name>

この操作により、FEPClusterとそれに関連するすべての子カスタムリソース(FEPVolume、FEPConfig、FEPCert、FEPUser)およびリソースが削除されます。

注意

FEPClusterを削除すると、バックアップおよびアーカイブされたWALボリュームを含む、クラスタに関連付けられているすべてのPVが削除されます(あらかじめ作成したPVやAWS S3を使用している場合を除く)。これは復元できないアクションです。

5.3 FEPClusterのリソース変更

5.3.1 CPUとメモリの割り当てリソースの変更

FEPClusterによって作成されたPodに割り当てられたCPUとメモリのリソース変更方法を説明します。

これにより、カスタムリソースを通してPodを垂直方向にスケーリングすることができます。

CPUとメモリのリソースを変更するためには、FEPClusterカスタムリソースのspec.fep.mcSpecセクション(注)を修正し、変更を適用してください。

変更が適用されると、レプリカサーバを新しいリソース設定値で再起動します。レプリカサーバが複数台ある場合は、1台ずつ順番に再起動します。すべてのレプリカサーバが再起動されると、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再起動します。これにより、業務停止を最小限にしてすべてのサーバのリソース設定値を変更することができます。

注)本セクションを修正することでFEPサーバコンテナがスケールアップされます。他コンテナのリソースセクションについては、“リファレンス”の“FEPClusterパラメータ”を参照してください。

5.3.2 PVCのサイズ変更

FEPClusterによって作成されたPodに割り当てられたPVCのサイズ変更方法を説明します。

これにより、カスタムリソースを通してPodに割り当てられたボリュームのサイズを拡張することができます。

PVCのサイズを変更するためには、FEPClusterカスタムリソースのspec.fepChildCrVal.storageセクションに含まれる各ボリュームのサイズを修正し、変更を適用してください。これらの変更が、FEPClusterによって作成されたPodに割り当てられたすべてのPVCに適用されます。

注意

- PVCのサイズ変更は、拡張のみ可能です。
- PVCのサイズ変更は、StorageClassがサイズの動的変更をサポートしている場合に限ります。

- StorageClassがPVCのサイズ変更をサポートしていない場合は、FEPRestoreカスタムリソースを利用して新規FEPClusterを作成することで、PVCのサイズを変更してください。詳細は、“リファレンス”の“FEPRestoreカスタムリソースパラメータ”を参照してください。

5.4 FEPPGPool2の構成の変更

FEPPGPool2の構成の変更について説明します。

FEPPGPool2の一覧表示

Kubernetesコマンド: kubectl get FEPPGPool2 (-A)

この操作では、名前空間内のすべてのFEPPGPool2が一覧表示されます。または、-Aオプションが指定されている場合は、すべての名前空間内のすべてのFEPPGPool2が一覧表示されます。

Default output format:

フィールド	値	詳細
Name	.metadata.name	pgpool2の名前

例)

```
# kubectl get feppgpool2 -A

NAMESPACE       NAME
namespace1      fep1-pgpool2
namespace2      fep2-pgpool2
```

FEPPGPool2の削除

Kubernetesコマンド: kubectl delete FEPPGPool2 <pgpool2_name>

この操作により、FEPPGPool2が削除されます。

FEPPGPool2の更新

Kubernetesコマンド: kubectl apply -f <new_spec>

“リファレンス”の“FEPPgpool2カスタムリソースパラメータ”を参照して、更新するパラメータを指定します。以下のパラメータのみ指定可能です。

カスタムリソースの仕様	変更内容
.spec.count: n	クラスタ内のノードの数を*n*に増やします。
.spec.serviceport	Pgpool-IIに接続するためのTCPポートを変更します。
.spec.statusport	PCPプロセスに接続するためのTCPポートを変更します。
.spec.limits.cpu	CPUの制限を変更します。
.spec.limits.memory	メモリの制限を変更します。
.spec.requests.cpu	CPUのリクエストを変更します。
.spec.requests.memory	メモリのリクエストを変更します。
.spec.fepclustername	接続するFepClusterを変更します。
.spec.customhba	pool_hba.confファイルを変更します。
.spec.customparams	pgpool2パラメータを変更します。
.spec.custompcp	pcp.confファイルを変更します。
.spec.customsslkey	秘密鍵の内容を変更します。

カスタムリソースの仕様	変更内容
.spec.customsslcert	x509公開鍵証明書の内容を変更します。.
.spec.customsslcacert	CAルート証明書の内容をPEM形式で変更します。

一部のcustomparamsパラメータ、customhbaおよびcustompcpでは、pgpool2を再起動する必要があります。

Kubernetesコマンド: Kubectl apply -f <new_spec>

「pgpool2_restart」アクションタイプでは、ユーザーが再起動するpgpool2の名前を指定する必要があります。

以下のように、FEPActionカスタムリソースのtargetPgpool2NameセクションでFEPPGPool2カスタムリソースのmetadata.Nameを指定します。

```
spec:
    targetPgpool2Name: fep1-pgpool2
    fepAction:
        type: pgpool2_restart
```

FEPPGPool2を更新すると、FEPPGPool2のPodが再起動されます。複数のFEPPGpool2で構成されている場合、順番に再起動されます。
接続が切断されるため、アプリケーションは接続を再接続するように設計する必要があります。

scram-sha-256認証を使用したクライアント認証に使用する各リソースの更新

新規データベースユーザーのパスワードを設定

データベースに新規ユーザーを追加した場合に、ユーザーとパスワードをFEPpgpool2コンテナに通知する必要があります。
この場合、既存のユーザーの情報に加えて、以下のように新規ユーザーの情報をシークレットに追記してください。

```
apiVersion: v1
kind: Secret
metadata:
  name: test-secret
type: Opaque
data:
  user_name_1:cGFzc3dvcmRfMQ==
  user_name_2:cGFzc3dvcmRfMg==
  user_name_new:cGFzc3dvcmRfbmV3
```

シークレットの更新後、FEPpgpool2コンテナは自動でパスワードファイルpool_passwdの内容を更新します。
オペレーターが自動で作成したシークレットを編集して新規ユーザーの情報を追加する場合、postgresキーのエントリを削除してください。
postgresキーのエントリが残存している場合、新規ユーザーの情報が反映されません。

既存のデータベースユーザーのパスワードを更新

FEPClusterコンテナでは、パスワード期限などの理由からデータベースユーザーのパスワードが更新される場合があります。
この場合、既存のユーザーの情報は維持したまま、更新したパスワードについて、以下のようにユーザー情報用シークレットを更新してください。

```
apiVersion: v1
kind: Secret
metadata:
  name: test-secret
type: Opaque
data:
  user_name_1:cGFzc3dvcmRfMQ==
  user_name_2:cGFzc3dvcmRfMl9uZXc=
```

シークレットの更新後、FEPpgpool2コンテナは自動でパスワードファイルpool_passwdの内容を更新します。

暗号化用キーファイルpgpoolkeyを更新

暗号化用キーファイルpgpoolkeyを更新したい場合、暗号化キー用シークレットpgpoolkeySecretに定義しているシークレットの内容を更新します。

```
apiVersion: v1
kind: Secret
metadata:
  name: pgpoolkey-secret
type: Opaque
data:
  pgpoolkey:bmV3LXBncG9vbGtleXBhc3N3b3Jk
```

暗号化用キーファイルpgpoolkeyを更新する場合は、更新されたpgpoolkeyを使用して、再度、scram-sha-256による暗号化を行う必要があります。また、暗号化キー用シークレットの更新後は、pgpool2を再起動してください。
シークレットの更新後、FEPpgpool2コンテナは自動で、更新されたpgpoolkeyを使用して、パスワードファイルpool_passwdの内容を更新します。

5.5 オペレーターからのバックアップのスケジュール

運用状態の確認

以下の例に示すように、バックアップに関する情報は、FEPバックアップコンテナでコマンドを実行することで確認できます。

```
$ oc exec pod/fepserver-XXXXX -c FEPbackup -- pgbackrest info
stanza: fepbackup
    status: ok
    cipher: none

    db (current)
        wal archive min/max (12-1): 000000010000000000000001/000000010000000000000005

        full backup: 20201125-025043F
            timestamp start/stop: 2020-11-25 02:50:43 / 2020-11-25 02:50:52
            wal start/stop: 000000010000000000000003 / 000000010000000000000003
            database size: 31.7MB, backup size: 31.7MB
            repository size: 3.9MB, repository backup size: 3.9MB

        incr backup: 20201125-025043F_20201125-025600I
            timestamp start/stop: 2020-11-25 02:56:00 / 2020-11-25 02:56:02
            wal start/stop: 000000010000000000000005 / 000000010000000000000005
            database size: 31.7MB, backup size: 24.3KB
            repository size: 3.9MB, repository backup size: 619B
            backup reference list: 20201125-025043F
```

FEPBackupの更新

Kubernetesコマンド: kubectl apply -f <new_spec>

“リファレンス”の“FEPBackup子カスタムリソースパラメータ”を参照して、更新するパラメータを指定します。以下のパラメータのみ指定可能です。

カスタムリソースの仕様	変更内容
spec.schedule.num	登録されているバックアップスケジュールの数を変更します。
spec.scheduleN.schedule	スケジュールされたバックアップ時間を変更します。
spec.scheduleN.type	スケジュールされたバックアップタイプを変更します。
spec.pgBackrestParams	pgBackRestパラメータを変更します。

カスタムリソースの仕様	変更内容
spec.scheduleN.repo	spec.pgBackrestParamsに複数のレポジトリを指定している場合、バックアップデータを格納するレポジトリを選択します。 省略時は1です。

注意

- バックアップ中に行われた変更は、次のバックアップから反映されます。
- バックアップスケジュールを変更しても、アプリケーションには影響しません。
- 下記の更新操作を実施した場合は、更新後に必ずバックアップを取得するようにしてください。
  - pgx_set_master_keyによりマスタ暗号化キーを更新した場合
  - 透過的データ暗号化の暗号化パスフレーズを更新した場合（FEPClusterカスタムリソースのtdepassphraseパラメータにより更新可能）

5.6 MTLS設定の変更

5.6.1 認証のローテーション

すべての証明書は、時間によって制限されています。証明書は、ある時点で、更新する必要があります。証明書がライフサイクルの3/4に達したとき、または侵害された場合はできるだけ早く証明書を更新することをお勧めします。証明書が更新されたら、FEPサーバコンテナ内で証明書をローテーションする必要があります。現時点では、FEPサーバコンテナは自動証明書ローテーションをサポートしていません。どの証明書が更新されたかに応じて、それを処理するためのさまざまな手順があります。

Patroni証明書のローテーション

Patroni証明書が更新されると、新しい証明書を取得するために、FEPサーバコンテナのすべてのPodを再デプロイする必要があります。FEPClusterに停止時間があります。

FEPサーバ証明書のローテーション

FEPサーバ証明書が更新されると、FEPActionカスタムリソースを使用してデータベースのリロードをトリガでき、FEPサーバはサービスを中断することなく新しい証明書を取得できます。

クライアント証明書のローテーション

クライアント証明書のいずれかが更新されると、FEPサーバコンテナは、次にFEPサーバへの接続を確立するときに、内部で新しい証明書を使用します。ただし、サービスの予期しない中断を回避するために、すべてのPodをできるだけ早く再デプロイすることをお勧めします。

5.7 モニタリング

モニタリングとは、過去のデータポイントを収集することです。このデータポイントを使用して、アラートの生成（異常の場合）、データベースの最適化、問題が発生した場合（データベースの障害など）に備えた予防策を講じます。

データベースをモニタリングする5つの主な理由があります。

1. 可用性

   データベースを稼働しなければアプリケーションは動作しません。データベースの可用性は、業務の継続性に直接影響します。

2. システムの最適化

   モニタリングは、システムのボトルネックを特定するのに役立ち、ユーザーは問題を解決するかどうかを確認するためにシステムに変更を加えることができます。たとえば、システムに非常に高い負荷をかける状況がある場合には、パラメータの値を変更し、システムを最適化することが必要です。

3. **パフォーマンス問題の特定**

プロアクティブなモニタリングは、将来のパフォーマンスの問題を特定するのに役立ちます。データベース側からは、肥大化、実行速度の遅いクエリ、テーブルとインデックスの統計、またはバキュームが追いつかないことに関連している可能性があります。

4. **業務プロセスの改善**

データベースユーザーごとに、異なるニーズと優先順位があります。システム（負荷、ユーザーアクティビティなど）を知っていると、顧客のタスク、レポート、またはダウンタイムに優先順位を付けるのに役立ちます。モニタリングは、業務プロセスの改善に役立ちます。

5. **キャパシティプランニング**

ユーザーまたはアプリケーションの増加は、システムリソースの増加を意味します。「より多くのディスクスペースが必要か？」、「新しい参照レプリカが必要か？」、「データベースシステムを拡張する必要があるか？」などの検討が必要となります。

モニタリングは、現在のシステム使用率を理解するのに役立ちます。データ、ポイントが数週間または数か月に分散している場合は、システム拡張のニーズを予測するのに役立ちます。

以降では、FEPExporterが提供するOpenShiftの標準Podアライブモニタリング、リソースモニタリング、およびデータベース統計を使用したモニタリングおよびアラート操作について説明します。

5.7.1 オペレーターとオペランドのモニタリング

Prometheusの標準の死活監視およびリソース監視によってオペレーターとオペランドをモニタリングできます。

メトリクス名	Details
Alive monitoring	Podステータスを監視できます。
Resouce monitoring	以下のリソース状況を監視できます。 ・ CPU使用率 ・ CPUクォータ ・ メモリ使用量 ・ メモリクォータ ・ 現在のネットワーク使用状況 ・ 受信帯域幅 ・ 送信帯域幅 ・ 受信パケット率 ・ 送信パケット率 ・ ドロップされた受信パケット率 ・ ドロップされた送信パケット率

