

FUJITSU Enterprise Postgres 14 for Kubernetes

User's Guide

Linux



Preface

Purpose of this document

This document describes system configuration, design, installation, setup, and operational procedures of the FUJITSU Enterprise Postgres for Kubernetes.

Intended readers

This document is intended for people who are:

- Considering installing FUJITSU Enterprise Postgres for Kubernetes
- Using FUJITSU Enterprise Postgres for Kubernetes for the first time
- Wanting to learn about the concept of FUJITSU Enterprise Postgres for Kubernetes
- Wanting to see a functional overview of FUJITSU Enterprise Postgres for Kubernetes

Readers of this document are also assumed to have general knowledge of:

- Linux
- Kubernetes
- Containers
- Operators

Structure of this document

This document is structured as follows:

[Chapter 1 System Requirements](#)

Describes the system requirements.

[Chapter 2 Overview of Operator Design](#)

Describes an overview of the operator design.

[Chapter 3 Operator Installation](#)

Describes the installation of the FEP operator.

[Chapter 4 Deployment Container](#)

Describes container deployment.

[Chapter 5 Post-Deployment Operations](#)

Describes the operation after deploying the container.

[Chapter 6 Maintenance Operations](#)

Describes the maintenance operation after deploying the container.

[Chapter 7 Abnormality](#)

Describes the actions to take when an error occurs in the database or an application.

[Appendix A Quantitative Values and Limitations](#)

Describes the quantitative values and limitations.

[Appendix B Adding Custom Annotations to FEPCluster Pods using Operator](#)

Describes instructions for adding custom annotations to a FEPCluster pod.

[Appendix C Utilize Shared Storage](#)

Describes how to build a FEPCluster when using shared storage.

Abbreviations

The following abbreviations are used in this manual:

Full Name	Abbreviations
FUJITSU Software Enterprise Postgres for Kubernetes FUJITSU Software Enterprise Postgres	FEP or FUJITSU Enterprise Postgres
Vertical Clustered Index	VCI
Transparent Data Encryption	TDE
Point in time recovery	PITR
Custom Resource	CR
Custom Resource Definition	CRD
Persistent Volume	PV
Universal Base Image	UBI
OpenShift Container Platform	OCP
Mutual TLS	MTLS

Abbreviations of manual titles

The following abbreviations are used in this manual as manual titles:

Full Manual Title	Abbreviations
FUJITSU Software Enterprise Postgres for Kubernetes Release Notes	Release Notes
FUJITSU Software Enterprise Postgres for Kubernetes Overview	Overview
FUJITSU Software Enterprise Postgres for Kubernetes User's Guide	User's Guide
FUJITSU Software Enterprise Postgres for Kubernetes Reference	Reference

Trademarks

- Linux is a registered trademark or trademark of Mr. Linus Torvalds in the U.S. and other countries.
- Red Hat and all Red Hat-based trademarks and logos are trademarks or registered trademarks of Red Hat, Inc. in the United States and other countries.
- S/390 is a registered trademark of International Business Machines Corporation in the United States or other countries or both.

Other product and company names mentioned in this manual are the trademarks or registered trademarks of their respective owners.

Export restrictions

If this document is to be exported or provided overseas, confirm legal requirements for the Foreign Exchange and Foreign Trade Act as well as other laws and regulations, including U.S. Export Administration Regulations, and follow the required procedures.

Issue date and version

Edition 6.0: June 2023
Edition 5.0: October 2022
Edition 4.0: September 2022
Edition 3.0: June 2022

Edition 2.0: April 2022
Edition 1.0: March 2022

Copyright

Copyright 2021-2023 FUJITSU LIMITED

Contents

Chapter 1 System Requirements.....	1
1.1 Components Embedded.....	1
1.2 CPU.....	1
1.3 Supported Platform.....	1
1.4 Collaboration Tool.....	2
Chapter 2 Overview of Operator Design.....	3
2.1 Design Task.....	3
2.2 System Configuration Design.....	3
2.2.1 Server Configuration.....	3
2.2.2 User Account.....	5
2.2.3 Basic Information of the Container.....	5
2.3 Design Perspective for Each Feature.....	9
2.3.1 Deployment.....	10
2.3.2 High Availability.....	10
2.3.3 Configurable Volume per Cluster.....	11
2.3.4 Deploying Pgpool-II and Connect to FEPCluster from Operator.....	12
2.3.5 Scheduling Backup from Operator.....	13
2.3.5.1 Important Setting Items.....	13
2.3.5.2 Parameters that cannot be Set.....	14
2.3.5.3 Restricted Parameters.....	16
2.3.5.4 About Sections in the Config File.....	16
2.3.6 Perform PITR and Latest Backup Restore from Operator.....	16
2.3.7 FEP Unique Feature Enabled by Default.....	17
2.3.8 Monitoring & Alert (FEPExporter).....	17
2.3.8.1 FEPExporter Custom Resource.....	17
2.3.8.2 Change to FEPCluster CR - metrics user.....	17
2.3.8.3 FEPExporter CR auto-create for FEPCluster.....	17
2.3.9 Scaling Replicas.....	18
2.3.9.1 Change to FEPCluster CR - auto scale out.....	19
2.3.10 Disaster Recovery.....	19
2.3.11 Transparent Data Encryption Using a Key Management System.....	19
Chapter 3 Operator Installation.....	20
3.1 Using the OperatorHub.....	20
3.1.1 Pre-requisite.....	20
3.1.2 Deploying Operator.....	21
3.2 Using the Helm Chart.....	23
3.2.1 Deploying Operator.....	23
3.2.2 Upgrading Operators.....	23
3.3 Using the Rancher UI.....	23
3.3.1 Pre-requisite.....	24
3.3.2 Register Helm Chart Repository.....	25
3.3.3 Deploying Operator.....	27
3.4 Implement Collaborative Monitoring Tools.....	28
3.5 Implement Client.....	29
Chapter 4 Deployment Container.....	30
4.1 Deploying FEPCluster using Operator.....	30
4.2 Deploy a Highly Available FEPCluster.....	34
4.3 Deploying FEPExporter.....	35
4.4 FEPExporter in Standalone Mode.....	37
4.5 Configuration FEP to Perform MTLS.....	39
4.5.1 Manual Certificate Management.....	40
4.5.2 Automatic Certificate Management.....	43
4.5.3 Deploy FEPCluster with MTLS support.....	48

4.5.4 Configurable Parameters.....	55
4.6 Replication Slots.....	57
4.6.1 Setting Up Logical Replication using MTLS.....	57
4.7 FEP Logging.....	60
4.7.1 FEPLogging Configuration.....	60
4.7.1.1 FEPLogging Custom Resources - spec.....	60
4.7.1.1.1 Define fepLogging image.....	62
4.7.1.1.2 Define fepLogging mcSpec.....	63
4.7.1.1.3 Define fepLogging restartRequired.....	63
4.7.1.1.4 Define fepLogging scrapeInterval and scrapeTimeout.....	63
4.7.1.1.5 Define fepLogging elastic.....	63
4.7.1.1.6 Define authSecret for elastic.....	64
4.7.1.1.7 Define fepLogging TLS.....	64
4.7.1.1.8 Define Prometheus TLS.....	64
4.7.2 FEPCluster Configuration.....	65
4.7.2.1 FEP Custom Resources - spec.fep.remoteLogging.....	65
4.7.2.1.1 Define remoteLogging enable and fluentdName.....	65
4.7.2.1.2 Define remoteLogging tls.....	66
4.7.2.1.3 Define remoteLogging image.....	66
4.7.3 FEPLogging Operations.....	67
4.7.3.1 Log Forwarding to Elasticsearch.....	67
4.7.3.2 Log severity based Alarms/Metrics.....	67
4.7.3.3 Forwarding auditlog to Elasticsearch.....	68
4.7.4 Limitations.....	68
4.8 Configuring pgBadger.....	68
4.8.1 FEP Custom Resources - spec.fep.pgBadger.....	68
4.8.2 Define pgBdager Schedules.....	69
4.8.3 Define pgBdager Options.....	69
4.8.4 Define Endpoint for Uploading Report.....	69
4.8.5 Uploaded File on Web Server.....	72
4.9 Transparent Data Encryption Using a Key Management System.....	72
4.9.1 Certificate Registration.....	72
4.9.2 Configuring FEPCluster Custom Resources.....	73
4.9.2.1 Define spec.fepChildCrVal.customPgParams.....	73
4.9.2.2 Define spec.fepChildCrVal.sysTde.....	73
Chapter 5 Post-Deployment Operations.....	75
5.1 How to Connect to a FEP Cluster.....	75
5.2 Configuration Change.....	76
5.3 FEPCluster Resource Change.....	77
5.3.1 Changing CPU and Memory Allocation Resources.....	77
5.3.2 Resizing PVCs.....	77
5.4 FEPPGPool2 Configuration Change.....	77
5.5 Scheduling Backup from Operator.....	79
5.6 Configure MTLS Setting.....	80
5.6.1 Certification Rotation.....	80
5.7 Monitoring.....	80
5.7.1 Monitoring FEP Operator and Operands.....	81
5.7.2 Monitoring FEP Server.....	81
5.7.2.1 Architecture.....	82
5.7.2.2 Default Server Metrics Monitoring	82
5.7.2.3 Default Alerts.....	84
5.7.2.4 Graphical user interface.....	85
5.7.3 Monitoring FEP Backup.....	85
5.7.3.1 pgbackrest_info_backup view.....	85
5.7.4 Monitoring FEP PGPool2.....	86
5.7.4.1 pgpool2_stat_load_balance view.....	86

5.7.4.2 pgpool2_stat_conn_pool view.....	86
5.7.4.3 pgpool2_stat_sql_command view.....	87
5.8 Event Notification.....	87
5.8.1 Events raised	87
5.8.2 Viewing the custom events.....	88
5.9 Scaling Replicas.....	89
5.9.1 Auto Scale Out.....	89
5.9.2 Manual Scale In/Out.....	89
5.10 Backing Up to Object Storage.....	90
5.10.1 Pre-creation of Resources.....	90
5.10.1.1 Storing CA Files (Root Certificates).....	90
5.10.1.2 Storing Repository Key.....	90
5.10.2 Defining a FEPCluster Custom Resource.....	90
5.11 Disaster Recovery.....	91
5.11.1 Disaster Recovery Prerequisites.....	91
5.11.2 Performing Disaster Recovery.....	92
5.11.2.1 Pre-creation of Resources.....	92
5.11.2.1.1 Storing CA Files (Root Certificates).....	92
5.11.2.1.2 Storing Repository Key.....	92
5.11.2.2 Defining a FEPCluster Custom Resource.....	92
5.12 Operation of Transparent Data Encryption Using Key Management System.....	94
5.12.1 Updating Custom Resource Parameters.....	94
5.12.2 Update Credentials.....	95
5.12.3 Encrypting a Tablespace.....	95
5.12.4 Backup/Restore.....	95
Chapter 6 Maintenance Operations.....	97
6.1 Minor Version Upgrade.....	97
6.2 Cluster Master Switchover.....	97
6.3 Perform PITR and the Latest Backup Restore from Operator.....	97
6.3.1 Setting Item.....	98
6.3.2 After Restore.....	98
6.4 Major Version Upgrade.....	98
6.4.1 Pre-work on the Data Source FEP Cluster.....	98
6.4.2 Operator Upgrade.....	98
6.4.2.1 Uninstalling the Old Operator.....	99
6.4.2.2 Installing a New Version of the Operator.....	99
6.4.3 Major Version Upgrade of FEP.....	99
6.4.3.1 Creating a New FEPCluster CR.....	99
6.4.3.2 Verifying FEP Major Upgrade Complete.....	102
6.4.4 Updating Each Custom Resource.....	102
6.4.4.1 Removing a FEPClusterCR for a Data Source.....	102
6.4.4.2 FEPPgpool2.....	102
6.4.4.3 FEPExporter Built in Standalone Mode.....	102
6.5 Assigned Resources for Operator Containers.....	103
6.5.1 How to Change Assigned Resources.....	103
6.5.1.1 When installing using OperatorHub.....	103
6.5.1.2 When installing using Helm Chart or RancherUI.....	104
Chapter 7 Abnormality.....	105
7.1 Handling of Data Abnormalities.....	105
7.2 Handling when the Capacity of the Data Storage Destination or Transaction Log Storage Destination is Insufficient.....	105
7.3 What to do when the Capacity of the Backup Data Storage Area is Insufficient.....	105
7.4 Handling Access Abnormalities When Instance Shutdown Fails.....	105
7.5 Collection of Failure Investigation Information.....	105
Appendix A Quantitative Values and Limitations.....	107
A.1 Quantitative Values.....	107

A.2 Limitations..... 107

Appendix B Adding Custom Annotations to FEPCluster Pods using Operator..... 108

Appendix C Utilize Shared Storage..... 110

 C.1 Creating a StorageClass..... 110

 C.2 Creating a PersistentVolume..... 110

 C.3 Creating FEPCluster..... 111

Chapter 1 System Requirements

This chapter describes the system requirements.

1.1 Components Embedded

The FEP Server container embeds following components. However it is understood that these components are bound to be upgraded in the maintenance phase.

No	Component	Version	Description
1	Red Hat UBI minimal	8	Meant to provide base OS image for the container
2	FUJITSU Enterprise Postgres Server	14.0	To provide server capabilities
3	Patroni	2.1.2	To provide HA capabilities and other management to the Cluster

1.2 CPU

It should be noted that it provides supports to both the following CPU Architectures to meet the scope of work.

No	CPU architecture
1	x86
2	s390x
3	ppc64le

1.3 Supported Platform

It supports running on the following platforms.

No	Platform		Version
1	OpenShift Container Platform		4.10, 4.11, 4.12
2	Rancher Kubernetes Engine (on Linux hosts) (*1)		1.4.0+
3	Vmware Tanzu Kubernetes Grid (*1)		1.6+
4	Full Managed Kubernetes Service	<ul style="list-style-type: none">- 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

*1: Kubernetes 1.24 - 1.26

Supports storage supported by OpenShift or Kubernetes (AKS, EKS, RKE, ACK, GKE, IKS and TKG).

However, you need shared storage, like NFS, or object storage for backup and archive WAL volumes. Object storage supports Amazon Simple Storage Service, Azure Blob Storage, and Google Cloud Storage.

1.4 Collaboration Tool

Supports integration with the following tools.

No	Tool	Version	How to obtain
1	Prometheus	<ul style="list-style-type: none"> - OpenShift The version installed OpenShift - Kubernetes - Prometheus v2.36.2 and later - AlertManager v0.24.0 and later - Rancher The version provided by Rancher Monitoring Chart 	<ul style="list-style-type: none"> - OpenShift Preinstalled with OpenShift - Kubernetes prometheus-operator (v0.61.1 and later) https://github.com/prometheus-operator/prometheus-operator/prometheus-operator - Rancher Using the Rancher Monitoring Chart
2	AlertManager		
3	Grafana	<ul style="list-style-type: none"> - OpenShift and Kubernetes Grafana v7.5.17 and later - Rancher The version provided by Rancher Monitoring Chart 	<ul style="list-style-type: none"> - OpenShift Provided by OperatorHub - Kubernetes grafana-operator (v4.7.1 and later) https://github.com/grafana-operator/grafana-operator - Rancher Using the Rancher Monitoring Chart
4	Helm	3.7.2 and later	<ul style="list-style-type: none"> - Kubernetes only Helm Web Site https://helm.sh/docs/intro/install/
5	Rancher	v2.6.2 and later	Rancher Web Site https://rancher.com/
6	Prometheus Adapter	<ul style="list-style-type: none"> - OpenShift and Kubernetes Confirmed the operation with v0.9.1 and later - Rancher The version provided by Rancher Monitoring Chart 	<ul style="list-style-type: none"> - OpenShift and Kubernetes Prometheus Adapter https://github.com/kubernetes-sigs/prometheus-adapter - Rancher Using the Rancher Monitoring Chart

Chapter 2 Overview of Operator Design

This chapter describes an overview of the operator design.

2.1 Design Task

Installation/operation using an operator and necessity of design are shown below.

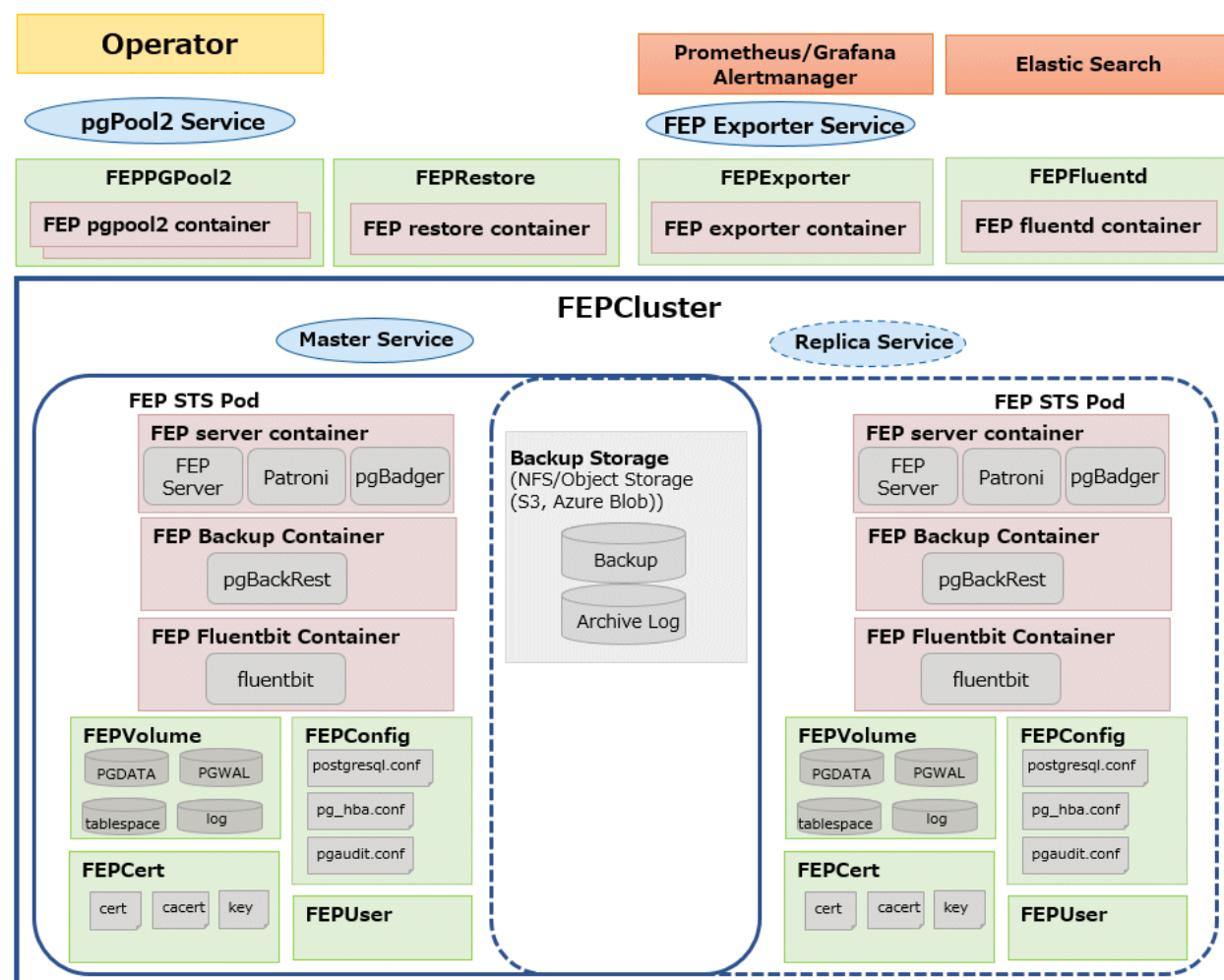
Task	Design required to operate FEP	Where to find
FEP setup	Required.	2.3.1 Deployment
High availability configuration	Optional. (When checking or changing the behavior of high availability. However, even by default, constant high availability operation is possible.)	2.3.2 High Availability
Volume settings	Optional. (When setting the volume. However, even by default, allocate a fixed volume.)	2.3.3 Configurable Volume per Cluster
Pgpool-II setup	Optional. (When using Pgpool-II.)	2.3.4 Deploying Pgpool-II and Connect to FEPCluster from Operator
Backup/restore settings	Optional. (When using a backup and restore.)	2.3.5 Scheduling Backup from Operator 2.3.6 Perform PITR and Latest Backup Restore from Operator
Monitoring & Alert(FEPExporter)	Optional. (When using Monitoring and Alert)	2.3.8 Monitoring & Alert (FEPExporter)
Scaling Replicas	Optional. (When using scaling feature)	2.3.9 Scaling Replicas

2.2 System Configuration Design

This section describes the system configuration.

2.2.1 Server Configuration

The following is an overview diagram of the server configuration:



System component

Describes various system resources.

Configuration server type	Description
FEP operator	A container that accepts user requests and is responsible for automating database construction and operational operations.
FEP server container	A container for the FEP server.
FEP backup container	A container that performs scheduled backup operations. Created on the same Pod as the FEP server container.
FEP Fluentbit container	A container that collect FEP database CSV log and forward to fluentd container for processing.
FEP pgpool2 container	A container that uses Pgpool-II to provide load balancing and connection pooling. If you do not use it, you do not need to create it.
FEP restore container	A container that performs the restore operation. Temporarily created during a restore operation.
FEP Exporter container	A container that exposes http/https endpoint for monitoring stats scraping.
FEP Fluentd container	A container that summarise FEP log severity as metrics for Prometheus to consume. Optionally, forward log entries to Elasticsearch for detailed log analysis.

Configuration server type	Description
Backup storage	Storage where backup data is stored. If you do not need to obtain a backup, you do not need to create one.
FEPCluster	Parent CR for FEP Cluster definition and configuration.
FEPBackup	Child CR for backup configuration.
FEPVolume	Child CR for volumes.
FEPConfig	Child CR for FEP configurations.
FEPCert	Child CR for system certificates.
FEPUser	Child CR for database users.
FEPAction	CR for performing actions.
FEPExporter	CR for monitoring configuration.
Master service	A service to connect to the master FEP server.
Replica service	A service to connect to the replica FEP server.
Pgpool2 service	A service for connecting to Pgpool-II.
Fepexporter service	A service to scrape metrics from all FEPCluster nodes.

2.2.2 User Account

The user accounts used by this product are as follows.

User type	User name	Description
Infrastructure administrator	Mandatory	A system administrator (superuser) who has root privileges on all the servers that make up this product.
Database administrator	Mandatory	Install, set up, start, stop, and perform operation and maintenance of this product.
Application developer	Mandatory	Develops and executes database applications.

2.2.3 Basic Information of the Container

This section describes the basic information of the container.

FEP server container

The naming convention for the FEP server container is as below.

`fujitsu-enterprise-postgres-14-server:OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH`

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	14	
<i>MajorVersion</i>	1,2, ...	To be used when major change in image, including server patch application
<i>MinorVersion</i>	0,1,2 ...	To be used when minor changes in image, e.g bug fix in container script

The first publishing will expect following names / tagging (Manifest and Child images).