これらの監視項目に基づいてアラートルールを設定することにより、オペレーターとオペランドをモニタリングできます。設定方法については、“リファレンス”の付録を参照してください。

オペレーターの死活監視でエラーを検出した場合は、Podを再作成することで対処できます。

リソース監視でエラーが検出された場合は、オペレーターまたはオペランドにさらにリソースを割り当てることを検討してください。

また、OperatorHubまたはRedHat Operator Catlogページをチェックして、現在使用しているバージョン、更新可能なバージョン、およびセキュリティの脆弱性を確認してください。

5.7.2 FEPサーバのモニタリング

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。

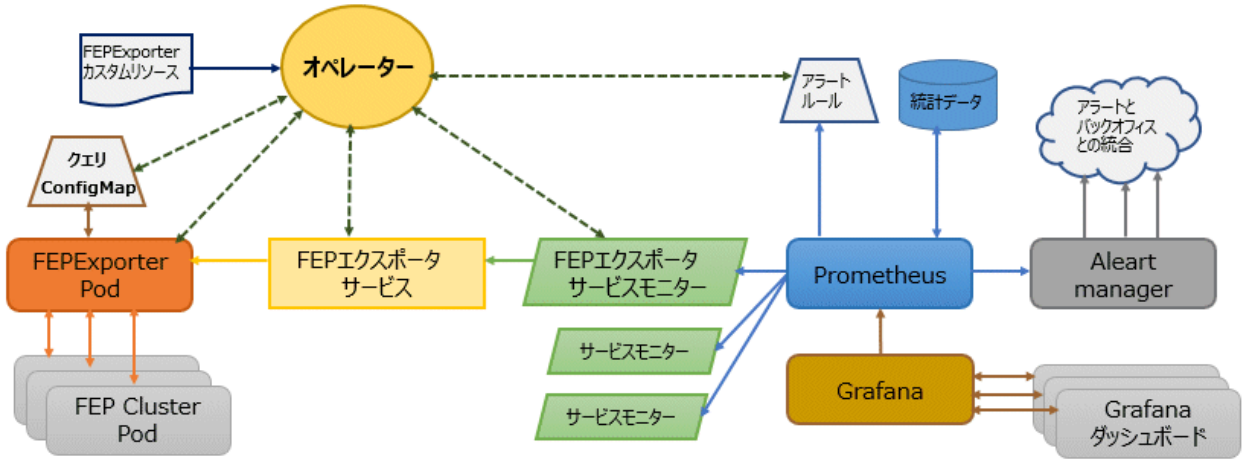
Prometheusルールが定義されている場合は、ルールを定期的に評価して、条件が満たされた場合にAlertmanagerにアラートを送信します。

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。

5.7.2.1 アーキテクチャー

FEPサーバのモニタリングの流れを以下に示します。



- FEPExporterカスタムリソースはオペレーターによって管理されます。
- FEPExporterカスタムリソースが作成されると、オペレーターは次のkubernetesオブジェクトを作成します。
  - 各ノードのデータベースクラスタからメトリクスを収集するためのデフォルトクエリとカスタムクエリを含むConfigMap
  - すべてのFEPClusterノードが接続してメトリクスを要求するためのJDBCURLを含むシークレット。この文字列には、JDBC接続を確立するための認証の詳細も含まれています。
  - デフォルトのアラートルールに対応するPrometheusルール
  - FEPエクスポータサービスを検出するためのPrometheusのServiceMonitor
  - FEPExporterイメージを使用してすべてのFEPClusterノードからメトリクスをスクレイピングするFEPエクスポータコンテナ

注意

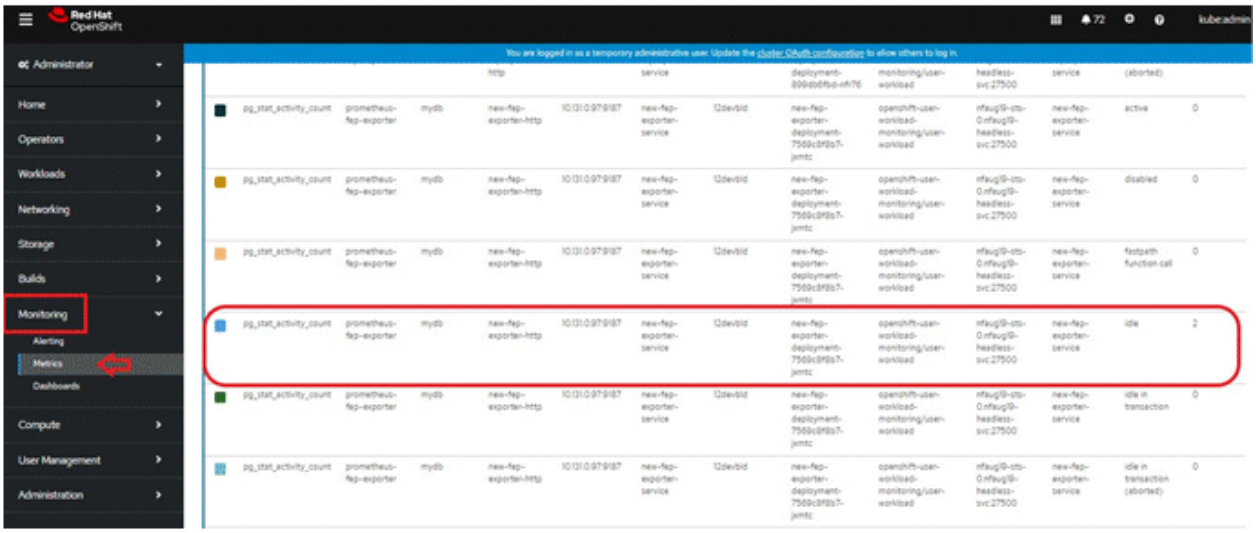
- メール/メッセージ/チケットの発行を送信するためのバックオフィスへのAlertmanagerの統合は、環境に基づいてユーザーが行います。
- Grafanaのインストールと統合はユーザーが行います。OperatorHubが提供するGrafanaオペレーターを使用します。
- Grafanaダッシュボードは、ユーザーの要件とデザインに基づいてユーザーが作成します。

5.7.2.2 デフォルトのサーバメトリクス

デフォルトでは、FEPExporterはサーバのいくつかの有用なメトリクスをスクレイプします。

FEPExporterが実行されると、ユーザーはOpenshift-> Monitoring-> Metricsサブメニューで収集されたメトリクスを確認できます。

以下の例を参照してください。



FEPExporterによって定義されたデフォルトのサーバメトリクスには2つの種類があります。

種類	説明
デフォルトで必須	FEPExporterによって収集されます。FEPExporterによってデフォルトで必須に設定されており、ユーザーが無効にすることはできません。
デフォルトで有効	稼働状態とパフォーマンスに焦点を絞ったメトリクス。ユーザーが無効にすることができます。

デフォルトで必須のメトリクス

これらのメトリクスは、データベースの基本的な統計ビューまたはFEPExporter自身のメトリクスから取得されます。

このカテゴリの各種メトリクスは以下のとおりです。

メトリクス名	説明
pg_stat_bgwriter_*	統計収集で表示するマップ
pg_stat_database_*	統計収集で表示するマップ
pg_stat_database_conflicts_*	統計収集で表示するマップ
pg_stat_archiver_*	統計収集で表示するマップ
pg_stat_activity_*	統計収集で表示するマップ
pg_stat_replication_*	統計収集で表示するマップ
pg_replication_slots_*	pg_replication_slotsシステムカタログへのマッピング
pg_settings_*	pg_settingsシステムカタログへのマッピング
pg_locks_*	pg_locksシステムカタログへのマッピング
pg_exporter_*	エクスポータのメトリクスを公開 ・ last_scrape_duration_seconds (PostgresSQLからのメトリクスの最後のスクレイプの期間) ・ scrapes_total (メトリクスのためにPostgresSQLがスクレイプされた合計回数)

メトリクス名	説明
	• last_scrape_error (PostgreSQLからのメトリクスの最後のスクレイプでエラーが発生したかどうか。エラーの場合は1、成功の場合は0)
pg_*	エクスポータメトリクスを公開します • pg_up（サービスへの接続が成功した場合は1に設定され、それ以外の場合は0に設定されます） • pg_static（postgresサーバのバージョン情報を含むラベルshort_version / バージョンをフェッチするために使用できます）

デフォルトで有効なメトリクス

データベースシステムの状態を評価するために追加されるいくつかの便利なクエリがあります。

メトリクス名	説明
pg_capacity_connection_*	接続に関するメトリクス（例：1時間実行されているtxns）
pg_capacity_schema_*	スキーマのディスク領域のメトリクス
pg_capacity_tblspace_*	表領域のディスク領域のメトリクス
pg_capacity_tblvacuum_*	数日間バキュームされていないテーブルのメトリクス
pg_capacity_longtx_*	5分以上実行されているトランザクションの数 情報を確認し、SQLの見直しとリソースの拡張を検討してください
pg_performance_locking_detail_*	ブロック状態のプロセスの詳細
pg_performance_locking_*	ブロックされた状態のプロセスの数
pg_replication_*	レプリケーションがマスターより数秒遅れる場合、参照レプリカ内の最新のデータをチェックする機能を提供します この問題を解決するには、ネットワークリソースの増加や負荷の軽減などの対策を検討する必要があります
pg_postmaster_*	ポストマスターが開始した時刻
pg_stat_user_tables_*	pg_stat_user_tablesからの重要な統計情報
pg_statio_user_tables_*	pg_statio_user_tablesからの重要な統計情報
pg_database_*	データベースサイズ データベースの容量が不足している場合は、データベースの復元が必要です
pg_stat_statements_*	サーバによって実行されたSQLステートメントの統計
pg_capacity_tblbloat_*	テーブルのフェッチによる肥大化
pg_tde_encrypted_*	テーブルスペースの透過的データ暗号化の有無と格納されているテーブルとインデックス数
pg_password_valid_*	データベースロールのパスワードの有効期間
pg_not_set_password_valid_*	パスワードの有効期間が設定されていないデータベースロール数

注意

情報収集クエリで指定された値を変更することにより、情報が収集される間隔としきい値を調整できます。詳細については、“リファレンス”の付録にあるクエリを参照して設定してください。

5.7.2.3 デフォルトのアラート

オペレーターによって設定される基本的なアラートルールを以下に示します。

アラートルール	アラートレベル	条件の永続性	説明
ContainerHighCPUUsage	Warning	5分	FEPサーバコンテナ/PodのCPU使用率がリソース制限の80%を超えています。
ContainerHighRAMUsage	Warning	30分	FEPサーバコンテナ/ポッドのメモリ使用量がリソース制限の80%を超えています
PVCLowDiskSpace	Warning	5分	FEP PVC（ボリューム）の使用可能なディスクは10%未満です。
ContainerDisappeared	Warning	60秒	FEPサーバコンテナ/Podが60秒以内に削除されました。
PostgresqlDown	Error	-	FEPサーバがダウンしたか、アクセスできません。
PostgresqlTooManyConnections	Warning	-	FEPサーバコンテナ/Pod接続の使用量が使用可能な容量の90%を超えています。
PostgresqlRolePasswordCloseExpierd	Warning	-	パスワードの有効期限が7日未満のPostgresqlロールが存在します。
PostgresqlRolePasswordExpired	Warning	-	パスワードの有効期限が切れているPostgresqlロールが存在します。

他の監視項目にアラートルールを追加することで、任意のアラートを構成できます。

アラートは統計/指標に基づいています。プラットフォームの統計が正しくない場合、誤ったアラームが発生する可能性があります。例えば、NFSストレージを利用している場合、ストレージドライバーがPVのバイト使用量の正しいメトリクスを表示していないとき、システムはPVCLowDiskSpaceの誤ったアラームを発生させる可能性があります。

5.7.2.4 グラフィカルユーザーインターフェース

デフォルトおよびカスタムのメトリクスを使用してカスタムダッシュボードを作成できます。

Grafanaダッシュボードのスクリーンショットの例を以下に示します

5.7.3 バックアップのモニタリング

バックアップデータとバックアッププロセスのステータスに関する情報は、FEPサーバテーブルとシステムビューで確認できます。バックアップ情報は、自動バックアッププロセスが完了したとき、または保存管理機能によってバックアップデータが削除されたときに更新されます。

以下のテーブルとビューが追加されます。これらのテーブルとビューは、FEPサーバのpostgresデータベースのfep_exporterスキーマの下に作成されます。

テーブル名/ビュー名	説明
pgbackrest_info_backup	バックアップの処理状況

5.7.3.1 pgbackrest_info_backupビュー

バックアップの状態に関する情報は、バックアップごとに1行で表されます。

列	型	説明
label	text	バックアップを識別する情報
type	text	full: フルバックアップ,incr: インクリメンタルバックアップ
prior	text	最初に適用する必要があるバックアップのラベル（インクリメンタルバックアップの場合のみ）
database_size	bigint	データベースサイズ
database_size_comp	bigint	データベースサイズ（圧縮後）
backup_size	bigint	バックアップサイズ
backup_size_comp	bigint	バックアップサイズ（圧縮後）
archive_start	text	リストアに必要なWALの範囲（開始）
archive_stop	text	リストアに必要なWALの範囲（終了）
backup_start	timestamp with timezon	バックアップ開始時間
backup_stop	timestamp with timezone	バックアップ終了時間
backup_exec_time	interval	バックアップ期間