- fujitsu-enterprise-postgres-14-server:ubi8-14-1.1
- fujitsu-enterprise-postgres-14-server:ubi8-14-1.1-amd64
- fujitsu-enterprise-postgres-14-server:ubi8-14-1.1-s390x
- fujitsu-enterprise-postgres-14-server:ubi8-14-1.1-ppc64le

FEP backup container

Use the same naming convention for FEP backup containers as for FEP server containers.

fujitsu-enterprise-postgres-14-backup: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	14	
<i>MajorVersion</i>	1,2, ...	To be used when major change in image, including server patch application
<i>MinorVersion</i>	0,1,2 ...	To be used when minor changes in image, e.g bug fix in container script

The first publishing will expect following names / tagging (Manifest and Child images)

- fujitsu-enterprise-postgres-14-backup:ubi8-14-1.1
- fujitsu-enterprise-postgres-14-backup:ubi8-14-1.1-amd64
- fujitsu-enterprise-postgres-14-backup:ubi8-14-1.1-s390x
- fujitsu-enterprise-postgres-14-backup:ubi8-14-1.1-ppc64le

FEP restore container

Use the same naming convention for FEP restore containers as for FEP server containers.

fujitsu-enterprise-postgres-14-restore: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	14	
<i>MajorVersion</i>	1,2, ...	To be used when major change in image, including server patch application
<i>MinorVersion</i>	0,1,2 ...	To be used when minor changes in image, e.g bug fix in container script

The first publishing will expect following names / tagging (Manifest and Child images)

- fujitsu-enterprise-postgres-14-restore:ubi8-14-1.1
- fujitsu-enterprise-postgres-14-restore:ubi8-14-1.1-amd64
- fujitsu-enterprise-postgres-14-restore:ubi8-14-1.1-s390x

- fujitsu-enterprise-postgres-14-restore:ubi8-14-1.1-ppc64le

FEP pgpool2 container

Use the same naming convention for FEP pgpool2 containers as for FEP server containers.

fujitsu-enterprise-postgres-14-pgpool2: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	14	
<i>MajorVersion</i>	1,2, ...	To be used when major change in image, including server patch application
<i>MinorVersion</i>	0,1,2 ...	To be used when minor changes in image, e.g bug fix in container script

The first publishing will expect following names / tagging (Manifest and Child images)

- fujitsu-enterprise-postgres-14-pgpool2:ubi8-14-1.1
 - fujitsu-enterprise-postgres-14-pgpool2:ubi8-14-1.1-amd64
 - fujitsu-enterprise-postgres-14-pgpool2:ubi8-14-1.1-s390x
 - fujitsu-enterprise-postgres-14-pgpool2:ubi8-14-1.1-ppc64le

FEP Exporter container

FEP Exporter container as for FEP server containers.

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

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	14	
<i>MajorVersion</i>	1,2, ...	To be used when major change in image, including server patch application
<i>MinorVersion</i>	0,1,2 ...	To be used when minor changes in image, e.g bug fix in container script

The first publishing will expect following names / tagging (Manifest and Child images)

- fujitsu-enterprise-postgres-exporter:ubi8-14-1.1
 - fujitsu-enterprise-postgres-exporter:ubi8-14-1.1-amd64
 - fujitsu-enterprise-postgres-exporter:ubi8-14-1.1-s390x
 - fujitsu-enterprise-postgres-exporter:ubi8-14-1.1-ppc64le

FEP Fluentd container

FEP Fluentd container as for FEP server containers.

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

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	14	
<i>MajorVersion</i>	1,2, ...	To be used when major change in image, including server patch application
<i>MinorVersion</i>	0,1,2 ...	To be used when minor changes in image, e.g bug fix in container script

The first publishing will expect following names / tagging (Manifest and Child images)

- fujitsu-enterprise-postgres-fluentd:ubi8-14-1.1
 - fujitsu-enterprise-postgres-fluentd:ubi8-14-1.1-amd64
 - fujitsu-enterprise-postgres-fluentd:ubi8-14-1.1-s390x
 - fujitsu-enterprise-postgres-fluentd:ubi8-14-1.1-ppc64le

FEP Fluentbit container

FEP Fluentbit container as for FEP server containers.

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

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	14	
<i>MajorVersion</i>	1,2, ...	To be used when major change in image, including server patch application
<i>MinorVersion</i>	0,1,2 ...	To be used when minor changes in image, e.g bug fix in container script

The first publishing will expect following names / tagging (Manifest and Child images)

- fujitsu-enterprise-postgres-fluentbit:ubi8-14-1.1
 - fujitsu-enterprise-postgres-fluentbit:ubi8-14-1.1-amd64
 - fujitsu-enterprise-postgres-fluentbit:ubi8-14-1.1-s390x
 - fujitsu-enterprise-postgres-fluentbit:ubi8-14-1.1-ppc64le

FEP Cronjob container

FEP Cronjob container as for FEP server containers.

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

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	14	
<i>MajorVersion</i>	1,2, ...	To be used when major change in image, including server patch application
<i>MinorVersion</i>	0,1,2 ...	To be used when minor changes in image, e.g bug fix in container script

The first publishing will expect following names / tagging (Manifest and Child images)

- fujitsu-enterprise-postgres-cronjob:ubi8-14-1.1
- fujitsu-enterprise-postgres-cronjob:ubi8-14-1.1-amd64
- fujitsu-enterprise-postgres-cronjob:ubi8-14-1.1-s390x
- fujitsu-enterprise-postgres-cronjob:ubi8-14-1.1-ppc64le

FEP Upgrade container

FEP Upgrade container as for FEP server containers.

fujitsu-enterprise-postgres-14-upgrade: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	14	
<i>MajorVersion</i>	1,2, ...	To be used when major change in image, including server patch application
<i>MinorVersion</i>	0,1,2 ...	To be used when minor changes in image, e.g bug fix in container script

The first publishing will expect following names / tagging (Manifest and Child images)

- fujitsu-enterprise-postgres-14-upgrade:ubi8-14-1.1
- fujitsu-enterprise-postgres-14-upgrade:ubi8-14-1.1-amd64
- fujitsu-enterprise-postgres-14-upgrade:ubi8-14-1.1-s390x
- fujitsu-enterprise-postgres-14-upgrade:ubi8-14-1.1-ppc64le

2.3 Design Perspective for Each Feature

This section describes the design of each feature.

postgresql-cfg format

A postgresql-cfg represent ConfigMap for containing postgresql parameters. The file is used to contain the parameters which need to be reflected in postgresql.conf of the instance. Since patroni ignores all parameters which are not known by OSS postgresql.conf, an approach is defined to treat FEP Parameters in a special way.

The content of the ConfigMap is defined by key=value format. The following table shows the detail:

Spec	Example	Comment
The content may have multiple key/value pairs	foo=bar foo1=bar1	-
The value cannot have space unless quoted.	foo=bar bar2	Invalid
The quoted value cannot have another value after	foo='bar bar2' something	Invalid
The key value pair must have a '=' sign	-	-
White spaces are allowed before/after/between the key value pair	foo = bar	-
Any content after '#' will be ignored	# this is a comment foo=bar #this is a comment	-
The value may be quoted by single quotes	foo='bar bar2'	-
Single quote can be escaped by two single quotes	foo='It's ok'	Note: single quotes are not supported by Patroni edit-config command
Backslash '\' will be replaced by '\\' when invoking patronictl edit-config command	-	To avoid command line escape
When a key value pair is invalid, it will be ignored. the update continue to process next pair	foobar foo2=bar2	The 'foobar' will be ignored
The container script does not validate the key and value as long as they are in correct format.	-	-

It is recommended to use the psql's show command to verify parameter is setting correctly.

2.3.1 Deployment

Information for the FEPCluster

Equivalent Kubernetes command: `kubectl apply -f FEPClusterCR.yaml`

This operation will create a FEPCluster with supplied information in FEPClusterCR.yaml.

Refer to "FEPCluster parameter" in the Reference for details.

2.3.2 High Availability

Describes the settings for using the highly available features.

Arbitration

Patroni is used to control and monitor FEP instance startup, shutdown, status and trigger failover should the master instance fails. It plays a significant role in the solution. If the Patroni process dies, especially on master POD, without notice, the Pod will not update the Patroni cluster lock. This may trigger an unwanted failover to one of the Replica, without corresponding corrective action on the running master. This can create a split brain issue. It is important to monitor Patroni's status to make sure it is running. This is done using liveness probe. Important to note that this is not expected to be configured by end user.

```
livenessProbe:
  httpGet:
```

```

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

```

2.3.3 Configurable Volume per Cluster

Cluster node (Pod) volumes are created according to the values set in the storage section of `fehChildCrVal` in the FEPCluster custom resource.



Note

- After you create the FEPCluster for the first time, you cannot add new volumes later or modify the `storageClass` or `accessModes`.
- You can resize the initially created volume only if the underlying `storageClass` supports dynamic resizing.

The following is the schema for the storage section of the FEPCluster customer resource:

Field	Mandatory	Sub-Field	Default	Description
archivewalVol	No	size	1Gi	Volume size of the archive log. Refer to "Estimating Database Disk Space Requirements" in the FUJITSU Enterprise Postgres Installation and Setup Guide for Server to help you design the size.
		storageClass	Defaults to platform default if omitted	SC is only set at start
		accessModes	Defaults to ReadWriteOnce if omitted	Access mode is only set at start
backupVol	No	size	2Gi	Volume size of the backup. Estimate based on the following formula: (full backup generations + incr backup generations + 1) * dataVol size
		storageClass	Defaults to platform default if omitted	SC is only set at start
		accessModes	Defaults to ReadWriteOnce if omitted	Access mode is only set at start
dataVol	Yes	size	2Gi	Volume size of the data. Refer to "Estimating Database Disk Space Requirements" in the FUJITSU Enterprise Postgres Installation and Setup Guide for Server and base the design on table/index size.
		storageClass	Defaults to platform default if omitted	SC is only set at start

Field	Mandatory	Sub-Field	Default	Description
		accessModes	Defaults to ReadWriteOnce if omitted	Access mode is only set at start
logVol	No	size	1Gi	Volume size of the log. If you change the log output level (default: WARNING) or enable the audit log feature, measure the actual amount of log output in a test environment.
		storageClass	Defaults to platform default if omitted	SC is only set at start
		accessModes	Defaults to ReadWriteOnce if omitted	Access mode is only set at start
tablespaceVol	No	size	512Mi	Volume size of the tablespace. When using tablespaces, as with dataVol, you should refer to "Estimating Database Disk Space Requirements" in the FUJITSU Enterprise Postgres Installation and Setup Guide for Server for information on sizing.
		storageClass	Defaults to platform default if omitted	SC is only set at start
		accessModes	Defaults to ReadWriteOnce if omitted	Access mode is only set at start
walVol	Yes	size	1200Mi	Volume size of the transaction log. Refer to "Estimating Database Disk Space Requirements" in the FUJITSU Enterprise Postgres Installation and Setup Guide for Server to help you design the size. Note that the default value for max_wal_size is 1 GB.
		storageClass	Defaults to platform default if omitted	SC is only set at start
		accessModes	Defaults to ReadWriteOnce if omitted	Access mode is only set at start

The 'accessMode' is been incorporated for the inclusion of pgBadger layer later. Giving it a shared volume capability will allow pgBadger Container to read logs from multiple server instance (master / replica) and expose it via a WebServer.

2.3.4 Deploying Pgpool-II and Connect to FEPCluster from Operator

Equivalent Kubernetes command: `kubectl create FEPpgpool2`

This operation will create a FEP pgpool2 container with supplied information.

Refer to "FEPpgpool2 Custom Resource Parameters" in the "Reference" for more information.

2.3.5 Scheduling Backup from Operator

When creating a FEPCluster, users can obtain scheduled backups by setting up backup definitions. Users can also modify the backup schedule by modifying the Backup custom resource that was created.

A backup definition includes the following:

- Acquisition time (Specify in crontab format)
- Backup type (Full or incremental backups)

Backup is taken on master Pod only.

Backup processing is performed by pgBackRest.

Parameter can be set to pgbackrestParams in CR definition.

The maximum number of backup schedules is 5.

See the pgBackRest User's Guide for details on the parameters.

However, some parameters are limited. Details are given below.

- [2.3.5.1 Important Setting Items](#)
- [2.3.5.2 Parameters that cannot be Set](#)
- [2.3.5.3 Restricted Parameters](#)
- [2.3.5.4 About Sections in the Config File](#)

2.3.5.1 Important Setting Items

Here are the important parameters for setting pgBackRest. This parameter sets the retention period of backup information. If automatic backup is set and this parameter is not set, the risk of overflowing the backup area increases.

Parameter	Overview of parameters	Setting value
Full Retention Option (repo retention -full)	Specify number of full backups to keep No default (should be set according to user backup policy)	natural number
Full Retention Type Option (repo retention-full-type)	spec.retention -full Specifies whether the setting is a number of retention days (time) or a number of retention times (count) No default (should be set according to user backup policy)	time/count

The following is a sample CR example of changing the backup retention period (How long the PITR is valid) to 30 days after a FEPCluster deployment by setting the above parameters.

```
apiVersion: fep.fujitsu.io/v1
kind: FEPClusterBackup
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 Parameters that cannot be Set

The following parameters in the pgBackRest Configuration Reference are not configurable.

Parameter	Overview of parameters	Reason
Copy Archive Option (--archive -copy)	Copy the WAL segments needed for consistency to the backup	To use internal fixed values
Check Archive Mode Option (--archive-mode-check)	Check the PostgreSQL archive_mode setting.	Limited to backup from master
Backup from Standby Option (--backup-standby)	Back up from the standby cluster	Limited to backup from master
Stop Auto Option (--stop-auto)	Stops a previously failed backup on a new backup.	Because they are 9.6 not supported in
SSH client command Option (--cmd-ssh)	Path to ssh client executable	Not using ssh
Compress Option (--compress)	Use File Compression	For obsolete options (Use compress-type option instead)
Config Option (--config)	pgBackRest configuration file.	To use internal fixed values
Config Include Path Option (--config-include-path)	Path to additional pgBackRest configuration files.	To use internal fixed values
Config Path Option (--config-path)	Base path of pgBackRest configuration files.	To use internal fixed values
Delta Option (--delta)	Restore or Backup with Checksum	For new restores only
Dry Run Option (--dry-run)	Execute a dry-run for the command.	command-line only option
Lock Path Option (--lock-path)	Path where the lock file is stored	To use internal fixed values
Keep Alive Option (--sck -keep-alive)	Enable keep-alive messages on socket connections	To use internal fixed values
Spool Path Option (--spool-path)	Path to store temporary data for asynchronous archive-push and archive-get commands	For automatic determination from FEPCluster CR values
Stanza Option (--stanza)	Defines the stanza.	To use internal fixed values
Console Log Level Option (--log-level-console)	Console Log Level	It is not expected to operate on Pod.
Std Error Log Level Option (--log-level-stderr)	Stderr log level	It is not expected to operate on Pod.
Log Path Option (--log-path)	Log File Destination	For automatic determination from FEPCluster CR values
Repository Host Option (--repo-host)	Repository host for remote operations via SSH	Repository Host is not used
Repository Host Command Option (--repo-host-cmd)	Path of pgBackRest on Repository Host	
Repository Host Configuration Option (--repo-host-config)	Repository Host Configuration File Path	
Repository Host Configuration Include Path Option (--repo-host-config-include-path)	Repository hosts configuring include path	

Parameter	Overview of parameters	Reason
Repository Host Configuration Path Option (--repo-host-config-path)	Repository Host Configuration Path	
Repository Host Port Option (--repo-host-port)	Repository host port when "repo-host" is configured	
Repository Host User Option (--repo-host-user)	Repository host user when "repo-host" is configured	
Repository Path Option (--repo-path)	Path where backups and archives are stored	For automatic determination from FEPCluster CR values
Archive Retention Option (--repo-retention-archive)	The number of consecutive WAL backups to keep.	This option is not recommended, and WAL retention is controlled by the Full Retention Option and Full Retention Type Option.
Archive Retention Type Option (--repo-retention-archive-type)	Backup Type for WAL Retention	It is recommended not to change from the default.
Differential Retention Option (--repo-retention-diff)	Number of incremental backups to keep	No incremental backups
Archive Mode Option (--archive-mode)	Retains or disables the archive for the restored cluster.	To use internal fixed values
Exclude Database Option (--db-exclude)	Restore excluding the specified databases.	To restore the entire FEP cluster, including all databases
Include Database Option (--db-include)	Restore only the specified database	To restore the entire FEP cluster, including all databases
Link All Option (--link-all)	Restore all symbolic links.	To use internal fixed values
Link Map Option (--link-map)	Changes the destination of a symbolic link.	To use internal fixed values
Recovery Option Option (--recovery-option)	Setting options in postgresQL recovery.conf	To use internal fixed values
Tablespace Map Option (--tablespace-map)	Restoring tablespace to a specified directory	For automatic determination from FEPCluster CR values
Map All Tablespaces Option (--tablespace-map-all)	Restores all tablespaces to the specified directory	No tablespace required because there is only one tablespace per FEPCluster
PostgreSQL Database Option (--pg-database)	PostgreSQL database.	To use internal fixed values
PostgreSQL Host Option (--pg-host)	PostgreSQL host for remote operations via SSH	No SSH connection required
PostgreSQL Host Command Option (--pg-host-cmd)	Path of pgBackRest exe on the PostgreSQL host	To use internal fixed values
PostgreSQL Host Configuration Option (--pg-host-config)	Path of the pgBackRest configuration file	To use internal fixed values
PostgreSQL Host Configuration Include Path Option (--pg-host-config-include-path)	Setting pgBackRest on PostgreSQL host include path	To use internal fixed values

Parameter	Overview of parameters	Reason
PostgreSQL Host Configuration Path Option (--pg-host-config-path)	Path to configure pgBackRest on the PostgreSQL host	To use internal fixed values
PostgreSQL Host Port Option (--pg-host-port)	SSH Port Specification	No SSH connection required
PostgreSQL Host User Option (--pg-host-user)	The logon user when hosting PostgreSQL, if pg-host is set.	No SSH connection required
PostgreSQL Path Option (--pg-path)	PostgreSQL data directory.	For automatic determination from FEPCluster CR values
PostgreSQL Port Option (--pg-port)	PostgreSQL Ports	For automatic determination from FEPCluster CR values
PostgreSQL Socket Path Option (--pg-socket-path)	PostgreSQL Unix socket path	For automatic determination from FEPCluster CR values
PostgreSQL Database User Option (--pg-user)	PostgreSQL database user	To use internal fixed values

2.3.5.3 Restricted Parameters

Of the parameters in the pgBackRest Configuration Reference, the following parameters limit the configurable values.

Parameter	Overview of parameters	Possible Values
repoX-gcs-key-type	The type of key file you specify when using Google Cloud Storage	service

2.3.5.4 About Sections in the Config File

In FEPCluster CR, you can write the contents of pgbackrest.conf, but the setting for stanza (Backup space for pgBackRest) is specified internally.

The following sections are not allowed;

[stanza: command] , [stanza]

2.3.6 Perform PITR and Latest Backup Restore from Operator

There are two types of restore: one is to restore backup data to an existing FEPCluster, and the other is to create a new FEPCluster and restore backup data.

The former retains the attributes of the FEPCluster, such as IP address and name, while the latter is created from scratch.

The restore process deploys a FEP restore container. The FEP restore container performs the pgBackRest restore operation from the backup data to be restored to the master server of the FEPCluster. After the data is restored to the master server, the FEPCluster is created by synchronizing the data to two replica servers.

If user create a new FEPCluster, the newly created FEPCluster will inherit the settings of the source cluster, unless otherwise specified

User can also create a cluster with different settings from the source cluster by including the settings in FEPRestore CR.

Switching connections to the new cluster

The restore creates a new FEPCluster. If necessary, you need to set up Pgpool-II and change the access point of the application to the new cluster or the new Pgpool-II.

About recovering a failed FEPCluster

Even if the existing FEPCluster fails and the FEP is not running, if the volume of the backup area is safe, it is possible to restore from the backup data.

2.3.7 FEP Unique Feature Enabled by Default

Enable the following FEP features:

- Data masking
- Transparent Data Encryption (TDE)

Data masking

The Data masking is enabled by default in the example FEPClster CR (in openshift UI). The postgresql.conf in container contains the following parameters:

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

The user can overwrite these values in config map.

TDE

TDE is enabled by default. For details on how to specify the passphrase, refer to "FEPCluster parameter" in the Reference.

2.3.8 Monitoring & Alert (FEPEXporter)

As the operator is level 5 certified, the system expose various metrics about its operand i.e. FEP containers.

FEP generates lot of useful database statistics via various views. The default statistics can be further augmented by using extensions like pg_stat_statements.

FEPEXporter container by default is configured to extract useful database statistics and make the metrcs available to Prometheus on the platform. External components and utilities can be used to visualise, analyse, trigger alerts and take operational decision based on exposed metrics.

FEPEXporter also sets default alert rules based on Prometheus metrics which are useful for active monitoring of FEP cluster.

2.3.8.1 FEPEXporter Custom Resource

Refer to "FEPEXporter Custom Resource" in the Reference for FEPEXporter Custom Resource parameters.

- Custom queries to scrape metrics can be added in CR in optional section.
- Custom Prometheus alert rules are created by user manually.

2.3.8.2 Change to FEPCluster CR - metrics user

User may define pgMetricsUser, pgMetricsPassword and pgMetricsUserTls in target FEPCluster. If it is defined, FEPEXporter will use metrics user details to connect to FEP cluster machines. All metrics user fields are optional and can be omitted in FEPCluster.