5.7.4 Pgpool2のモニタリング

pgpool2のアクティビティとレプリケーションのステータスに関する情報は、、FEPサーバテーブルとシステムビューで確認できます。

pgpool2の統計情報は、パラメータで指定されたスケジュールに従って更新されます。

以下の表とビューが追加されます。これらのテーブルとビューは、FEPサーバのpostgresデータベースのfep_exporterスキーマの下に作成されます。

テーブル名/ビュー名	説明
pgpool2_stat_load_balance	pgpool 2の負荷分散情報
pgcluster_stat_replication	レプリケーションの状態
pgpool2_stat_conn_pool	pgpool 2の接続プール状態
pgpool2_stat_sql_command	SQLコマンドの統計情報

5.7.4.1 pgpool2_stat_load_balanceビュー

マスターサービス、レプリカサービスのそれぞれ1行で表されます。

列	型	説明
node_id	integer	データベースノードID（0または1）
status	text	状態（アップまたはダウン）
lb_weight	double precision	負荷分散の状態
role	text	役割（プライマリまたはスタンバイ）

列	型	説明
last_status_change	timestamp with time zone	最終ステータス変更時刻

5.7.4.2 pgpool2_stat_conn_pool view

コネクションプールの状態を表します。各pcpool2インスタンスのコネクションプール情報が含まれます。

列	型	説明
pgpool2_node_id	integer	pgpool2ノードID（0-pgpool2インスタンスの数-1）
pool_pid	integer	表示されたPgpool-IIプロセスのPID
start_time	timestamp with timezone	このプロセスが開始されたときのタイムスタンプ
pool_id	integer	プール識別子（0からmax_pool-1の間である必要があります）
backend_id	integer	バックエンド識別子（0から、構成済みバックエンドの数から1を引いた数の間である必要があります）
role	text	役割（プライマリまたはスタンバイ）
database	text	このプロセスのプールID接続のデータベース名
username	text	このプロセスのプールID接続のユーザー名
create_time	timestamp with timezo	コネクションの作成日時
majorversion	integer	このコネクションで使用されるプロトコルバージョン番号
minorversion	integer	このコネクションで使用されるプロトコルバージョン番号
pool_counter	integer	このコネクションプール（プロセス）がクライアントによって使用された回数
pool_connected	boolean	True（1） フロントエンドが現在このバックエンドを使用している場合

5.7.4.3 pgpool2_stat_sql_commandビュー

SQLコマンドの統計を表します。

列	型	説明
node_id	integer	バックエンド識別子（0から、構成済みバックエンドの数から1を引いた数の間である必要があります）
role	text	役割（プライマリまたはスタンバイ）
select_cnt	integer	SQLコマンドの数：SELECT
insert_cnt	integer	SQLコマンドの数：INSERT
update_cnt	integer	SQLコマンドの数：UPDATE
delete_cnt	integer	SQLコマンドの数：DELETE
ddl_cnt	integer	SQLコマンドの数：DDL
other_cnt	integer	SQLコマンドの数：その他
panic_cnt	integer	失敗したコマンドの数
fatal_cnt	integer	失敗したコマンドの数
error_cnt	integer	失敗したコマンドの数

5.8 イベント通知

オペレーターは、カスタマイズされたKubernetesイベントを発生させることができます。カスタムイベントは、カスタムリソースの作成中に発生します。発生するイベントを説明します。

5.8.1 発生するイベント

- fepcluster - FEPClusterカスタムリソースの作成中
  - FEPVolumeカスタムリソースの作成が開始されたとき、およびFEPVolumeカスタムリソースの作成の開始が失敗したときにイベントが発生します。
  - FEPConfigカスタムリソースの作成が開始されたとき、およびFEPConfigカスタムリソースの作成の開始が失敗したときにイベントが発生します。
  - FEPUserカスタムリソースの作成が開始されたとき、およびFEPUserカスタムリソースの作成の開始が失敗したときにイベントが発生します。
  - FEPCertカスタムリソースの作成が開始されたとき、およびFEPCertカスタムリソースの作成の開始が失敗したときにイベントが発生します。
  - ステートフルセットの作成が成功し、ステートフルセットの作成が失敗すると、イベントが発生します。
  - PDBの作成が成功した場合、およびPDBの作成が失敗した場合にイベントが発生します。
  - FEPBackupカスタムリソースの作成が開始されたとき、およびFEPBackupカスタムリソースの作成の開始が失敗したときにイベントが発生します。
  - Veleroとの連携が有効化されたFEPClusterカスタムリソースで本番環境と災害対策環境で同じオブジェクトストレージのパスを指定していたときにイベントが発生します。

  以下のカスタムリソースのイベントは、FEPCluster作成時に発生することに注意してください。

  - fepcert - FEPCertカスタムリソースの作成中
    - FEPCertカスタムリソースの作成が成功した場合、FEPCertカスタムリソースがFEPClusterへの注釈付けに失敗した場合、およびFEPCertカスタムリソースの作成が失敗した場合にイベントが発生します。
  - fepconfig -FEPConfigカスタムリソースの作成中
    - FEPConfigカスタムリソースの作成が成功した場合、FEPConfigカスタムリソースがFEPClusterへの注釈付けに失敗した場合、およびFEPConfigカスタムリソースの作成が失敗した場合にイベントが発生します。
  - fepvolume - FEPVolumeカスタムリソースの作成中
    - FEPVolumeカスタムリソースの作成が成功した場合、FEPVolumeカスタムリソースがFEPClusterへの注釈付けに失敗した場合、およびFEPVolumeカスタムリソースの作成が失敗した場合にイベントが発生します。
  - fepbackup -FEPBackupカスタムリソースの作成中
    - FEPBackup cronjob1の作成が成功した場合、およびFEPBackup cronjob1の作成が失敗した場合にイベントが発生します。
    - ·FEPBackup cronjob2の作成が成功した場合、およびFEPBackup cronjob2の作成が失敗した場合にイベントが発生します。
    - ·FEPBackup cronjob3の作成が成功した場合、およびFEPBackup cronjob3の作成が失敗した場合にイベントが発生します。
    - ·FEPBackup cronjob4の作成が成功した場合、およびFEPBackup cronjob4の作成が失敗した場合にイベントが発生します。
    - ·FEPBackup cronjob5の作成が成功した場合、およびFEPBackup cronjob5の作成が失敗した場合にイベントが発生します。
- feppgpool2-FEPPgPool2カスタムリソースの作成中
  - FEPPgPool2カスタムリソースの作成が成功した場合、およびFEPPgPool2カスタムリソースの作成が失敗した場合に、イベントが発生します。
  - FEPPgPool2Certカスタムリソースの作成が開始されたとき、およびFEPPgPool2Certカスタムリソースの作成の開始が失敗したときにイベントが発生します。

以下のカスタムリソースのイベントは、FEPPgPool2作成時に発生することに注意してください。

ー feppgpool2cert-FEPPgPool2Certカスタムリソースの作成中

- FEPPgPool2Certカスタムリソースの作成が成功した場合、FEPPgPool2CertカスタムリソースがFEPPgPool2への注釈付けに失敗した場合、およびFEPPgPool2Certカスタムリソースの作成が失敗した場合にイベントが発生します。

・ feprestore -FEPRestoreカスタムリソースの作成中n

ー FEPRestoreカスタムリソースの作成が成功した場合、およびFEPRestoreカスタムリソースの作成が失敗した場合に、イベントが発生します。

5.8.2 カスタムリソースの更新時に発生するイベント

カスタムリソースに“sysExtraEvent”パラメータを指定している場合、FEPCluster、FEPLogging、FEPExporterの変更を検知したとき、または、変更の反映に成功/失敗したときにイベントが発生します。

発生するイベントは、“リファレンス”の“オペレータ操作のイベント通知”を参照してください。

5.8.3 カスタムイベントの表示

カスタムイベントは、CLIおよびOpenshiftコンソールで確認できます。

CLIの場合

以下のコマンドを実行します。

kubectl get events

または

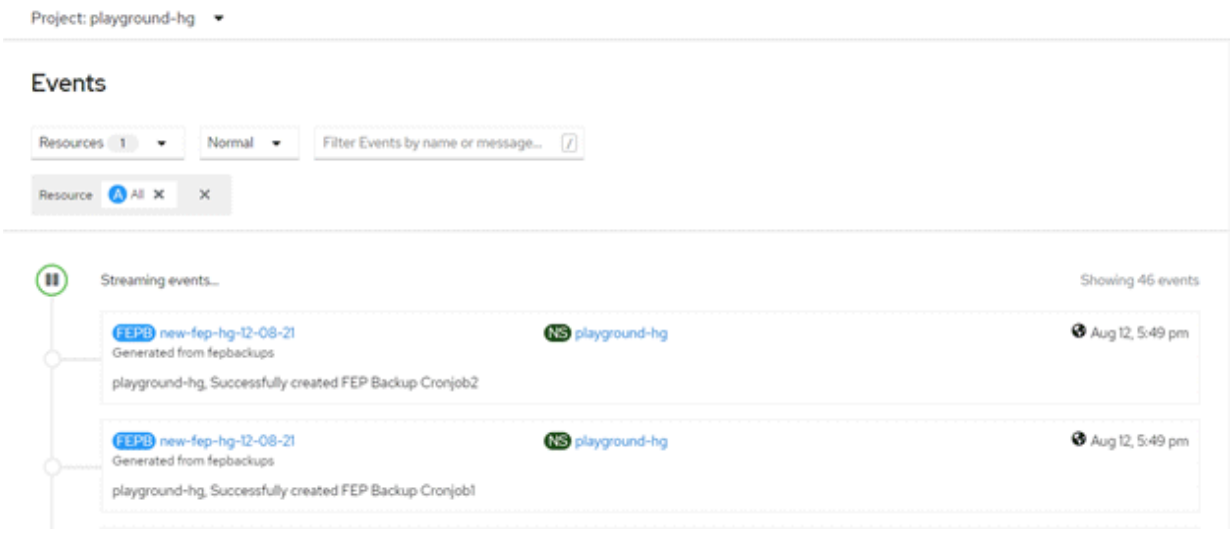
oc get events

以下は、上記のコマンドが実行されたときに出力されるイベント表示例の抜粋です。

```
14m         Normal    InitiatedChildCRCreate          fepcluster/new-fep-hg-12-08-21                  playground-hg, Started FEP Volume CR creation
13m         Normal    InitiatedChildCRCreate          fepcluster/new-fep-hg-12-08-21                  playground-hg, Started FEP User CR creation
13m         Normal    InitiatedChildCRCreate          fepcluster/new-fep-hg-12-08-21                  playground-hg, Started FEP Cert CR creation
13m         Normal    InitiatedChildCRCreate          fepcluster/new-fep-hg-12-08-21                  playground-hg, Started FEP Backup CR creation
13m         Normal    SuccessfulFepVolumeCreate       fepvolume/new-fep-hg-12-08-21                   playground-hg, Successfully created FEP Volume
13m         Normal    SuccessfulFepUserCreate         fepuser/new-fep-hg-12-08-21                     playground-hg, Successfully created FEP User
13m         Normal    SuccessfulFepCertCreate         fepcert/new-fep-hg-12-08-21                     playground-hg, Successfully created FEP Cert
13m         Normal    SuccessfulFepConfigCreate       fepconfig/new-fep-hg-12-08-21                   playground-hg, Successfully created FEP Config
13m         Normal    SuccessfulFepBackupCronjob1Create   fepbackup/new-fep-hg-12-08-21               playground-hg, Successfully created FEP Backup Cronjob1
13m         Normal    SuccessfulFepBackupCronjob2Create   fepbackup/new-fep-hg-12-08-21               playground-hg, Successfully created FEP Backup Cronjob2
13m         Normal    SuccessfulFepVolumeCreate       fepvolume/new-fep-hg-12-08-21                   playground-hg, Successfully created FEP Volume
```

Openshiftコンソールの場合

以下に示すように、特定のプロジェクト/名前空間について、カスタムイベントをイベントの下のKubernetesイベントと一緒に表示できます。



5.9 レプリカの拡張

5.9.1 自動スケールアウト

自動スケールアウトは、DBコンテナの平均CPU使用率またはコネクション数がしきい値を超えると発生します。

マスターコンテナを除くレプリカコンテナの最大数は15です。

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。必要に応じて手動スケールインを実施してください。

自動スケールアウトを実行する場合は、FEPClusterカスタムリソースで「spec.fepChildCrVal.autoscale.scaleout」を指定します。指定する値については、“リファレンス”の“FEPClusterパラメータ”を参照してください。

```
$ oc edit fepcluster <FEPClusterCR name>
```

5.9.2 手動スケールイン/スケールアウト

FEPClusterを手動でスケールインまたはスケールアウトするには、FEPClusterカスタムリソースの「spec.fep.instances」を編集します。

値は1から16の間でなければなりません。(1つのマスターを持つインスタンスの数)

```
$ oc edit fepcluster <FEPClusterCR name>
```

注意

- syncModeが「on」の場合は、レプリカインスタンスを2から1にスケールインしないでください。更新SQLが実行できなくなります。
- スケールイン実施時には削除されるレプリカPodへのデータベースコネクションは強制的に切断されます。

5.10 オブジェクトストレージへのバックアップ

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方法について説明します。

5.10.1 リソースの事前作成

5.10.1.1 CAファイル(ルート証明書)の格納

オブジェクトストレージの接続でデフォルト以外のルート証明書を利用する場合、ConfigMap に登録します。

```
$ oc create configmap storage-cacert --from-file=ca.crt=storage-ca.pem -n my-namespace
```

5.10.1.2 GCSリポジトリキーの格納

pgBackRestのパラメータ(repo-gcs-key)を使用する場合、GCSリポジトリキーをSecretに登録します。

```
$ oc create secret generic storage-key-secret --from-file=key.json=storage-key.json -n my-namespace
```

5.10.2 FEPClusterカスタムリソースの定義

FEPClusterカスタムリソースのspec.fepChildCrVal.backup配下にバックアップの設定を記載します。

pgbackrestParamsにバックアップデータ格納先のオブジェクトストレージを指定します。pgbackrestParamsに設定可能な値は“2.3.5 オペレーターからのバックアップのスケジュール設定”を参照してください。

caNameに“5.10.1.1 CAファイル(ルート証明書)の格納”で作成したConfigMap名を指定します。

FEPClusterカスタムリソースの例: オブジェクトストレージのみをバックアップレポジトリに利用する場合

```
apiVersion: fep.fujitsu.io/v2
kind: FEPCluster
metadata:
  …
spec:
  fepChildeCrVal:
    backup:
      pgbackrestParams: |
        repo1-type=s3
        repo1-path=/backup/cluster1
        repo1-s3-bucket= sample-bucket
        repo1-s3-endpoint=s3.ap-northeast-1.amazonaws.com
        repo1-s3-region=ap-northeast-1
        repo1-storage-ca-file=/pgbackrest/storage-certs/ca.crt
      pgbackrestKeyParams: |
        repo1-s3-key=SAMPLEKEY
        repo1-s3-key-secret=SAMPLESECRET
      caName:
       - storage-cacert
…
```

spec.fepChildeCrVal.storage.backupVolに指定した永続ボリュームとオブジェクトストレージをバックアップレポジトリに併用する場合は、オブジェクトストレージの設定を“repo2”以降に指定します。

“repo1”の定義がない場合、自動的にバックアップボリュームの格納先に永続ボリュームが指定されます。

FEPClusterカスタムリソースの例: オブジェクトストレージとPVを利用する場合

```
  …
spec:
  fepChildeCrVal:
    backup:
      pgbackrestParams: |
        repo2-type=s3
        repo2-path=/backup/cluster1
        repo2-s3-bucket= sample-bucket
        repo2-s3-endpoint=s3.ap-northeast-1.amazonaws.com
        repo2-s3-region=ap-northeast-1
        repo2-storage-ca-file=/pgbackrest/storage-certs/ca.crt
      pgbackrestKeyParams: |
        repo2-s3-key=SAMPLEKEY
        repo2-s3-key-secret=SAMPLESECRET
      caName:
       - storage-cacert
…
```

オブジェクトストレージGCSをバックアップリポジトリに利用する場合は下記のように指定します。

“5.10.1.2 GCSリポジトリキーの格納”で作成したSecretをrepoKeySecretNameに指定します。

また、gcs-key-typeにはserviceを指定してください。

FEPClusterカスタムリソースの例: GCSをバックアップリポジトリに利用する場合

```
apiVersion: fep.fujitsu.io/v1
kind: FEPCluster
metadata:
  …
spec:
  fepChildeCrVal:
    backup:
      pgbackrestParams: |
        repo1-type=gcs
        repo1-path=/backup-ct/test2
```

```
        repo1-gcs-bucket=dbaas-gcs
        repo1-gcs-endpoint=localhost
        repo1-storage-ca-file=/pgbackrest/storage-certs/ca.crt
        repo1-gcs-key=/pgbackrest/storage-keys/key.json
        repo1-gcs-key-type=service
      caName:
       - storage-cacert
      repoKeySecretName:
       - storage-key-secret
...
```

5.11 ディザスタリカバリ

利用可能なディザスタリカバリの方式として、バックアップ/リストア方式、継続的リカバリ方式、ストリーミングレプリケーション方式があります。

5.11.1 バックアップ/リストア方式でのディザスタリカバリ

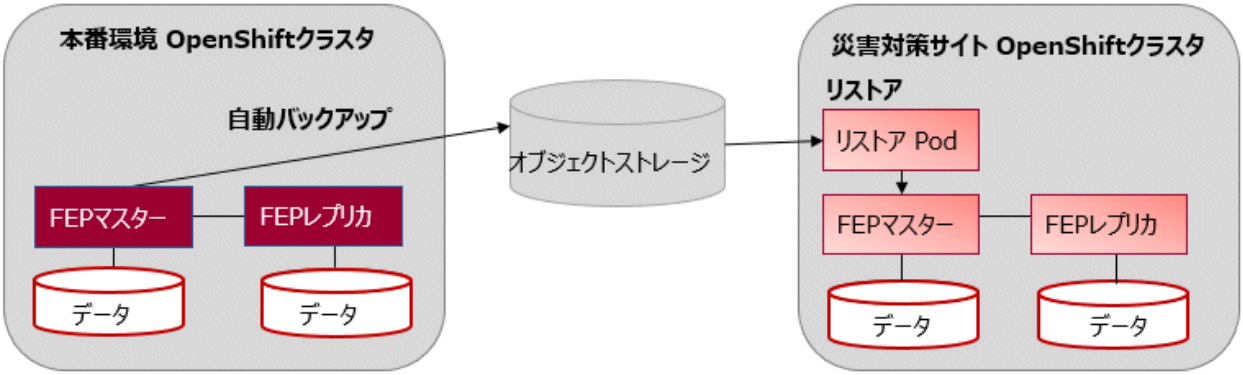
5.11.1.1 ディザスタリカバリの前提条件

バックアップ/リストア方式を用いてディザスタリカバリを実行するためのバックアップ機能の前提条件となるPodの配置、バックアップリポジトリについて構成図を以下に示します。

バックアップを取得するFEPClusterではspec.fepChildCrVal.backup.pgbackrestParamsでオブジェクトストレージをバックアップデータの格納先に指定します。

想定する災害の範囲に対して安全と思われる地域にあるオブジェクトストレージを指定します。

ディザスタリカバリ実行時にFEPClusterカスタムリソースの定義は引き継がれません。災害発生時に備えて本番環境のFEPClusterカスタムリソースの定義を保管することを推奨します。



5.11.1.2 ディザスタリカバリの実施

オブジェクトストレージに格納されたバックアップデータを利用して、リストア元と異なるOCP環境にリストアする手順を説明します。

5.11.1.2.1 リソースの事前作成

CAファイル(ルート証明書)の格納

オブジェクトストレージの接続でデフォルト以外のルート証明書を利用する場合、ConfigMap に登録します。

```
$ oc create configmap storage-cacert --from-file=ca.crt=storage-ca.pem -n my-namespace
```

GCSレポジトリキーの格納

pgBackRestのパラメータ(repo-gcs-key)を使用する場合、GCSリポジトリキーをSecretに登録します。

```
$ oc create secret generic storage-key-secret --from-file=key.json=storage-key.json -n my-namespace
```

5.11.1.2.2 FEPClusterカスタムリソースの定義

FEPClusterの設定に加えて、下記のRestoreの設定を記載します。

FEPClusterカスタムリソースの定義例

```
apiVersion: fep.fujitsu.io/v1
kind: FEPCluster
metadata:
  …
spec:
  fepChildCrVal:
    restore:
      pgbackrestParams: |
        repo1-type=s3
        repo1-path=/backup/cluster1
        repo1-s3-bucket=sample-bucket
        repo1-s3-endpoint=s3.ap-northeast-1.amazonaws.com
        repo1-s3-region=ap-northeast-1
        repo1-storage-ca-file=/pgbackrest/storage-certs/ca.crt
      pgbackrestKeyParams: |
        repo1-s3-key=SAMPLEKEY
        repo1-s3-key-secret=SAMPLESECRET
      caName:
       - storage-cacert

…
```

オブジェクトストレージGCSをバックアップリポジトリに利用する場合は下記のように指定します。

“GCSレポジトリキーの格納”で作成したSecretをrepoKeySecretNameに指定します。

また、gcs-key-typeにはserviceを指定してください。

```
apiVersion: fep.fujitsu.io/v1
kind: FEPCluster
metadata:
  …
spec:
  fepChildeCrVal:
    backup:
      pgbackrestParams: |
        repo1-type=gcs
        repo1-path=/backup-ct/test2
        repo1-gcs-bucket=dbaas-gcs
        repo1-gcs-endpoint=localhost
        repo1-storage-ca-file=/pgbackrest/storage-certs/ca.crt
        repo1-gcs-key=/pgbackrest/storage-key/key.json
        repo1-gcs-key-type=service
      caName:
       - storage-cacert
      repoKeySecretName:
       - storage-key-secret
…
```

設定値

フィールド	省略値	詳細
spec.fepChildCrVal.restore		オブジェクトストレージに格納されているバックアップデータを指定してリストアする場合に定義します。

フィールド	省略値	詳細
spec.fepChildCrVal.restore.pgbackrestParams		オプション “|” 固定とし、その下の行よりpgbackrest.confで設定するパラメータを記載します。 バックアップデータが格納されているオブジェクトストレージを指定します。 デフォルト以外のルート証明書の利用が必要な場合は、下記を指定します。 repo1-storage-ca-path=/pgbackrest/storage-certs/<ファイル名> CAファイルはConfigMapに登録し、spec.fepChildCrVal.restore.caNameにConfigMap名を記載します。
spec.fepChildCrVal.restore.pgbackrestKeyParams		オプション “|” 固定とし、その下の行よりpgbackrest.confで設定するパラメータを記載します。このパラメータで記載された値は*****でマスクされます。パスワードなどマスクしたいパラメータを指定します。
spec.fepChildCrVal.restore.caName		オプション システムのデフォルト以外のCAファイルを使用する場合に設定します。 作成したConfigMapの名前をリスト形式で指定します。 指定されたConfigMapは/pgbackrest/storage-certsにマウントされます。
spec.fepChildCrVal.restore.mcSpec.limits	cpu: 200m memory: 300Mi	オプション リストアを実行するコンテナに割り当てるCPUとメモリ
spec.fepChildCrVal.restore.mcSpec.requests	cpu: 100m memory: 200Mi	オプション リストアを実行するコンテナに割り当てるCPUとメモリ
spec.fepChildCrVal.restore.restoretype	latest	オプション リストアのタイプ（latest or PITR）
spec.fepChildCrVal.restore.restoredate		オプション spec.fepChildCrVal.restore.restoretypeが"PITR"の場合に、復元する日付を指定します。
spec.fepChildCrVal.restore.restoretime		オプション spec.fepChildCrVal.restore.restoretypeが"PITR"の場合に、復元する時間を指定します。
spec.fepChildCrVal.restore.image		オプション リストアを実行するコンテナのイメージ デフォルトでは省略されています。この場合、オペレーターコンテナ環境からimageのURLが取得されます。
spec.fepChildCrVal.restore.imagePullPolicy	IfNotPresent	オプション

5.11.2 継続的リカバリ方式でのディザスタリカバリ

5.11.2.1 ディザスタリカバリの前提条件

継続的リカバリ方式でディザスタリカバリを実施する場合は、“4.13.3.1 継続的リカバリ方式の定義”に基づいてデータベースクラスタが構成されていることが条件となります。この際、定期的にオブジェクトストレージにデータベースをバックアップする必要があります。

5.11.2.2 ディザスタリカバリの実施

災害発生時には、災害対策環境のFEPClusterを昇格させる必要があります。

ホットスタンバイ構成で配備されている災害対策環境のFEPClusterを昇格させる手順に関して説明します。災害発生時に本番環境が使用できなくなった場合、FEPAction(promote_standby)を実行することで災害対策環境を本番環境へと昇格させることができます。

以下にFEPActionの定義例を示します。

```
…
apiVersion: fep.fujitsu.io/v1
kind: FEPAction
metadata:
  name: new-fep-promote-standby-action
  namespace: my-namespace
spec:
  fepAction:
    type: promote_standby
  sysExtraEvent: true
  sysExtraLogging: true
  targetClusterName: my-fep
...
```

5.11.3 Veleroを利用したディザスタリカバリ

Veleroを利用したディザスタリカバリについて説明します。

5.11.3.1 ディザスタリカバリの前提条件

本番環境と災害対策環境で以下を準備してください。本番環境においては本機能を有効化する前までに、また災害対策環境においては本機能を利用してリストアする前までに以下を準備する必要があります。

- VeleroCLIのインストール
- Veleroインストール
- FEPOperatorのインストール
- StorageClassやnamespace、CRDなどシステム構築に必要なリソース

FEPClusterで静的なPVを利用する場合は、災害対策環境へリストアする前にPVを用意してください。本番環境と異なるストレージクラスや証明書などを使用する場合は、本番環境と同名のストレージクラスや証明書を災害対策環境に用意してください。ただし、データベースのバックアップデータの格納先であるオブジェクトストレージの証明書やシークレットが本番環境と災害対策環境で異なる場合は、異なる名前を設定してください。詳細は、“リファレンス”の“FEPClusterパラメータ”を参照してください。

本機能は、x86のみに対応しています。

5.11.3.2 ディザスタリカバリの実施

本機能を有効化して、本番環境から災害対策環境へディザスタリカバリを実行する手順を説明します。また、災害対策環境から本番環境へ再びディザスタリカバリを実行する手順を説明します。

FEPClusterとFEPPool2のLOGボリュームは、復元できません。本機能を使用する前に、必ず本番環境と災害対策環境の両方で事前にテストおよび検証を実施してください。

5.11.3.2.1 FEPClusterカスタムリソースの設定

本機能を使用するためには、本番環境のFEPClusterカスタムリソースの通常の設定に加え、以下を設定してKubernetes上に配備してください。詳細は、“リファレンス”の“FEPClusterパラメータ”を参照してください。