Refer to "FEPCluster Parameter" in the Reference for FEPCluster parameters.

2.3.8.3 FEPEXporter CR auto-create for FEPCluster

User may define enableMonitoring flag as part of FEPCluster CR to monitor FEPCluster. It will automatically create FEPCluster specific FEPEXporter so matrices scraping for FEPCluster will work.

Refer to "FEPCluster Parameter" in the Reference for FEPCluster parameters.

- FEPEXporter will be named as <cluster-name>-fepexporter.
- Once FEPEXporter created automatically, user can modify it manually from FEPEXporter CR.
- If FEPCluster will be deleted, it will delete dependent FEPEXporter as well.
- MTLS for FEPEXporter will only supported when tls configuration defined for both Prometheus & FEPEXporter specs.

2.3.9 Scaling Replicas

Auto scale out occurs when the average database CPU utilization or number of connections exceeds the threshold. Select whether the criteria for auto scale out is CPU usage or the number of connections, depending on the resource that is the bottleneck of the database.

The maximum number of replica containers, excluding the master container, is 15.

Scale out based on CPU utilization

Performs a scale out if the average CPU utilization of all pods (primary pods and all replica pods) in the FEPCluster exceeds the threshold for a period of time.

CPU utilization is calculated with the value specified in `spec.fep.mcSpec.requests.cpu` specified for the FEPCluster custom resource as the denominator.

Scale out based on the number of connections

Performs a scale out if the average number of connections for all pods (primary pods and all replica pods) in the FEPCluster exceeds the threshold for a period of time.

Specify the threshold for the number of connections to perform automatic scale-out with a value less than or equal to the `max_connections` parameter of the FEP server.

The prerequisites for using the scale out feature based on the number of connections are as follows.

- The monitoring feature (see "[2.3.8 Monitoring & Alert \(FEPEXporter\)](#)") is enabled.
- Metrics for the number of FEP server connections are collected by the monitoring feature.
- A custom metrics server is installed in the OCP/Kubernetes cluster.
- The custom metrics server publishes the average number of connections collected by the monitoring feature.

When using the scale out feature based on the number of connections, the auto scale out feature requests the custom metrics server for metrics associated with the following Kubernetes resources.

- `kind`: FEPCluster
- `apiVersion`: `fep.fujitsu.io/v2`
- `name`: Name of FEP Cluster
- `namespace`: The name of the namespace in which FEP Cluster is deployed

The name of the requested metric is the name specified in the `metricName` parameter.

This metric should represent the average number of connections for each pod in the specified FEPCluster.

Limitations

- If you want to use the scale out feature based on the number of connections, deploy FEPEXporter according to the procedure of "[4.3 Deploying FEPEXporter](#)".
- If FEPCluster metrics are collected by FEPEXporter in standalone mode (see "[4.4 FEPEXporter in Standalone Mode](#)"), the scale out feature based on the number of connections is not available.



Note

When using the auto scale out feature, the FEPCluster sync mode should be "off".

Precautions when designing auto scale out

- The auto scale out feature adds replicas one at a time. In addition, additional replicas take time to service, depending on the environment and the amount of data stored. As a result, replica growth may not be able to keep up with the increased load.

- Even if the auto scale out feature increases the number of replicas, incoming requests are not given priority to those replicas. As a result, existing FEP instances may continue to be temporarily overloaded after the number of replicas increases.
- The auto scale out feature increases the number of replica requests that can be handled only by reference requests to the database. Requests with updates continue to be processed on the primary FEP instance. Therefore, the auto scale out feature may not reduce the load on the primary FEP instance.
- Currently, the auto scale out feature does not delete replicas (reduce the number of replicas). If the load decreases after the number of replicas increases due to a temporary increase in load, the number of replicas remains increased. If necessary, manually change the number of replicas.

2.3.9.1 Change to FEPCluster CR - auto scale out

If you want to use Auto Scale Out, set the parameter to FEPClusterCR.

Refer to "FEPCluster Parameter" in the Reference for FEPCluster parameters.

2.3.10 Disaster Recovery

By using OSS (pgBackRest) functionality to store backup data in object storage, data can be migrated to a database cluster in a different OCP environment.

Even if it is difficult to operate in an OCP environment with a database cluster due to a disaster, it is possible to continue operating in a different OCP environment.

2.3.11 Transparent Data Encryption Using a Key Management System

FUJITSU Enterprise Postgres provides unique features that enhance the security of PostgreSQL. These security features help users keep their data safe from unauthorized access. One such security feature is Transparent Data Encryption (TDE), which encrypts data at rest, i.e. data stored on disk/persistent volume.

In contrast, TDE's default format stores the master encryption key in a password-protected file. A key management system allows you to store your master encryption key (MEK) in a cloud-based keystore, taking your security to the next level.

Refer to the FUJITSU Enterprise Postgres Installation and Setup Guide for Server for the requirements of the key management system.

Transparent data encryption using a key management system can only be configured when the FEPCluster is first created. Users cannot configure an existing FEPCluster for transparent data encryption using a key management system.

If the master encryption key on the key management system is lost, the encrypted/backup data cannot be decrypted. As long as the data encrypted with the master encryption key remains valid, the master encryption key must also be available and maintained on a key management system.

Even if the master encryption key is updated by key rotation, if you have encrypted backup data with the old encryption key, you must keep the old encryption key.

In addition, the key custodian must retain the referenced master encryption key for as long as the data encrypted under the old master encryption key remains valid.

Chapter 3 Operator Installation

This chapter describes how to install FEP operator.

Refer to "6.5 Assigned Resources for Operator Containers" for more information about the resources assigned to installed operator containers and how to change them.

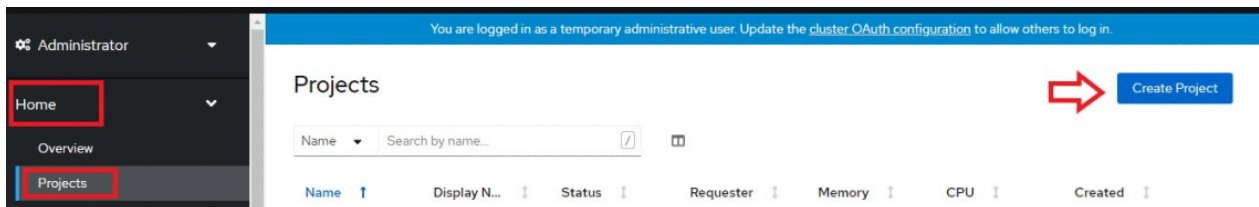
3.1 Using the OperatorHub

Describes how to use OperatorHub to install FEP operators into a new namespace on Openshift.

3.1.1 Pre-requisite

A project on openshift is essentially a namespace. It is a good practice to install FEP in a separate name space. On the RedHat OpenShift platform, click "Home" under "Projects" main menu and hence click on "Create Project".

(Screen Shot 1 and 2 - Create Project on OCP - *for ref.*)



In the dialog box, specify a unique name for your namespace and an optional display name and description.

Create Project

Name * ?

Display name

Description

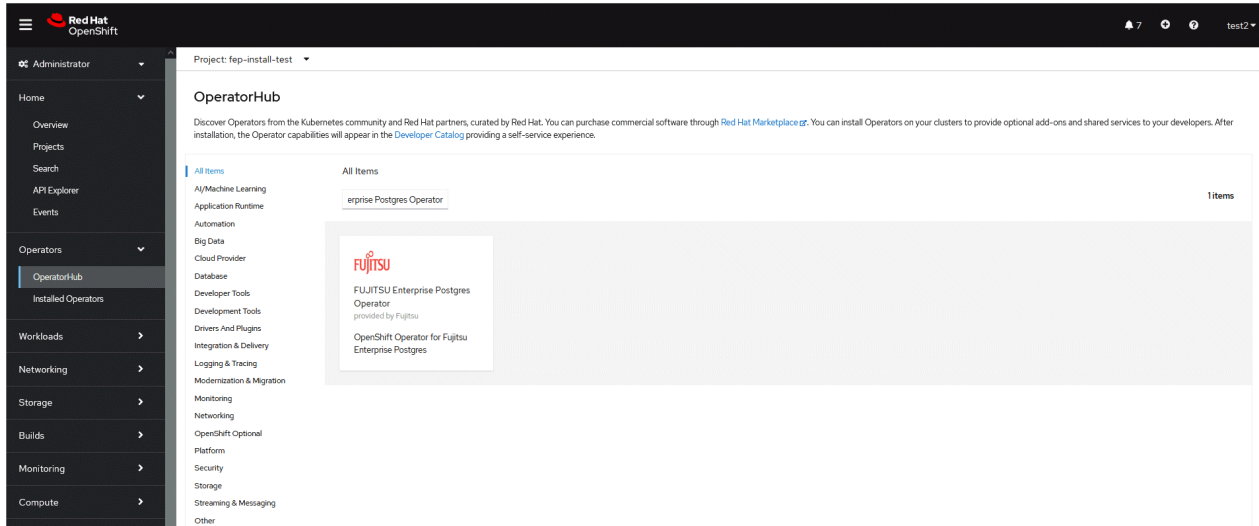


Operator installation needs Prometheus to be pre-installed in the Openshift cluster.

3.1.2 Deploying Operator

Once operator is certified by RedHat, it is made available on OperatorHub on all RedHat OpenShift container platform.

On OpenShift platform, logon with credentials that has privileges to install operator. Click on OperatorHub on menu item under Operators and type filter keyword "FUJITSU Enterprise Postgres Operator" to find FUJITSU Enterprise Postgres Operator.



Click on FUJITSU Enterprise Postgres Operator to install operator. It will bring up details page with install button as below.



FUJITSU Enterprise Postgres Operator

4.1.0 provided by Fujitsu



Install

Latest version

4.1.0

FUJITSU Enterprise Postgres 14 delivers an enterprise-grade PostgreSQL on OpenShift Container Platform.

Capability level

- ✓ Basic Install
- ✓ Seamless Upgrades
- ✓ Full Lifecycle
- ✓ Deep Insights
- ✓ Auto Pilot

This solution provides the flexibility of a hybrid cloud solution while delivering an enhanced distribution of PostgreSQL to support enterprise-level workloads and provide improved deployment and management, availability, performance, data governance and security.

Available as a multi-architecture container built for both amd64 and s390x.

The download and Use of the Product is strictly subject to the terms of the End User License Agreement with Fujitsu Limited found at <https://www.fast.fujitsu.com/fujitsu-enterprise-postgres-license-agreements>. Where the Product that has been embedded as a whole or part into a third party program, only Authorised Customers may download and use the Product.

Source

Certified

Provider

Fujitsu

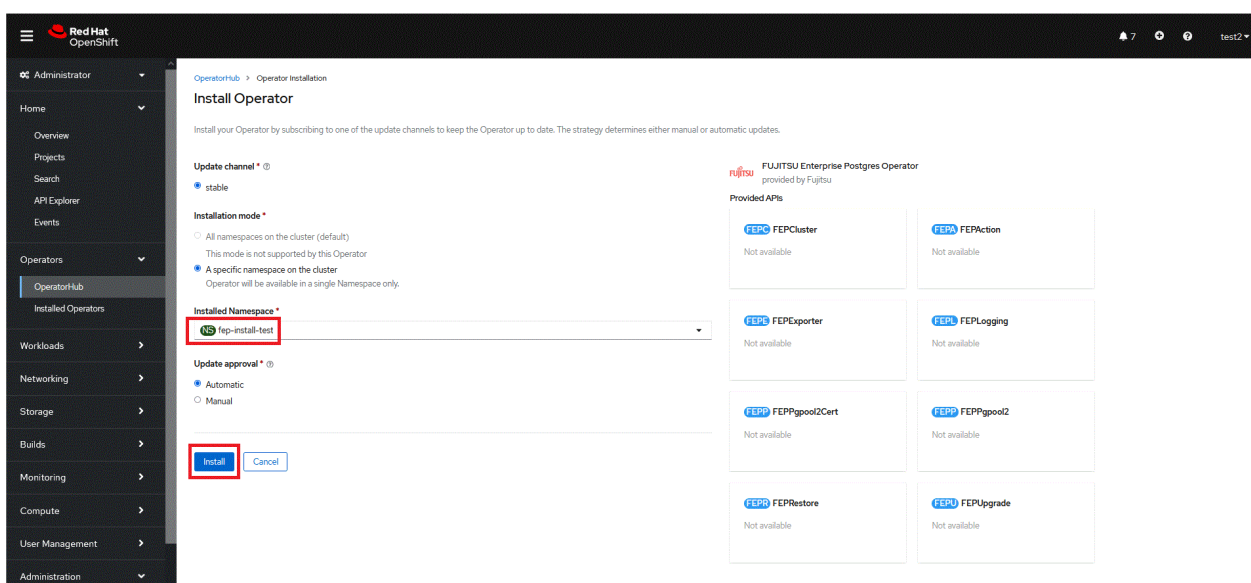
Repository

N/A

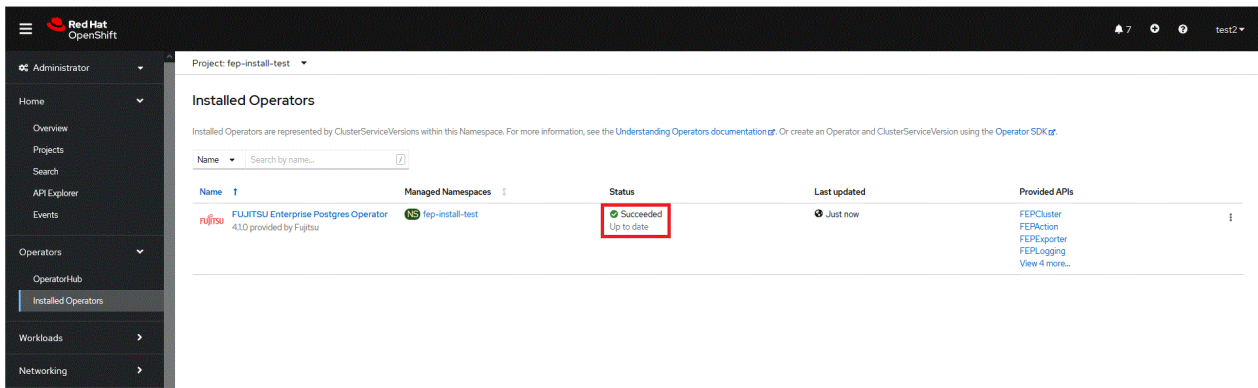
Container image

quay.io/fujitsu/fujitsu-enterprise-postgres-operator@sha256:0bc97f1c1f2af9f059349c8a8fcd3d16800fbb5f3993ae1f5859776ef713543c

Click on "Install" button, to bring up following screen to choose namespace and approval strategy. Select "A specific namespace on the cluster" and choose desired namespace. Leave everything else to default and click install.



Wait still installation is complete and status changes to "Succeeded".



3.2 Using the Helm Chart

Describes how to install FEP operators into a new namespace on Kubernetes using the Helm feature.

3.2.1 Deploying Operator

1. Add a Helm Chart repository for the operator.

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

2. Create a namespace to install the operator.

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



Note

Operator installation needs Prometheus to be installed in the Kubernetes cluster in advance.

3. Run the helm command to install the operator.

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

3.2.2 Upgrading Operators

1. Refresh Helm Chart repository information.