- fep.velero.enable
- fep.velero.labels

- fep.velero.backup
- fep.velero.restore
- fepChildCrVal.backup

例)FEPClusterカスタムリソースの定義例

```
spec:
  fep:
    velero:
      enable: true
      labels:
        backup-dev: my-backup1
        backup-dep: my-backup2
      backup:
        pgbackrestParams: |
          [global]
           repo1-retention-full=5
           repo1-retention-full-type=count
           repo2-retention-full=5
           repo2-retention-full-type=count
           log-path=/database/log/backup
           log-level-file=debug
           repo2-path=/ velero-backup/ velero-backup-for-dr2
           repo2-s3-bucket=my-s3-backet
           repo2-s3-endpoint=s3.ap-northeast-1.amazonaws.com
           repo2-s3-region=ap-northeast-1
           repo2-type=s3
         pgbackrestKeyParams: |
           repo2-s3-key=XXXXXXXXXX
           repo2-s3-key-secret=YYYYYYYYYY
         caName: DR-objectstorage-cert
         repoKeySecretName: XXX

resotre:
         image:
           image:  “XXX-amd64”
           pullPolicy: IfNotPresent
         mcSpec:
           limit:
             cpu: 200m
             memory: 300Mi
           request:
             cpu: 100m
             memory: 200Mi
         restoreTargetRepo: 2
  fepChildCrVal:
      backup:
        pgbackrestParams: |
        [global]
         repo1-retention-full=5
         repo1-retention-full-type=count
         repo2-retention-full=5
         repo2-retention-full-type=count
         log-path=/database/log/backup
         log-level-file=debug
         repo2-path=/ velero-backup/ velero-backup-for-dr1
         repo2-s3-bucket=my-s3-backet
         repo2-s3-endpoint=s3.ap-northeast-1.amazonaws.com
         repo2-s3-region=ap-northeast-1
         repo2-type=s3
       pgbackrestKeyParams: |
         repo2-s3-key=XXXXXXXXXX
```

```
    repo2-s3-key-secret=YYYYYYYYY
  caName: objectstorage-cert
  repoKeySecretName: ZZZ

  schedule:
    num: 1
    schedule1:
      schedule: "0-59/3 * * * *" #schedule1.schedule
    type: "full" #schedule1.type
    repo: 2
```

5.11.3.2.2 Veleroでのバックアップ

本環境でFEPClusterを構築後、またはFEPClusterカスタムリソースを修正した後、以下のコマンドでFEPClusterカスタムリソースを含めたkubernetes上のリソースをオブジェクトストレージにバックアップします。

例)

```
velero backup create <バックアップの名前> --selector <バックアップ対象label>
```

本機能では、FEPClusterカスタムリソースのfep.velero.labelsで記述したラベルを指定し、ラベルが付与されているリソースのみをバックアップする、Veleroのバックアップコマンドのみサポートしています。それ以外のコマンドを実行すると、不要なリソースがバックアップ/リストアされ、構築やデータ復旧が失敗する可能性があります。ラベル指定時は、既存のラベルと混同しないように、以下のkeyは指定しないでください。

- app
- control-plane
- name
- pod-template-hash
- vendor
- app.kubernetes.io/component
- app.kubernetes.io/instance
- app.kubernetes.io/managed-by
- app.kubernetes.io/name
- app.kubernetes.io/part-of
- control-plane
- controller-revision-hash
- fepclustername
- feprole
- statefulset.kubernetes.io/pod-name
- controller-uid
- job-name
- pod-template-hash
- release

本機能により、ラベルが付与されるのは以下のリソースです。

- FEPClusterカスタムリソース
- FEPCluster構築に必要なSecret

以下のリソースをVeleroを用いてバックアップする場合、各種カスタムリソースやカスタムリソースに指定したConfigMapやSecretにspec.fep.velero.labelsで指定したラベル(spec.fep.velero.labelsが省略されている場合はデフォルトのbackup-group: fep-backup)を付与してください。

- FEPPgpool2
- FEPExporter
- FEPLogging

また、手動で作成したリソース(アプリケーションシステム、ConfigMap、Secretなど)は、spec.fep.velero.labelsで指定したラベル(spec.fep.velero.labelsが省略されている場合はデフォルトのbackup-group: fep-backu)を付与し、Veleroバックアップの対象にすることでバックアップできます。

Veleroコマンドの使用方法についての詳細は、公式ドキュメントを参照してください。

注意

FEPClusterなどのカスタムリソースを変更した場合、その都度、Veleroバックアップが必要です。データベースはオブジェクトストレージによりデータが最新の状態に復元されます。カスタムリソースが変更前の状態の場合、データベースと不整合が生じ、データの損失や性能の低下、セキュリティなどの問題を引き起こす可能性があります。

5.11.3.2.3 データベースのバックアップ

データベースのバックアップを定期的に実行してください。

5.11.3.2.4 Veleroでのリストア

災害発生などにより、本番環境で業務続行が不可能な場合、以下のコマンドでオブジェクトストレージに格納したFEPClueter カスタムリソースを含めたkubernetes上のリソースをリストアします。データベースのデータは、オブジェクトストレージに格納されたアーカイブWALの最新状態まで復元されます。

例)

```
velero restore create ＜リストアの名前＞ --from-backup ＜バックアップの名前＞
```

Veleroコマンドの使用方法についての詳細は、公式ドキュメントを参照してください。

5.11.3.2.5 災害対策環境から本番環境への復元

災害対策環境から本番環境へ再度、Veleroを用いて復元する場合、災害対策環境に配備されているFEPClusterカスタムリソースに修正が必要です。ディザスタリカバリ後のFEPClsuter カスタムリソースのfep.velero.backup, fep.velero.restore には、ディザスタリカバリ前の情報が定義されています。本番環境で使用するオブジェクトストレージの情報へ最新化してください。その後、災害対策環境でVeleroを用いたバックアップ、本番環境でVeleroを用いたリストアを実施してください。

5.11.4 ストリーミングレプリケーション方式でのディザスタリカバリ

5.11.4.1 ディザスタリカバリの前提条件

ストリーミングレプリケーション方式でディザスタリカバリを実施する場合は、“4.13.3.2 ストリーミングレプリケーション方式の定義”に基づいてデータベースクラスタが構成されていることが条件となります。配備後は、データベースのデータはストリーミングレプリケーション方式で本番環境と災害対策環境間で同期されるため、運用中の操作は不要です。

5.11.4.2 ディザスタリカバリの実施

災害発生時には災害対策環境のFEPClusterを昇格させる必要があります。災害対策環境のFEPClusterの昇格については、“5.11.2.2 ディザスタリカバリの実施”を参照してください。

5.11.5 災害対策環境でのパラメータ変更

本番環境でパラメータ(FEPClusterカスタムリソースなど)を変更した場合は、災害対策環境に手動で変更を反映させてください。ただし、データベースのパスワードは、災害対策環境に自動的に変更後の値が反映されるため、手動での反映が不要です。

5.12 鍵管理システムを利用した透過的データ暗号化の運用

5.12.1 カスタムリソースのパラメータ更新

鍵管理システムで新しく生成されたマスタ暗号化キーを利用するとき、FEPClusterカスタムリソースのfepChildCrVal.sysTde.tdek.targetKeyIdに新しいマスタ暗号化キーのIDに更新します。この値が更新されたとき、オペレーターが自動的にTDEの再有効化を実行します。

また、鍵管理システムと接続するための資格情報が更新された場合、FEPClusterカスタムリソースの対応する値を更新します。資格情報が更新されると、オペレーターが自動的にキーストアのオープンを実行します。

TDEの再有効化、または、キーストアのオープンが完了したとき、下記のイベントが通知されます。

```
# TDEが再有効化が成功したとき
$ kubectl get event
LAST SEEN   TYPE    REASON                    OBJECT                     MESSAGE
164m        Normal  SuccessfulTdeSetMasterKey  fepconfig/<FEPClusterCR名> <namespace>, Successfully set TDE masterKey

# TDEの再有効化が失敗したとき
$ kubectl get event
LAST SEEN   TYPE    REASON                    OBJECT                     MESSAGE
164m        Warning    FailedTdeSetMasterKey   fepconfig/<FEPClusterCR名> <namespace>, Error/Failure set TDE masterKey
```

処理が失敗していた場合は、FEPClusterカスタムリソースに定義したパラメータを見直し、正しい値を入力し直します。

資格情報を格納したSecretやConfigMapの内容だけを更新し、カスタムリソースの修正がない場合は、“5.12.2 資格情報の更新”で説明している、FEPActionカスタムリソースを利用して、キーストアのオープンを実行します。

5.12.2 資格情報の更新

鍵管理システムの資格情報が更新された場合、対応するSecretやConfigMapの内容を更新します。FEPクラスタのカスタムリソースに指定する値に変更がない場合、FEPActionカスタムリソースをアプライしてFEPが利用する資格情報を更新します。

例)FEPActionカスタムリソースの定義例

```
apiVersion: fep.fujitsu.io/v1
kind: FEPAction
metadata:
  name: new-fep-action
spec:
  sysExtraLogging: false
  targetClusterName: nf-131851
  fepAction:
    type: open_tde_masterkey
```

5.12.3 テーブルスペースの暗号化

暗号化するテーブルスペースを作成する場合は、ランタイム・パラメータで暗号化アルゴリズムを構成します。たとえば、暗号化アルゴリズムとして256 ビットの鍵長のAESを使用して、secure_tablespaceという名前のテーブルスペースを作成するには、次のように定義します。

```
-- Specify the encryption algorithm for the tablespace to be created below
SET tablespace_encryption_algorithm = 'AES256';
CREATE TABLESPACE secure_tablespace LOCATION /database/tablespaces/tbspace1;
```

```
-- Specify that the tablespace to be created below is not to be encrypted
SET tablespace_encryption_algorithm = 'none';
```

または

```
CREATE TABLESPACE tbs_tst_new LOCATION '/database/tablespaces/tbspace1' WITH (tablespace_encryption_algorithm =
'AES256' );
```

暗号化されたテーブルスペースのチェック

以下のSQLを実行することで、どのテーブルスペースが暗号化されているかが確認できます。

```
SELECT spcname, spcencalgo FROM pg_tablespace ts, pgx_tablespaces tsx WHERE ts.oid = tsx.spctablespace;
```

5.12.4 バックアップ・リストア

FEPクラスタが破損または失われた場合に備えて、以下のタイミングでバックアップを作成する必要があります。

- クラスタが最初に作成されたとき
- マスタ暗号化キーが変更されたとき

FEPRestoreカスタムリソースを使用してバックアップから復元されたクラスタを作成すると、復元されたクラスタは、ソースクラスタ(バックアップの作成元)でのバックアップ取得時点でのマスタ暗号化キーによって復元されます。

ソースのFEPClusterのバックアップ時点より新しいマスタ―暗号化キーがsysTde.tdek.targetKeyIdに指定されているとき、リストア先のFEPClusterカスタムリソースに値が引き継がれ、データ復元後にオペレーターが自動的に新しいマスター暗号化キーでTDEを再有効化します。

またリストア実行前に、鍵管理システムへの認証情報を最新化してください。認証情報が最新化されていないと鍵管理サービスに接続することができず、データを復元できません。

sysTde.tdek.kmsDefinition配下の鍵管理システムへ接続するための情報をFEPCluster構築後に誤って更新した場合、データの復元時に鍵管理システムを参照できません。リストア処理実行前に、FEPClusterカスタムリソースに正しい値が記載されていることを確認してください。

5.12.5 鍵管理システムの定義の変更

鍵管理システムを変更するときなど、鍵管理システムへの接続情報に追加や変更がある場合はFEPClusterカスタムリソースのspec.fepChildCrVal.sysTde.tdek.kmsDefinition配下のパラメータを修正します。

以下に該当する変更を行うと、レプリカサーバを新しいパラメータで再起動します。レプリカサーバが複数台ある場合は、1台ずつ順番に再起動します。すべてのレプリカサーバが再起動されると、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再起動します。これにより、業務停止を最小限にしてすべてのサーバの鍵管理システムの定義を変更することができます。

- 新しい鍵管理システムの定義を追加する
- 既存の鍵管理システムの定義を削除する
- 鍵管理システムの定義の順序を変更する
- 資格情報として指定する、ConfigMapやSecretリソースを追加、削除、またはリソース名を変更する

再起動を伴う変更を行う場合、変更する前にデータベースの自動スケールアウト機能を一時的に無効にしてください。自動スケールアウト機能は、FEPClusterカスタムリソースのspec.fepChildCrVal.autoscale.scaleout.policyパラメータで無効にできます。

現在、キーストアとして利用している鍵管理システムの資格情報として指定している、ConfigMap/Secretリソースの名前は変更できません。

5.13 機密管理支援機能

5.13.1 機密管理支援機能の有効化

FEPCluster構築時に、下記のデータベースに機密管理支援機能の拡張“pgx_confidential_management_support”のインストールとセットアップがされています。

- template1
- postgres
- spec.fepChildCrVal.sysUsers.pgdbに指定したデータベース

また、機密管理者ロール（spec.fepChildCeVal.sysUsers.pgSecurityUser）を作成した場合、このロールには機密管理支援機能の実行に必要な下記の機能が割り当てられています。

- CREATE ROLE
- 拡張に含まれるすべてのテーブルへのSELECT権限、INSERT権限、UPDATE権限、およびDELETE権限

そのため、FEPCluster構築直後から拡張“pgx_confidential_management_support”がインストールされているデータベースまたはtemplate1から作成されたデータベースで、機密管理支援機能によるデータベースオブジェクトの管理をすることができます。

機密管理支援機能の操作の詳細は、“Fujitsu Enterprise Postgres セキュリティ運用ガイド”の“機密管理支援機能”を参照してください。

拡張に含まれるテーブルは、“Fujitsu Enterprise Postgres セキュリティ運用ガイド”の“機密管理支援機能が利用するテーブル”を参照してください。

また、機密管理支援機能によるデータベースオブジェクトを管理するデータベースロールをスキーマごとに用意するなど、機密管理者ロール以外のデータベースロールで機密管理支援機能の操作が必要な場合、機密管理支援機能から対象のデータベースロールに下記の権限を割り当てます。

- CREATE ROLE
- 拡張に含まれるすべてのテーブルへのSELECT権限、INSERT権限、UPDATE権限、およびDELETE権限

他のユーザーが作ったデータベースオブジェクトを機密管理支援機能で管理する場合は、機密管理支援機能を操作するデータベースロールにデータベースオブジェクトの所有権を与える必要があります。

例）機密管理者ユーザー“security_user”にテーブル“security_table”の所有権を与える場合

```
ALTER TABLE security_table OWNER TO security_user;
```

データベースオブジェクトの所有者はPostgreSQLのメタコマンド“¥d”などを利用して確認します。

5.13.2 機密管理支援機能の監視

FEPログ機能を利用してpgAuditの監査ログをElasticseachに転送することができます。

転送した監査ログを分析し、意図しないユーザーによるデータベースオブジェクトに対する権限変更がされていないことを監視してください。

FEPログ機能については、“4.9 FEPログ機能”を参照してください。

第6章 保守運用

本章では、コンテナをデプロイした後の保守運用について説明します。

6.1 マイナーバージョンアップグレード

FEPのマイナーバージョンのアップグレードは、FEPClusterカスタムリソースのイメージを新しいものに置き換えることによって行われます。手順については、“概説書”の“マイナーバージョンのアップグレード”を参照してください。

新しいFEPサーバコンテナがリリースされたかどうかを確認するための更新情報は、RedHatカタログにあります。

アップグレードはローリングアップデートされるため、停止を局所化できますが、接続されたアプリケーションは接続エラーを引き起こすため、業務運用中の実行は避けることをお勧めします。

注意

アップグレードプロセスにより、マスターレプリカと同期レプリカの両方をアップグレードしている間、クラスタが停止します。クラスタに同期レプリカがない場合、停止はマスターをアップグレードする時間の長さに制限されます（または、実際には、別のレプリカを取得するために必要なフェイルオーバ時間は、Patroniによって昇格されます）。

6.2 マスターインスタンスのスイッチオーバ

マスターインスタンスのパフォーマンス障害または計画されたノードメンテナンスが発生した場合は、マスターインスタンスをレプリカインスタンスに切り替えることができます。

FEPAction CRを更新するには、FEPAction CRのアクションタイプに「switchover」を指定します。

Kubernetesコマンド：kubectl apply -f <new_spec>

「switchover」アクションタイプでは、ユーザーがスイッチオーバを実行するターゲットクラスタの名前を指定する必要があります。FEPActionが内部的に切り替え元のPodを特定して新しいマスターPodを昇格させるため、スイッチオーバではargsセクションは必要ありません。

```
spec:
fepAction:
 type: switchover
 targetClusterName: new-fep
```

パラメータについての詳細は、“リファレンス”の“FEPActionカスタムリソースパラメータ”を参照してください。

6.3 PITRによるリストアの実行

PITRによるリストアは、アプリケーションの障害が原因でデータベースを特定の場所にリストアしたり、本番用に複製データベースを準備したりする場合に実行します。

リストアプロセスでは、次のようにリストア用のカスタムリソース（FEPRestoreカスタムリソース）を作成することにより、データを復元できます。

oc create -f [Custom Resource Files]

Example)

```
$oc create -f config/samples/postgres_v1_restore.yaml
```

リストアには、既存のFEPClusterにデータを復元する方法と、新しいFEPClusterにデータを復元する方法の2つがあります。

既存のFEPClusterに復元する場合、FEPCluster名、IPアドレス、さまざまな設定などの情報は同じままです。

新しいFEPClusterに復元する場合、FEPCluster名はCRで指定した名前であり、新しいIPアドレスも指定されます。設定値が指定されていない場合、新しいクラスタはリストア元のクラスタから設定を継承しますが、CRで指定することにより、設定を変更して新しいクラスタを作成できます。

6.3.1 設定項目

カスタムリソースファイルに設定する項目については、“リファレンス”の“FEPRestoreカスタムリソースパラメータ”を参照してください。

6.3.2 リストア後について

新しいクラスタへの接続の切り替え

リストアにより、新しいFEPClusterが作成されます。必要に応じて、Pgpool-IIをセットアップし、アプリケーションの接続先を新しいクラスタまたは新しいPgpool-IIに変更する必要があります。

クラスタのバックアップデータ

リストアの完了後に新しいクラスタのバックアップが開始されるため、リストア前の時間にPITRをリストアすることはできません。

6.4 メジャーバージョンアップグレード

オペレーターとFEPコンテナのメジャーバージョンのアップグレード手順を説明します。

FEPのメジャーバージョンアップグレードは、1つ前のメジャーバージョンのFEPと同じNamespaceに新しいメジャーバージョンのFEPを構築します。このときFEPClusterCRに“spec.fepChildCrVal.upgrade”フィールドを定義することで、オペレーターがアップグレード実行コンテナを作成します。アップグレード実行コンテナは“spec.fepChildCrVal.upgrade.sourceCluster”に指定された1つ前のバージョンのFEP Clusterをデータソースの FEPClusterとし新規作成したFEPClusterにデータを移行します。

6.4.1 データソースのFEPClusterの事前作業

メジャーバージョンアップグレード実行前に、実行中の業務アプリケーションを停止します。

次に、データソースのFEPClusterカスタムリソースの“spec.fepChildCrVal.customPgHba”を編集して、アップグレード実行コンテナの接続を許可します。

接続を許可するアドレスは下記のように指定します。

```
<fep>-upgrade-pod.<fep>- upgrade-headless-svc.<namespace>.svc.cluster.local
```

<fep>は新規作成するFEPClusterカスタムリソース名を指定します。

また、認証方法は、trust/md5/cert のいずれかが利用可能です。

データソースのFEPClusterカスタムリソースの編集例

```
apiVersion: fep.fujitsu.io/v2
kind: FEPCluster
metadata:
  name: source-fep
  namespace: my-namespace
spec:
  fepChildCrVal:
    customPgHba: |
      host  all  all  destination-fep-upgrade-pod. destination-fep-upgrade-headless-svc. my-namespace.svc.cluster.local
trust
...
```

6.4.2 オペレーターのアップグレード

オペレーターのアップグレード手順について説明します。

注意

オペレーターアップグレード後は、前のバージョンで定義したカスタムリソースの設定変更をコンテナに反映できなくなります。

6.4.2.1 前のバージョンのオペレーターのアンインストール

前のバージョンのオペレーターをアンインストールします。

"Operators" > "Installed Operators" > "Fujitsu Enterprise Postgres <Old version> Operator" > Actions から "Uninstall Operator" を選択します。

6.4.2.2 新しいバージョンのオペレーターのインストール

“第3章 オペレーターのインストール”を参照して新バージョンのオペレーターをインストールします。

6.4.3 FEPのメジャーバージョンアップグレード

6.4.3.1 新規FEPClusterCRの作成

“リファレンス”を参照し、新しいメジャーバージョンのFEPClusterカスタムリソースを定義します。このとき、“6.4.1 データソースのFEPClusterの事前作業”と同様に、アップグレード実行コンテナの接続を許可してください。

さらに、下記のFEPClusterカスタムリソースの定義例のように“spec.fepChildCrVal.upgrade”フィールドを定義することで、FEPのメジャーバージョンアップグレードが実行されます。

アップグレード実行コンテナではデータソースのFEPClusterから取得したダンプファイルを格納するためのPVを利用します。

Kubernetes環境でPVの自動プロビジョニング機能を有効にしていない場合は、FEPClusterカスタムリソースを作成する前に、新規作成のFEPCluster用のPVに加えて、アップグレード用のPVを作成します。

また、“6.4.1 データソースのFEPClusterの事前作業”と同様に、“spec.fepChildCrVal.customPgHba”を編集して、アップグレード実行コンテナの接続を許可します。

アップグレードを実行するFEPClusterカスタムリソースの定義例

```
apiVersion: fep.fujitsu.io/v2
kind: FEPCluster
metadata:
  name: destination-fep
  namespace: my-namespace
spec:
  fep:
    ...
  fepChildCrVal:
    upgrade
      sourceCluster: source-fep-cluster
      storage:
        size: 8Gi
    customPgHba: |
      host  all  all  destination-fep-upgrade-pod.destination-fep-upgrade-headless-svc.my-namespace.svc.cluster.local
trust
    ...
```

FEPClusterカスタムリソースのフィールド“spec.fepChildCrVal.upgrade”

フィールド	デフォルト値	説明
spec.fepChildCrVal.upgrade		オプション このフィールドが定義されるとメジャーバージョンアップグレードを実行します。

フィールド	デフォルト値	説明
		ただし、spec.fepChildCrVal.restoreが定義されているとFEPClusterの構築が停止します。
spec.fepChildCrVal.upgrade.sourceCluster		データ移行元のFEPClusterCR名を指定します。 spec.fepChildCrVal.upgradeを定義する場合、必ず指定します。
spec.fepChildCrVal.upgrade.mcSpec.limits	cpu: 200m memory: 300Mi	オプション アップグレード実行コンテナに割り当てるリソースの上限値を指定します。
spec.fepChildCrVal.upgrade.mcSpec.requests	cpu: 100m memory: 200Mi	オプション アップグレード実行コンテナに割り当てるリソースの下限値を指定します。
spec.fepChildCrVal.upgrade.image		オプション 省略時はオペレーターコンテナ環境からimageのURLが取得されます。
spec.fepChildCrVal.upgrade.imagePullPolicy	IfNotPresent	オプション コンテナイメージのpull policyを指定します。 ・ Always ・ IfNotPresent ・ Never
spec.fepChildCrVal.upgrade.source.pgAdminTls.certificateName		オプション データソースのFEPClusterがアップグレード実行コンテナの認証方法を"cert"にしていた場合、データソースのFEPClusterのspec.fepChildCrVal.sysUsers.pgAdminTls.certificateNameを定義しているシークレットの証明書を利用します。 上記のパラメータが定義されていない場合、このパラメータでデータソースのPostgresユーザー"postgres"の証明書を含むKubernetes TLSシークレットを指します。 シークレットの作成方法は“4.7.1 証明書を手動で管理する方法”を参照してください。
spec.fepChildCrVal.upgrade.destination.pgAdminTls.certificateName		オプション 新規作成のFEPClusterがアップグレード実行コンテナの認証方法を"cert"にしていた場合、新規作成のFEPClusterのspec.fepChildCrVal.sysUsers.pgAdminTls.certificateNameを定義しているシークレットの証明書を利用します。 上記のパラメータが定義されていない場合、このパラメータで新規作成のPostgresユーザー"postgres"の証明書を含むKubernetes TLSシークレットを指します。 シークレットの作成方法は“4.7.1 証明書を手動で管理する方法”を参照してください。

フィールド	デフォルト値	説明
spec.fepChildCrVal.upgrade.storage		オプション ダンプファイルを格納するためのストレージを定義します。
spec.fepChildCrVal.upgrade.storage.storageClass		オプション 省略された場合、運用している環境のデフォルトのストレージクラスを利用します。
spec.fepChildCrVal.upgrade.storage.size	2Gi	オプション ダンプファイルを格納するストレージのサイズを指定します。
spec.fepChildCrVal.upgrade.storage.accessModes	ReadWriteOnce	オプション ダンプファイルを格納するストレージのアクセスモード アクセスモードの配列として指定します。 例: [ReadWriteMany] 省略すると、[ReadWriteOnce]として扱われます。

注意

データベースに接続し、以下のSQLを実行して、事前にデータベースのサイズを確認してください。

```
$ SELECT pg_size_pretty(sum(pg_database_size(datname))) AS dbsize FROM pg_database;
```

アップグレード実行コンテナで利用する、pg_dumpallコマンドはデータベースのデータをSQLコマンドとして出力するため、実際に作成されるファイルは次のとおりです。

例えば、整数型の2147483647は、データベースデータの場合は4バイトです。

ただし、SQLコマンドはそれらを文字列として出力するため、これは10バイトです。したがって、ダンプファイルを格納するストレージ(PV)は十分なディスク容量を用意してください。

6.4.3.2 FEPメジャーアップグレードの完了確認

新しいFEPClusterにデータを移行し、FEPのメジャーバージョンアップグレードが成功すると下記のeventが出力されます。

```
$ kubectl get event
LAST SEEN  TYPE    REASON                OBJECT                              MESSAGE
164m       Normal  SuccessfulFepUpgrade  fepupgrade/<新規FEPClusterCR名>  <namespace>, Successfully FEP Upgrade
```

また、FEPClusterCRのYAMLに下記のアノテーションが追記されます。

```
apiVersion: fep.fujitsu.io/v2
kind: FEPCluster
metadata:
  annotations:
    FEPUpgradeDone: true
...
  name: destination-fep-cluster
  namespace: my-namespace
spec:
...
```

注意

FEPのメジャーアップグレードに失敗すると下記のようなeventが出力されます。

```
$ kubectl get event
LAST SEEN   TYPE     REASON            OBJECT                              MESSAGE
164m        Warning  FailedFepUpgrade  fepupgrade/<新規FEPClusterCR名>  <namespace>, Error/Failure in FEP Upgrade
```

OBJECTカラムに記載されているKubernetesリソースの情報を取得し、出力されているメッセージを確認のうえで、新規のFEPClusterカスタムリソースを再作成してください。

```
$ kubectl describe fepupgrade/<新規FEPClusterカスタムリソース名>
```

6.4.4 各カスタムリソースの更新

FEPのメジャーアップグレード完了後に、データソースのFEPClusterの運用で利用していた各カスタムリソースに関する手順を説明します。

この処理が完了後、中断していた業務アプリケーションを再開します。

6.4.4.1 データソースのFEPClusterCRの削除

データソースのFEPClusterを削除します。

OpenshiftのGUIコンソールの場合

"Operators" > "Installed Operators" > "Fujitsu Enterprise Postgres <New version> Operator" > "FEPCluster" > "FEPCluster name to delete" > Actionsから"Delete FEPCluster" を選択します。