```
helm repo update
```

2. Check the Helm Chart version of the latest operator.

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

3. Run the helm command to upgrade the operator.

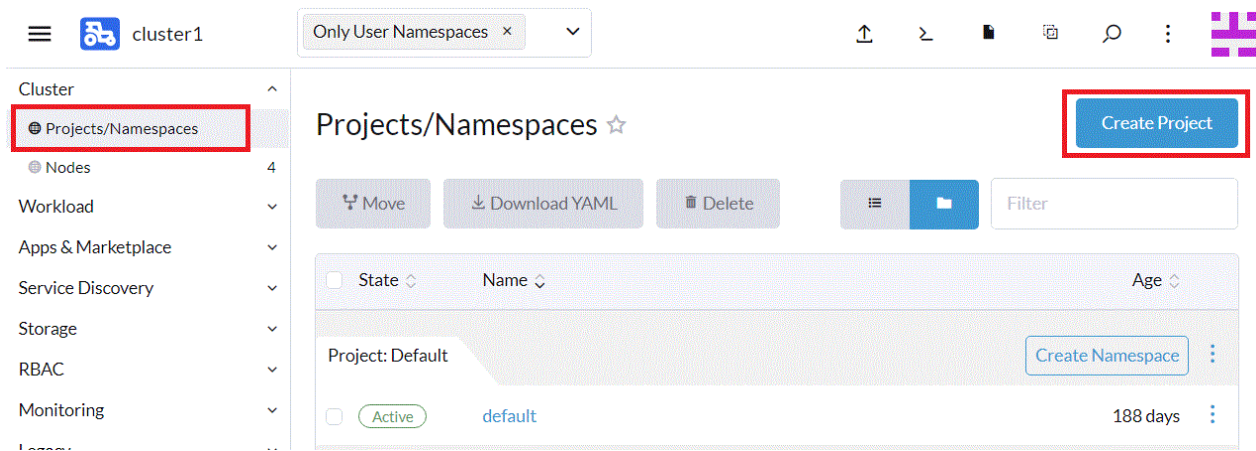
```
helm upgrade fep-operator-release fep-repo/fujitsu-enterprise-postgres-operator --namespace fep-operator
```

3.3 Using the Rancher UI

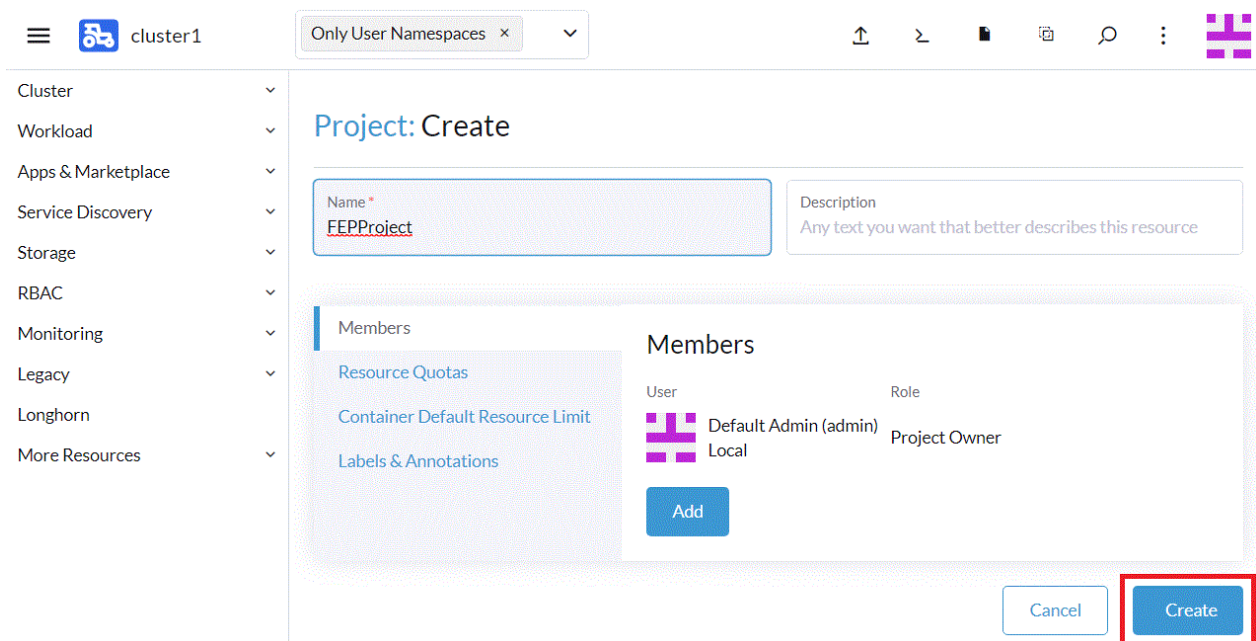
Describes how to install FEP operators into a new namespace on Rancher UI.

3.3.1 Pre-requisite

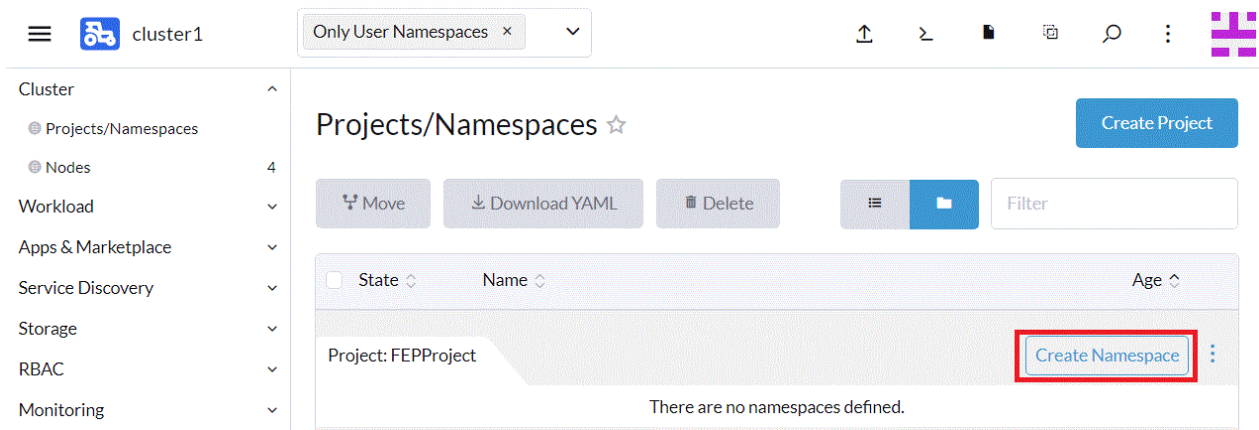
Create a project and its associated namespace on the Rancher UI. We recommend that you install FEP in a different namespace. In the Rancher UI, click [Projects/Namespaces], then click [Create Project] that appears.



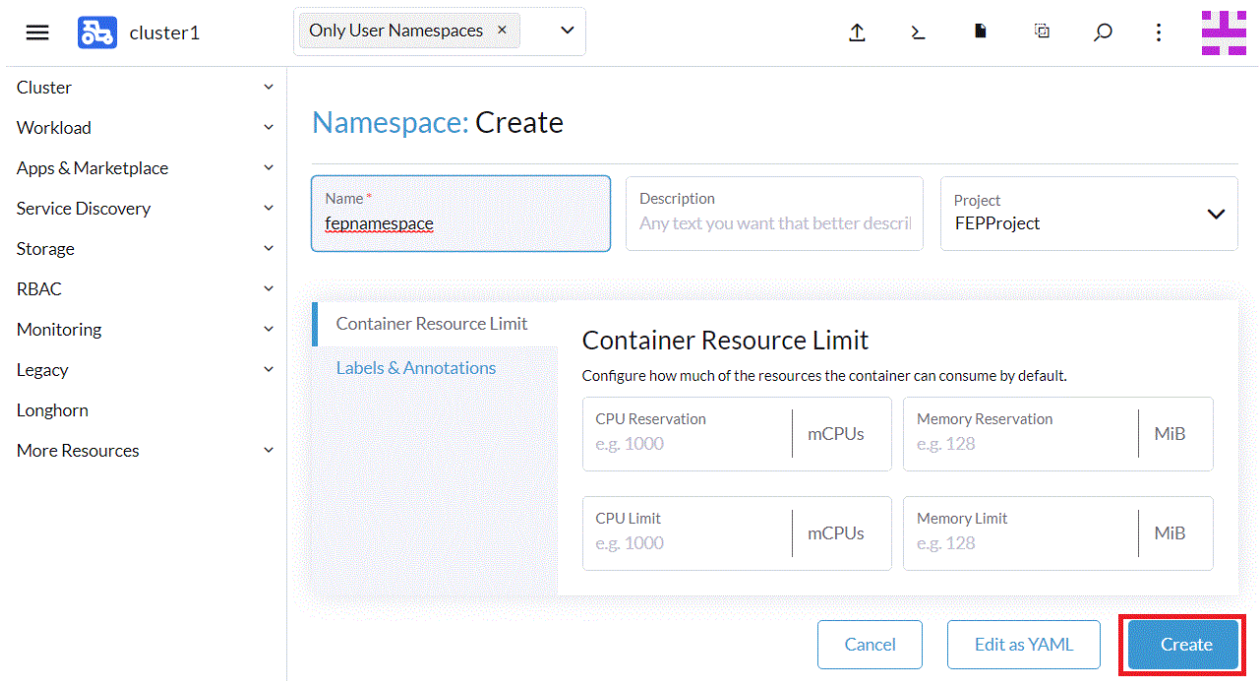
Specify a unique name for the project and click [Create].



Click [Create Namespace] displayed on the specified project.



Specify a unique name in the namespace and click [Create].



3.3.2 Register Helm Chart Repository

Register the Helm Chart repository of the operator feature on the Rancher UI.

In the Rancher UI, click [Apps & Marketplace], then click [Repositories] that appears.

cluster1 Only User Namespaces

Cluster
Workload
Apps & Marketplace
Charts
Installed Apps 4
Repositories 3
Recent Operations 0
Service Discovery
Storage
RBAC
Monitoring
Legacy

A chart repository is a Helm repository or Rancher git based application catalog. It provides the list of available charts in the cluster.

Repositories ☆

Create

Refresh Download YAML Delete Filter

State	Name	Type	URL	Branch	Age
Active	Partners	git	https://git.rancher.io/partner-charts	main	188 days
Active	Rancher	git	https://git.rancher.io/charts	release-v2.6	188 days

Click [Create] to create the Helm Chart repository.

cluster1 Only User Namespaces

Cluster
Workload
Apps & Marketplace
Charts
Installed Apps 4
Repositories 3
Recent Operations 0

A chart repository is a Helm repository or Rancher git based application catalog. It provides the list of available charts in the cluster.

Repositories ☆

Create

Refresh Download YAML Delete Filter

Enter the unique name of the catalog and the URL of the catalog below, and click [Create].

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

cluster1

Cluster
Workload
Apps & Marketplace
Charts
Installed Apps
Repositories
Recent Operations
Service Discovery
Storage
RBAC
Monitoring
Legacy
Longhorn
More Resources

Only User Namespaces

⬆ ⬇ 📄 🔍 ⋮

Repository: Create

Name *

fep-operator-helm

Description

Any text you want that better describes this resource

Target

☒ http(s) URL to an index generated by Helm

☐ Git repository containing Helm chart or cluster template definitions

Index URL *

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

Authentication

None

Labels

Add Label

Annotations

Add Annotation

Cancel

Create

3.3.3 Deploying Operator

On the Rancher UI, apply the operator function Helm Chart to the project / namespace created in ["3.3.1 Pre-requisite"](#) and install the operator.

From the leftmost tab, click [Charts], then click [fujitsu-enterprise-postgres-operator].

cluster1

Only User Namespaces x

Cluster

Workload

Apps & Marketplace

Charts

Installed Apps

Repositories

Recent Operations

Service Discovery

Storage

Charts

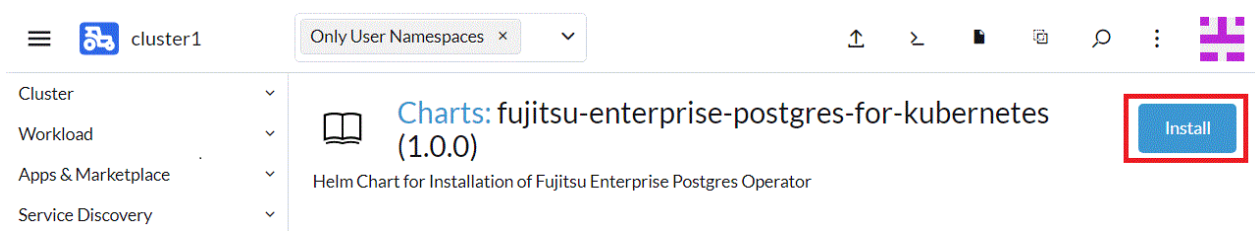
fep-operator-helm

All Categories

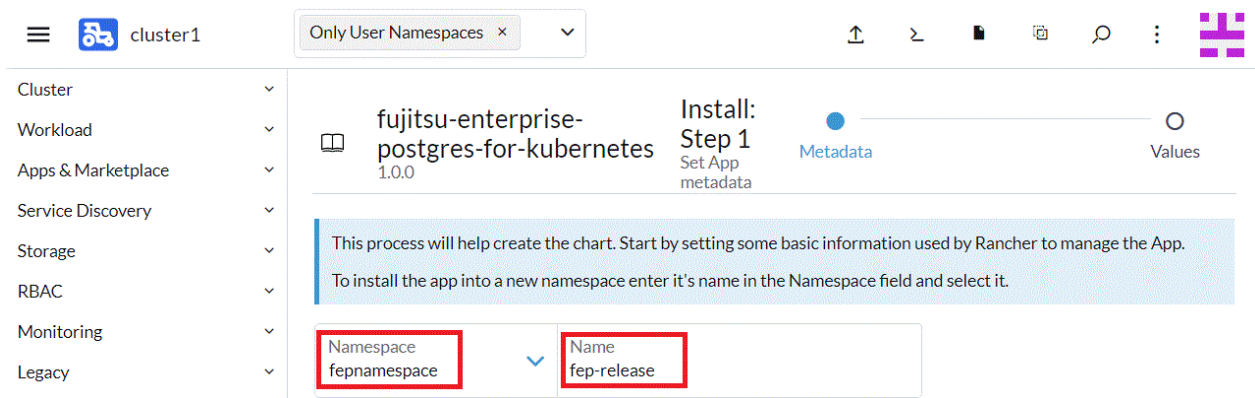
Filter

fujitsu-enterpris...
Helm Chart for
Installation of
Fujitsu Enterprise...

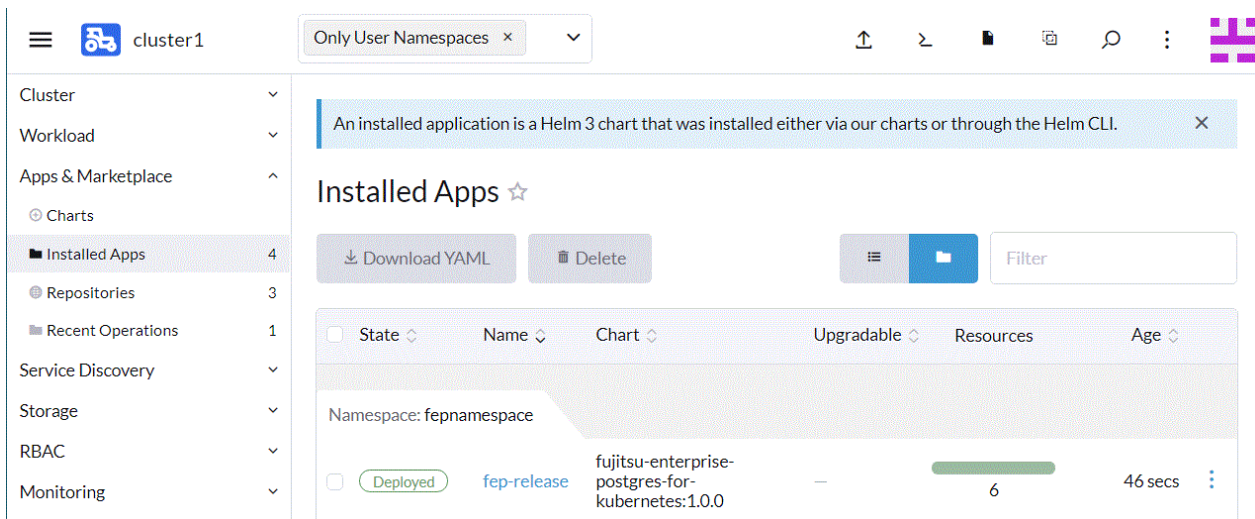
Click Install on the screen that appears.



Change the [Namespace] item to the name created in "3.3.1 Pre-requisite", enter the release name in the [Name] item, click [Next], and then click [Install] on the next screen.



The operator is deployed on the target namespace.



3.4 Implement Collaborative Monitoring Tools

There is a pre-requisite for running FEPEXporter.

- GAP(Grafana, AlertManager, Prometheus) stack is installed on host OpenShift or Kubernetes cluster
- FEPCluster that needs to be scraped is deployed and running properly
- FEPCluster has following setting postgresql.conf:
 - pg_stats_statements library pre-loaded

- track_activities and track_counts are turned on

For Prometheus and AlertManager, use the monitoring stack preinstalled on Openshift. Please refer to the following for deployment information.

(https://docs.openshift.com/container-platform/4.9/monitoring/monitoring-overview.html#understanding-the-monitoring-stack_monitoring-overview)

For Grafana, install and use the Grafana Operator provided by OperatorHub for x86. Grafana is not exposed by OperatorHub in s390x and ppc64le, so use Helm to build Grafana. Detailed instructions are available at the following site for your reference.

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

Grafana comes pre-installed on OpenShift, but it is recommended to use Grafana published in OperatorHub to customize the dashboard and monitor FEP performance information.

3.5 Implement Client

To use the FEP client, use the media or download the rpm module from the following site.

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

Chapter 4 Deployment Container

This chapter describes container deployment.



Note

Each volume of a Pod created by a FEPCluster deployment is sized by default for the following operations:

- Data size: 1 GB
- Daily update: about 50 MB

Refer to "2.3.3 Configurable Volume per Cluster" to design each volume size according to actual operation.

4.1 Deploying FEPCluster using Operator

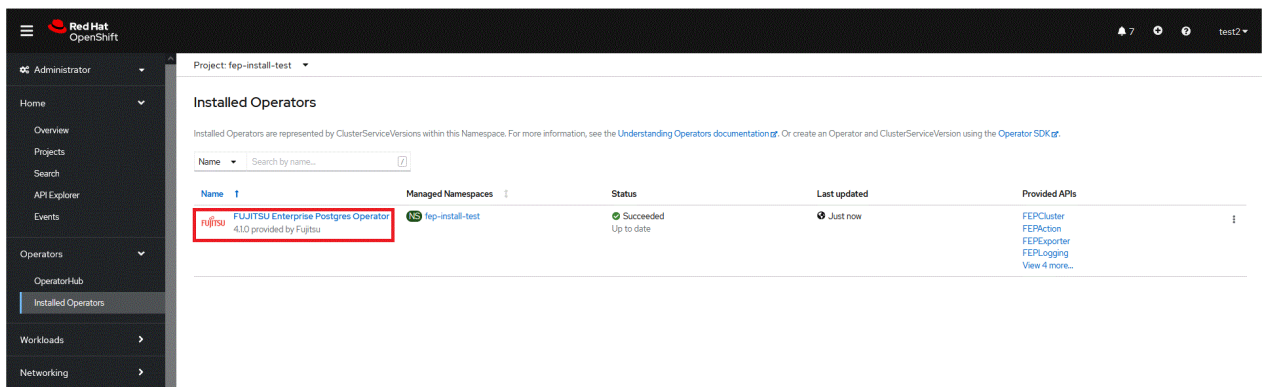
To deploy a FEPCluster in given namespace, follow these steps:



Note

If you are deploying on a Kubernetes cluster, Refer to "Custom Resource Parameters" in the Reference to create and apply a yaml file.

1. Under "Operators" menu item, click on "Installed Operators". You would see the installed FEP operator deployed in "Chapter 3 Operator Installation". Click on the name of operator.



2. It will display a page with all CRs this operator supports. FEPCluster is the main CR and all others are child CR. We would create the main CR and all other CRs will be created automatically by Operator.