6.4.4.2 FEPPgpool2

バージョンアップしたFEPとクライアントのバージョンを合わせるため、FEPPgpool2を再作成してください。

6.4.4.3 スタンドアロンモードで構築したFEPExporter

FEPExporterカスタムリソースの“spec.fepExporter.fepClusterList”を編集して、新しいバージョンのFEPClusterカスタムリソースを指定します。

パラメータの詳細は“リファレンス”の“FEPExporterカスタムリソース”を参照してください。

6.5 オペレーターコンテナの割り当てリソース

本製品で提供しているオペレーターのコンテナには下記のリソースがデフォルトで割り当てられています。

```
resources:
limits:
  cpu: 2
  memory: 1536Mi
requests:
  cpu: 500m
  memory: 768Mi
```

オペレーターの管理対象であるFEPClusterカスタムリソースが1つの場合は、デフォルトで割り当てられているリソースで運用が可能です。ただし、FEPClusterカスタムリソースを複数配備して運用する場合は、オペレーターコンテナの割当リソースを変更してください。

注意

リソースの変更をした場合、オペレーターのバージョンアップグレード後はリソースの値がデフォルト値に戻ります。そのため、オペレーターのアップグレード後は再度リソースの変更を実施してください。

6.5.1 割り当てリソースの変更方法

オペレーターコンテナに割り当てられたリソースの変更方法を説明します。

オペレーターコンテナに割り当てられたリソースを更新する際には、オペレーターコンテナが再作成されます。このとき、すでに構築済みのFEPClusterなどのコンテナの運用が止まることはありません。

割り当てリソースの変更方法はオペレーターのインストール方法により異なります。

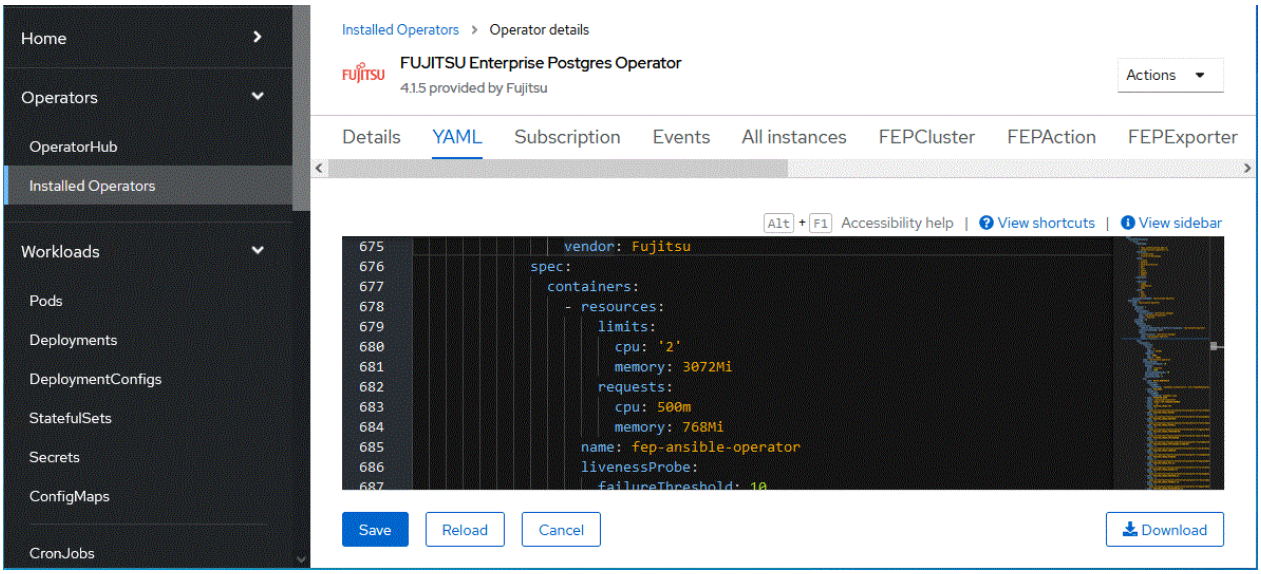
6.5.1.1 OperatorHubを利用してインストールした場合

オペレーターをOperatorHubからインストールして利用している場合、オペレーターコンテナに割り当てられたリソースを変更するためには、ClusterServiceVersion（CSV）を編集します。

CSVの「spec.install.spec.deployments[0].spec. template.spec.containers[0].resources」を編集することで、オペレーターコンテナが再作成され、指定したリソースが適用されます。

OCPのGUIコンソールからCSVを編集する場合

Operators配下のメニュー項目で[Installed Operators]をクリックし、インストールしたオペレーターを選択します。[YAML]タブで割り当てリソースの指定箇所を編集し、[Save]をクリックします。



OCクライアントを利用してCUIコンソールからCSVを編集する場合

“oc get”コマンドでインストールしているオペレーターのCSV名を確認します。

```
$ oc get csv
NAME                                          DISPLAY                                VERSION   REPLACES   PHASE
fujitsu-enterprise-postgres-operator.v4.1.5   Fujitsu Enterprise Postgres Operator   4.1.5               Succeeded
```

“oc edit”コマンドでCSVを編集します。

```
$ oc edit csv fujitsu-enterprise-postgres-operator.v4.1.5
```

6.5.1.2 Helm ChartまたはRancherUIを利用してインストールした場合

オペレーターをHelm Chart、またはRancherUIを利用してインストールしている場合、オペレーターコンテナに割り当てられたリソースを変更するためには、オペレーターコンテナのDeploymentを編集します。

Deploymentの「spec.template.spec.containers[0].resources」を編集することで、オペレーターコンテナが再作成され、指定したリソースが適用されます。

“kubectl edit”コマンドでDeployment“fep-ansible-operator”を編集します。

```
$ kubectl get deployment fep-ansible-operator
NAME                   READY   UP-TO-DATE   AVAILABLE   AGE
fep-ansible-operator   1/1     1            1           2m10s

$ kubectl edit deployment fep-ansible-operator
```

6.6 SUPERUSER権限の利用

6.6.1 CREATE EXTENSION

PostgreSQLの外部拡張をインストールするCREATE EXTENTIONコマンドを実行する際、SUPERUSERしかインストールできない拡張があります。そのような拡張をインストールするために、FEPActionカスタムリソースを利用します。

FEPActionカスタムリソースのspec.fepAction.typeに“create_extention”を指定することで、指定したFEPClusterのコンテナに対して、CREATE EXTENTIONを実行させることができます。

利用方法は、“リファレンス”を参照してください。

6.6.2 SUPERUSERのパスワード変更

SUPERUSER“postgres”のパスワードを更新する場合、FEPActionカスタムリソースのfepActionTypeにupdate_admin_passwordを指定します。

パスワードをランダムな値で再作成し、更新します。

利用方法は、“リファレンス”を参照してください。

6.6.3 SUPERUSERの利用

データベース運用でSUPERUSER権限が必要な場合、下記の手順でSUPERUSER“postgres”のパスワードを取得することができます。

1. FEPClusterカスタムリソース名と同じ名前のSecretからbase64でエンコードされたパスワードを取得します。

   例）FEPClusterカスタムリソース名がnew-fepの場合

   ```
   $ kubectl get -o yaml secret new-fep | grep PG_ADMIN_PASSWORD
     PG_ADMIN_PASSWORD: YWRtaW4tcGFzc3dvcmQ=
   ```

2. 取得したパスワードをデコードします。

   ```
   $ echo YWRtaW4tcGFzc3dvcmQ= | base64 -d
   admin-password
   ```

注意

第三者にSUPERUSERが利用されないように、Secretには、データベースの管理者以外が参照不可となるKubernetesのRole権限を設定してください。

第7章 異常時の対処

本章ではFEPの運用中における、データベースやアプリケーションに異常が発生した場合の対処方法について説明します。

異常の種類に応じて、バックアップ資材からのリカバリ、容量の確保、オペレーターのログの確認、FEPのログの確認を実施してください。

7.1 データ異常時の対処

以下のいずれの場合に障害が発生する直前のバックアップからデータベースクラスタを復旧してください。

- データ格納ディスク、またはバックアップデータ格納ディスクでハードウェア障害が発生した場合
- ディスク内のデータが論理的に破壊され、データベースが正常に動作しない場合
- ユーザーの誤操作によりデータ破壊が発生した場合

リストア手順については、“6.3 PITRによるリストアの実行”を参照してください。

7.2 データ格納先やトランザクションログ格納先の容量不足時の対処

データ格納先の容量が不足した場合は、まずディスク上に不要なファイルがないかを確認し、不要なファイルを削除して業務を継続できるようにしてください。

不要なファイルを削除しても問題を解消できない場合は、容量の大きなディスクへのデータの移行が必要になります。

データの移行には、バックアップリストアを利用してください。

FEPRestoreカスタムリソースを利用して新規FEPClusterを構築し、データを移行することができます。利用方法は、“リファレンス”の“FEPRestoreカスタムリソースパラメータ”を参照してください。

7.3 バックアップデータ格納先の容量不足時の対処

バックアップデータ格納先の容量が不足した場合は、まずディスク上に不要なファイルがないかを確認し、不要なファイルを削除するようにしてください。または、バックアップの保存世代を減らしてください。

FEPActionカスタムリソースのspec.fepAction.typeにbackup_expireを指定することで、バックアップの保存世代数を削減できます。詳細は、“リファレンス”の“FEPActionカスタムリソースパラメータ”を参照してください。

7.4 インスタンス起動停止に失敗したとき、アクセス異常の対処

インスタンスの起動や停止に失敗した場合は、オペレーターのログ、FEPのログを参照して原因を特定してください。

オペレーターのログ、FEPのログの確認は“7.5 障害調査情報の採取”を参照してください。

7.5 障害調査情報の採取

環境構築や運用中に発生したトラブルの原因が判明しない場合、初期調査のための情報を採取します。

初期調査のための情報の採取方法について説明します。

- 製品のログ
- オペレーターのログ

製品のログ

FEPのログ

コンテナ内の以下からログを採取してください。

ログの場所は、FEPClusterカスタムリソースのspec.startupValues.customPgParam パラメタ内の log_directoryで指定します。デフォルトは/database/log です。

Pgpool-IIのログ

コンテナ内の以下からログを採取してください。

ログの場所は、/var/log/pgpool/pool.log.です。

オペレーターのログ

オペレーターのログは、以下の方法で確認できます。

確認例

```
$oc get po
NAME                                      READY   STATUS    RESTARTS   AGE
fep-ansible-operator-7dc5fd9bf7-4 smzk    1/1     Running   0          20m
```

ログの確認方法

```
$oc logs pod fep-ansible-operator-7dc5fd9bf7-4 smzk -c manager
```

ログがコンソールに出力されますので、リダイレクトなどでファイル出力し確認してください。

付録A　定量制限および制限事項

A.1　定量制限

定量制限については、“Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”を参照してください。

A.2　制限事項

注意

コンテナにログインして構成ファイルを直接編集する場合、コンテナを再起動すると変更が取り消される場合があります。

設定を変更する場合は、“5.2 構成の変更”の説明に従ってカスタムリソースファイルを変更し、再適用してください。変更するパラメータによっては、コンテナが再デプロイされる場合があります。パラメータの詳細については、“5.2 構成の変更”を参照してください。

使用できないFEPの機能

FEPサーバコンテナは他のコンポーネント(UBIやPatroniなど)に基づいているため、VMベースのサーバインスタンスに対しては以下の制限があります。

No	制限	制限の理由	説明
1	CryptoExpressカードはサポートされていません	IBM LinuxOneは、現段階ではOpenshiftコンテナープラットフォームでCryptoExpressカードをサポートしていません	FEP TDEz拡張機能は、Linux One Openshift環境では使用できません。 ただし、ユーザーはLinuxOne Openshift環境とAzure(x86)Openshift環境の両方でTDEを引き続き使用できます。

変更できないパラメータ

一部のパラメータは変更できません。“2.3.5.2 設定できないパラメータを参照してください。

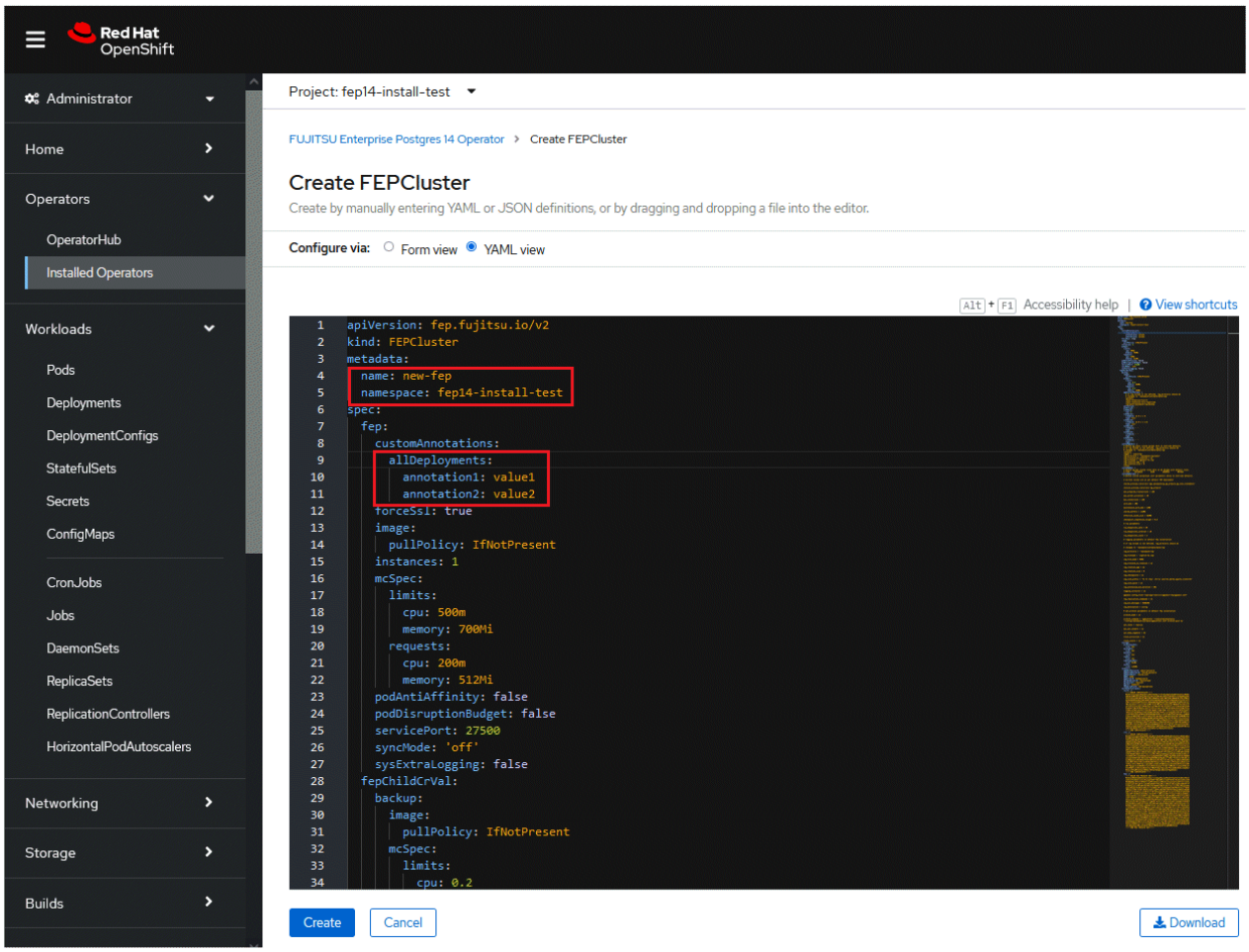
使用時に設定が必要なFEP機能

“2.3.7 デフォルトで有効なFEP機能”を参照してください。

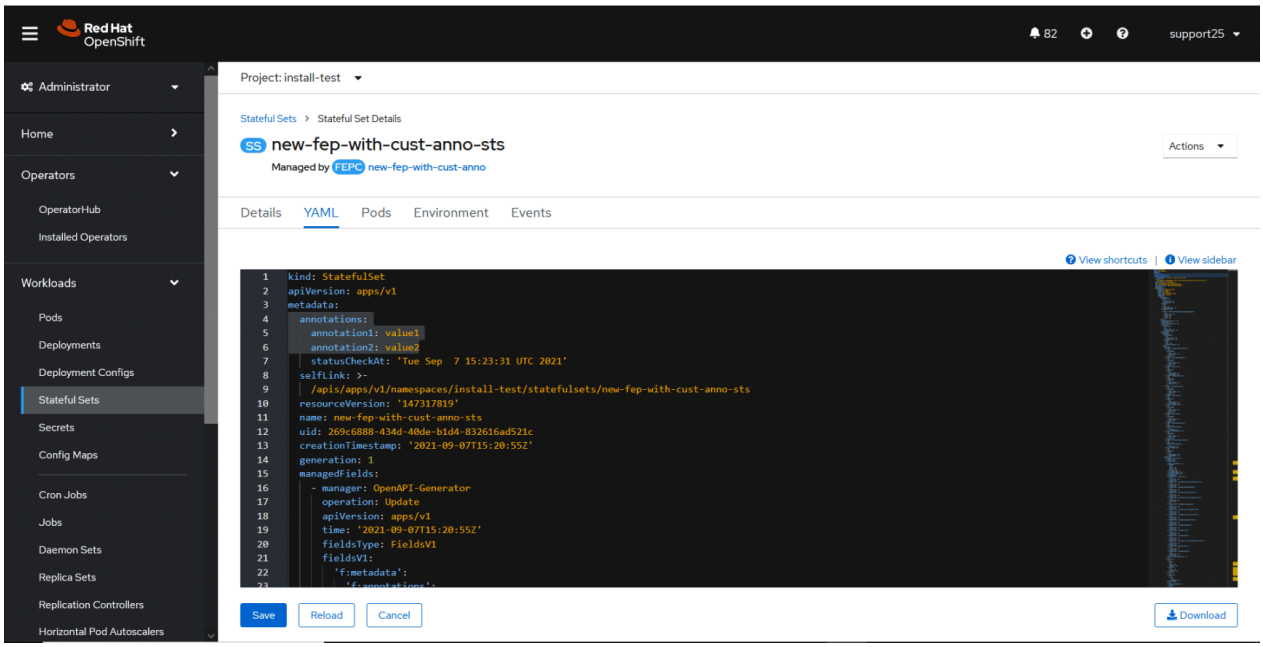
付録B　オペレーターを使用したFEPCluster Podへのカスタムアノテーションの追加

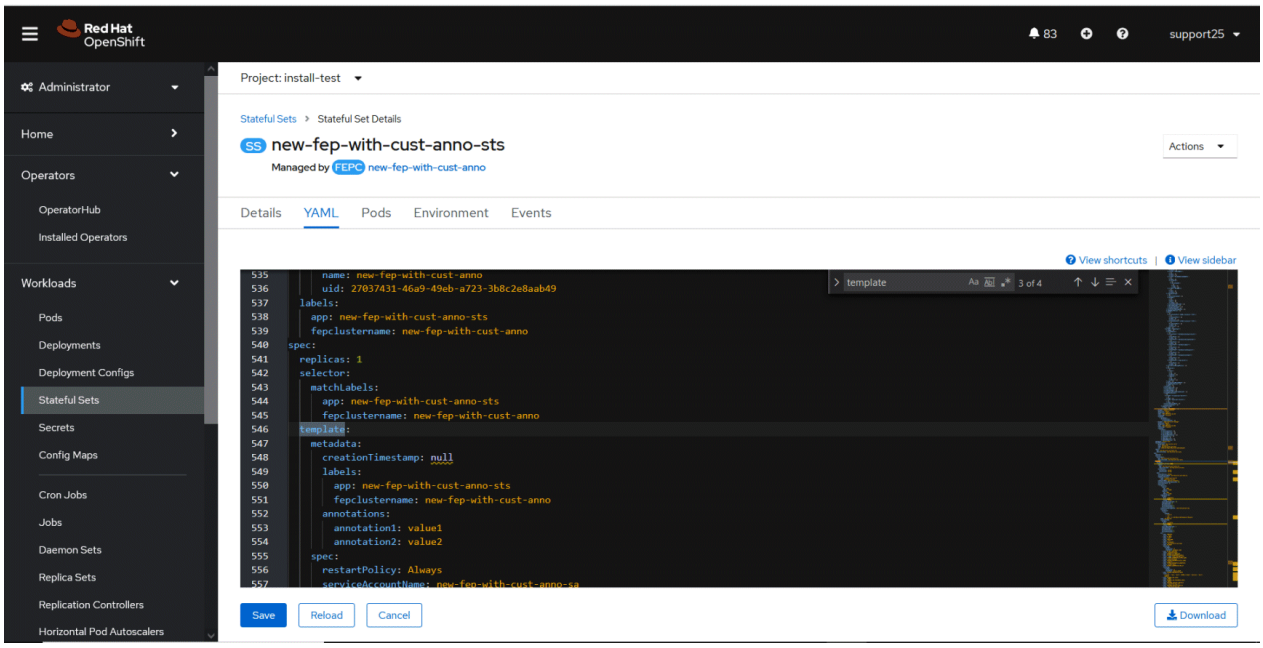
カスタムアノテーションをFEPClusterのPodに追加する手順について説明します。

1. [Create FEPCluster]セクションのYAML Viewで、以下のようにカスタムアノテーションを追加し、[Create]をクリックします。



2. Statefulsetとその結果のPodの両方にアノテーションが追加されます。archivewalVolとbackupVolはReadWriteManyである必要があります。.





付録C　共有ストレージの使用

共有ストレージを使用してFEPClusterを構築する方法について説明します。

PVaccessModesでReadWriteManyを指定できるディスクを使用してください。

ここでは、静的プロビジョニングでPVとしてNFSを使用する例を示します。

C.1 StorageClassの作成

StorageClassを作成します。

OCP WebGUI画面で、メインメニューの[Storage]の[StorageClass]をクリックし、[Create Storage Class]> [YAMLの編集]をクリック後、YAMLを編集してStorageClassを作成します。

CLIを使用している場合は、yamlファイルを作成し、以下のコマンドを使用してStorageClassを作成します。

```
$ oc create -f <file_name>.yaml
```

YAML定義は、以下の例を参照してください。

例)

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: < StorageClass Name >
provisioner: kubernetes.io/no-provisioner
reclaimPolicy: Delete
volumeBindingMode: WaitForFirstConsumer
```

C.2 PersistentVolumeの作成

必要な数のPersistentVolumes(PV)を作成します。

Web GUI画面で、メインメニューの[Storage]の[PersistentVolumes]をクリックし、[Create PersistentVolume]をクリック後、YAMLを編集してPVを作成します。

CLIを使用している場合は、yamlファイルを作成し、以下のコマンドを使用してPVを作成します。

```
$ oc create -f <file_name>.yaml
```

YAML定義は、以下の例を参照してください。

StorageClass名は、“C.1 StorageClassの作成”で作成したStorageClassを指定します。

PVごとに異なるNFSディレクトリを割り当てます。

さらに、accessModesはReadWriteManyです。

例)

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: < PV name >
spec:
  capacity:
    storage: < Capacity Required ex.8Gi >
  accessModes:
  - ReadWriteMany
  persistentVolumeReclaimPolicy: Retain
  mountOptions:
    - hard
  nfs:
```

```
    path: < NFS directory path (Assign a different directory for each PV) ex. /nfs/pv >
    server: < IP address of the NFS server ex. 192.168.1.10>
  storageClassName: < “C.1 StorageClassの作成” で作成したStorageClass名>
```

C.3 FEPClusterの作成

“4.1 オペレーターを使用したFEPClusterのデプロイ”の手順4のYAML定義でReadWriteManyPVを使用するように指定します。

「spec.fepChildCRVal.storage」で、“C.2 PersistentVolumeの作成”で作成したPVのStorageClassとAccessModesを指定します。

「spec.fepChildCRVal.storage.<ボリュームタイプ> .size」は、割り当てられたPVのサイズ以下である必要があります。

例）archivewalVolとbackupVolによって作成されたPVを使用する

```
apiVersion: fep.fujitsu.io/v2
kind: FEPCluster
metadata:
  name: t3-fep
spec:
  ~ Suppress ~
  fepChildCrVal:
    storage:
      archivewalVol:
        size: < Capacity Required ex. 8Gi >
        storageClass: < “C.1 StorageClassの作成” で作成したStorageClass名 >
        accessModes:
        - "ReadWriteMany"
      backupVol:
        size: < Capacity Required ex. 8Gi >
        storageClass: < “C.1 StorageClassの作成” で作成したStorageClass名>
        accessModes:
        - "ReadWriteMany"
  ~ Suppress ~
```

付録D　透過的データ暗号化で利用できる鍵管理システム

透過的データ暗号化で利用できる鍵管理システムについて説明します。

D.1　KMIPサーバ

KMIPサーバの要件は、“Fujitsu Enterprise Postgres 導入ガイド(サーバ編)”の“KMIPプロトコルを使用して鍵管理システムに接続する場合”を参照してください。

D.2　AWSの鍵管理サービス

D.2.1　利用できるサービス

AWS KMSアダプタを利用することで、AWSが提供するKey Management Service（以降、AWS KMSと呼びます）上の暗号鍵を利用できます。AWS KMSが対応しているリージョンであれば、リージョンの制限はありません。

D.2.2　利用できるAWS KMSのキー

KMSキーのキー仕様(key spec)は対称暗号鍵である必要があります。非対称暗号鍵は使用できません。また、KMSキーの用途(key usage)はENCRYPT_DECRYPTである必要があります。

D.2.3　必要な権限

使用するKMSキーに対し、AWS KMSに対してアクセスするユーザーに以下の操作が許可されている必要があります。

- Encrypt
- Decrypt
- DescribeKey

D.2.4　鍵ID

TDEの鍵管理システム連携機能における鍵IDとして、以下が指定できます。

- キーARN

D.3　Azureの鍵管理サービス

D.3.1　利用できるサービス

Azure KMSアダプタを利用することで、AzureのKey Vault APIによってアクセス可能で、かつ対称鍵を使用できる鍵管理サービスを利用できます。

参照

対称鍵が利用可能な鍵管理サービスについては以下を参照してください。

https://learn.microsoft.com/en-us/azure/key-vault/keys/about-keys#key-types-and-protection-methods

Azureの鍵管理サービスについては以下を参照してください。

https://learn.microsoft.com/en-us/azure/security/fundamentals/key-management#azure-key-management-services

D.3.2 利用可能なキー

対象鍵が利用できます。

D.3.3 利用可能なアルゴリズム

暗号化/復号操作の際に以下のアルゴリズムが利用できます。

- A256GCM

D.3.4 キーの操作

使用するキーに対し、Azureの鍵管理サービスに対してアクセスするユーザーに以下の操作が許可されている必要があります。

- encrypt
- decrypt
- get

D.3.5 鍵ID

TDEの鍵管理システム連携機能における鍵IDとして、以下が指定できます。

- キーのオブジェクト識別子

D.3.6 サインイン

サービスプリンシパルを使用してAzureにサインインします。サインインのために、アプリケーションID、テナントID、および資格情報が必要です。

参照

サービスプリンシパルについては以下を参照してください。

https://learn.microsoft.com/en-us/cli/azure/create-an-azure-service-principal-azure-cli#4-sign-in-using-a-service-principal