To create Cluster CR, either
(1) Click on "Create Instance" under FEPCluster.

OR

(2) Click on "**FEPCluster**" on top and then click on "**Create FEPCluster**" on the next page.

Project: fep-install-test

Installed Operators > Operator details

FUJITSU Enterprise Postgres Operator
4.1.0 provided by Fujitsu

(2) FEPCluster

Details YAML Subscription Events All Instances FEPCluster FEPAAction FEPEXporter FEPLogging FEPPgpool2Cert FEPPgpool2 FEPRestore FEPUUpgrade

Provided APIs

API	Status	Action
FEPCluster	Not available	(1) Create instance
FEPAAction	Not available	Create instance
FEPEXporter	Not available	Create instance
FEPLogging	Not available	Create instance
FEPPgpool2Cert	Not available	Create instance
FEPPgpool2	Not available	Create instance
FEPRestore	Not available	Create instance
FEPUUpgrade	Not available	Create instance

Description

FUJITSU Enterprise Postgres 14 delivers an enterprise-grade PostgreSQL on OpenShift Container Platform.

This solution provides the flexibility of a hybrid cloud solution while delivering an enhanced distribution of PostgreSQL to support enterprise-level workloads and provide improved deployment and management, availability, performance, data governance and security.

Available as a multi-architecture container built for both amd64 and s390x.

The download and Use of the Product is strictly subject to the terms of the End User License Agreement with Fujitsu Limited found at <https://www.fujitsu.com/fujitsu-enterprise-postgres-license-agreements>. Where the Product that has been embedded as a whole or part into a third party program, only Authorised Customers may download and use the Product.

Provider: Fujitsu

Created at: Mar 28, 2022, 9:45 AM

Links: <https://www.postgresql.fastware.com/>

Maintainers: Fujitsu, pgtechnquiry@au.fujitsu.com

3. This will bring to "Create FEPCluster" page. Here you have two options to configure. The first one is Form View. At the moment, in Form View, one can change only the name of cluster being deployed. The default name is "new-fep". This name must be unique within a namespace.

Project: fep-install-test

FUJITSU Enterprise Postgres Operator > Create FEPCluster

Create FEPCluster

Create by completing the form. Default values may be provided by the Operator authors.

Configure via: Form view (selected) YAML view

Note: Some fields may not be represented in this form. Please select "YAML View" for full control of object creation.

Name *

new-fep

Labels

app=frontend

Create Cancel

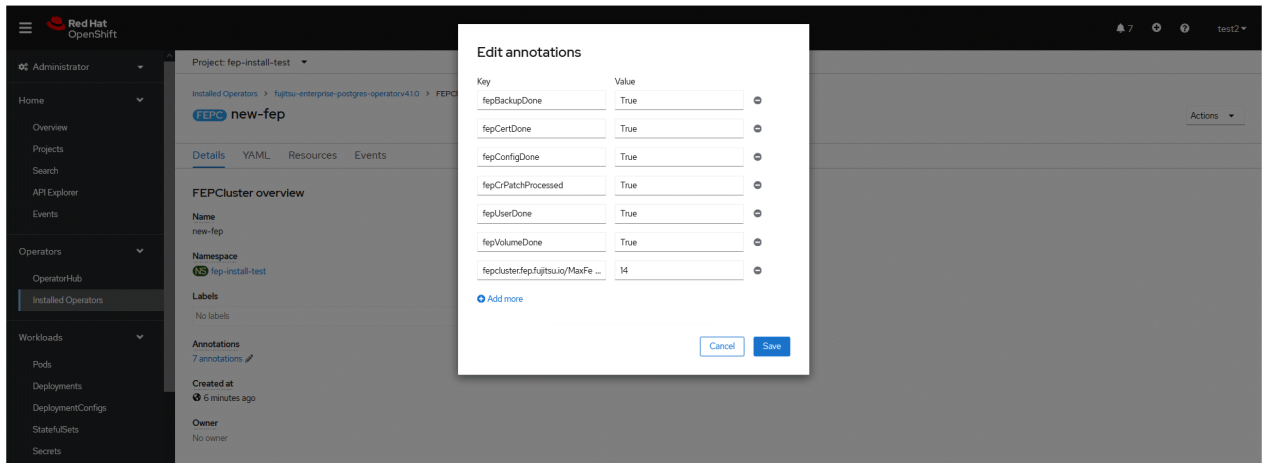
- In YAML View, starting value of CR is visible and one can choose to modify parameters before creating CR. Refer to the Reference for details of parameters.

The screenshot shows the Red Hat OpenShift console interface. On the left is a sidebar with navigation menus: Administrator, Home, Overview, Projects, Search, API Explorer, Events, Operators (with sub-items OperatorHub and Installed Operators), Workloads, Networking, Storage, Builds, Monitoring, Compute, User Management, and Administration (with sub-items Cluster Settings and Namespaces). The main content area is titled 'Project: fep-install-test' and 'FUJITSU Enterprise Postgres Operator > Create FEPCluster'. Below this is the 'Create FEPCluster' heading and a subtext: 'Create by manually entering YAML or JSON definitions, or by dragging and dropping a file into the editor.' There are two radio buttons for 'Configure via': 'Form view' and 'YAML view', with 'YAML view' selected and highlighted by a red box. The YAML editor shows a configuration for 'FEPCluster' with fields like 'apiVersion', 'kind', 'metadata', 'spec', 'customAnnotations', 'forceSsl', 'image', 'pullPolicy', 'instances', 'mcSpec', 'limits', 'requests', 'podAntiAffinity', 'podDisruptionBudget', 'servicePort', 'syncMode', 'sysExtraLogging', 'fepChildCrVal', and 'backup'. At the bottom of the editor are 'Create' and 'Cancel' buttons, and a 'Download' button on the right.

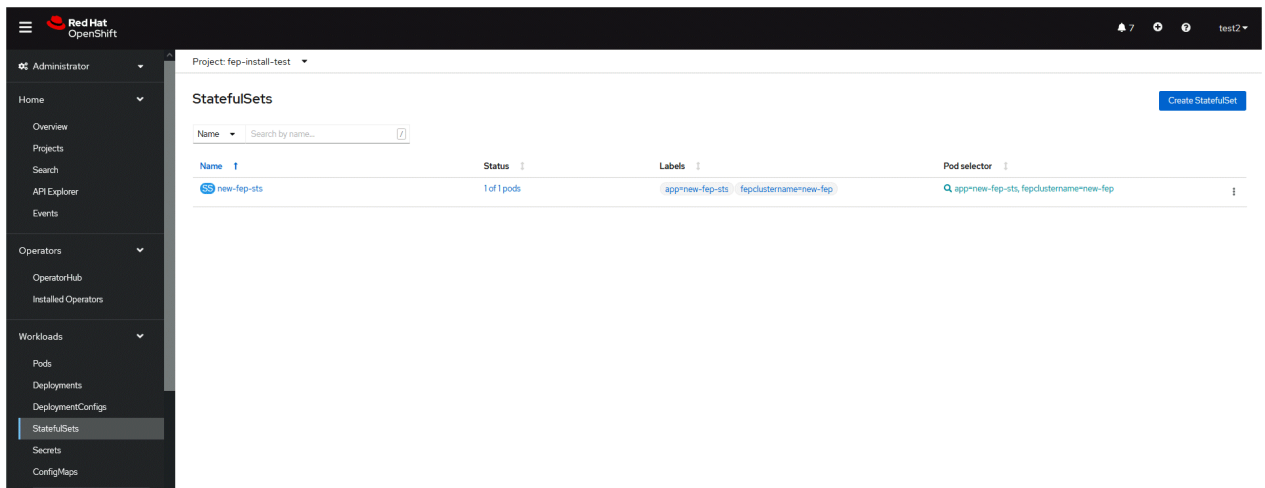
- When "Create" is clicked on either of the two pages above, the operator creates FEPCluster CR, and there after one by one FEPBackup, FEPConfig, FEPVolume, FEPUser, and FEPcert child CRs are created automatically. The starting values for child CRs are taken from the "fepChildCrVal" section of the FEPCluster CR YAML file. Modifying value in FEPCluster "fepChildCrVal" section. Operator reflects changes from FEPCluster parent CR to respective child CRs. Only allowable changes are reflected in child CRs. Child CRs are marked internal objects and hence will not be visible on the OCP console. However, you can check child CRs using command-line tools.

The screenshot shows the Red Hat OpenShift console interface. The sidebar is the same as in the previous screenshot. The main content area is titled 'Project: fep-install-test' and 'FUJITSU Enterprise Postgres Operator 4.1.0 provided by Fujitsu'. Below this is a navigation bar with tabs: Details, YAML, Subscription, Events, All instances, FEPCluster (selected), FEPAAction, FEPEXporter, FEPLogging, FEPPgpool2Cert, FEPPgpool2, FEPRestore, and FEPUUpgrade. The 'FEPClusters' section is displayed, showing a table with columns: Name, Kind, Status, Labels, and Last updated. The table contains one entry: 'FEPC new-fep' of kind 'FEPCluster' with status 'Condition Running' (highlighted by a red box), 'No labels', and '5 minutes ago'. A 'Create FEPCluster' button is visible in the top right corner of the table area.

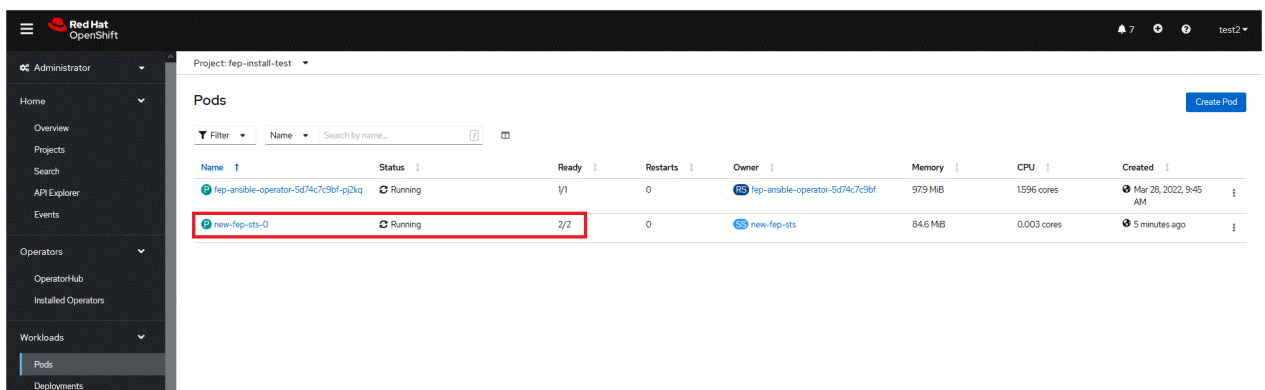
- In FEPCluster CR, annotations are added to indicate that child CRs are created successfully and has initialised properly. It may take some time to complete.



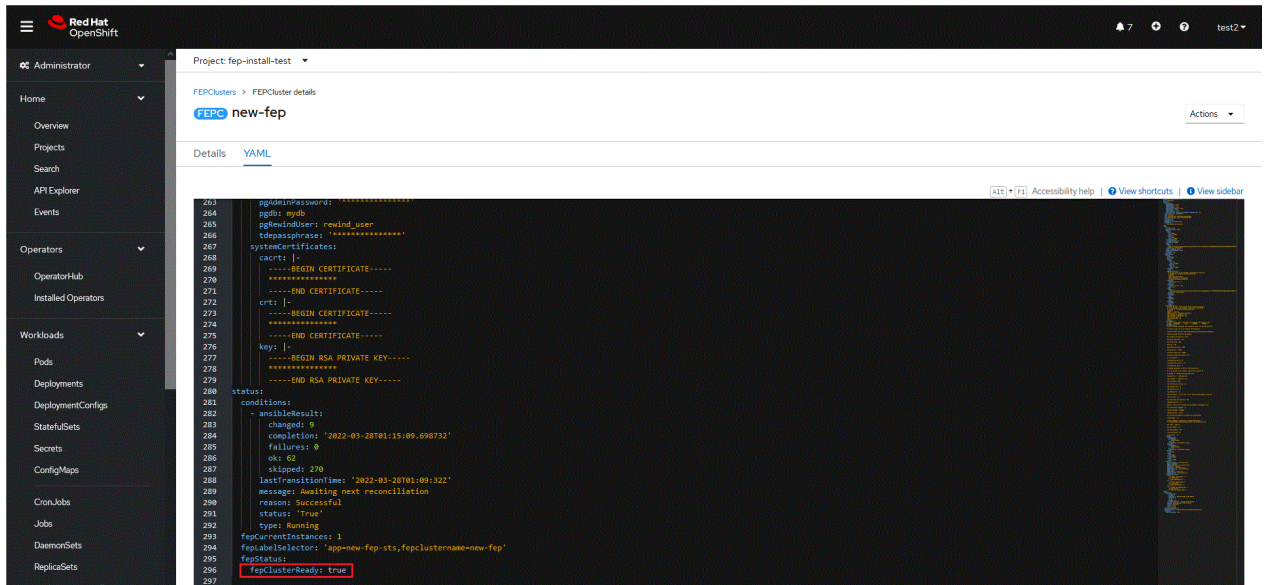
- Once child CRs are marked done in annotations, operator creates StatefulSet for the cluster.



- StatefulSet will start one FEP instance at one time and will wait it to be ready before starting next one.



- Once all instances of FEP servers are started, the operator marks a flag "fepClusterReady" under "status.fepStatus" section of CR to be **true**, indicating that FEPCluster is ready for use. Looking at YAML of FEPCluster CR, it would look like as below:



```
263 pgAdminPassword: "*****"
264 pgdb: mydb
265 pgAdminUser: pgAdmin_user
266 pgAdminPassphrase: "*****"
267 systemCertificates:
268   cacert: |
269     -----BEGIN CERTIFICATE-----
270     .....
271     -----END CERTIFICATE-----
272   crt: |
273     -----BEGIN CERTIFICATE-----
274     .....
275     -----END CERTIFICATE-----
276   key: |
277     -----BEGIN RSA PRIVATE KEY-----
278     .....
279     -----END RSA PRIVATE KEY-----
280
281 status:
282   conditions:
283     - availableResult:
284       changed: 9
285       completion: "2022-03-28T01:15:09.690732"
286       failures: 0
287       ok: 62
288       skipped: 270
289       lastTransitionTime: "2022-03-28T01:09:32Z"
290       message: "Awaiting next reconciliation"
291       reason: "Successful"
292       status: "True"
293       type: "Running"
294     fepCurrentInstances: 1
295     fepLabelSelector: "app=new-fep-sts,fepclustername=new-fep"
296     fepStatus:
297       fepClusterReady: true
```

- Operator also masks the sensitive fields like passwords, passphrase, certificates and keys in FEPCluster fepChildCrVal and also in respective child CRs.

4.2 Deploy a Highly Available FEPCluster

In a highly available FEP cluster, load balancing is possible by distributing read queries to replica instances.

In addition, if the master instance fails, the user can switch to the replica instance immediately to localize the business interruption period.

In a highly available configuration, you can select the synchronization mode for the replica instance. Synchronous replication is recommended for systems that cannot tolerate data loss in the event of a master instance failure.

Because multiple instances are created in a highly available configuration, licenses are required for each.

To deploy a highly available FEPCluster in given namespace, follow these steps:

[Prerequisites]

If the FEP cluster is running in HA mode, the backup and archive WAL volumes must be configured with shared storage (NFS, etc.) that supports ReadWriteMany. See the Openshift documentation for instructions on setting up shared storage. Also, the reference procedure is described in "[Appendix C Utilize Shared Storage](#)", so please check if necessary.

If you do not have shared storage, you can remove the backup section and the backup and archive volume sections to disable the backup feature and deploy the FEP cluster.



If you are deploying on a Kubernetes cluster, Refer to "Custom Resource Parameters" in the Reference to create and apply a yaml file.

- It is the same as the procedure from step 1 to step 3 in "[4.1 Deploying FEPCluster using Operator](#)".

- Instead of step 4 in "4.1 Deploying FEPCluster using Operator", change to the YAML view and specify '3' for the "instances" parameter of "fep" in "spec". Specify the storage class for the prepared shared storage for the backup and archive WAL volumes.

Project: fep-install-test

FUJITSU Enterprise Postgres Operator > Create FEPCluster

Create FEPCluster

Create by manually entering YAML or JSON definitions, or by dragging and dropping a file into the editor.

Configure via: ☐ Form view ☒ YAML view

```

1  apiVersion: fep.fujitsu.io/v2
2  kind: FEPCluster
3  metadata:
4    name: new-fep
5    namespace: fep-install-test
6  spec:
7    fep:
8      customAnnotations:
9        allDeployments: {}
10     forceSsl: true
11     image:
12       pullPolicy: IfNotPresent
13     instances: 3
14     mcSpec:
15       limits:
16         cpu: 500m
17         memory: 700Mi
18       requests:
19         cpu: 200m
20         memory: 512Mi
21     podAntiAffinity: false
22     podDisruptionBudget: false
23     servicePort: 27500

```

- It is the same as the procedure from step 5 to step 10 in "4.1 Deploying FEPCluster using Operator".
- Three pods deployed and ready for a highly available FEPCluster.

Project: fep-install-test

Pods

Name	Status	Ready	Restarts	Owner	Memory	CPU	Created
fep-ansible-operator-5d74c7c9bf-p2kq	Running	1/1	0	fep-ansible-operator-5d74c7c9bf	349.7 MB	2.267 cores	Mar 28, 2022, 9:45 AM
new-fep-sts-0	Running	2/2	0	new-fep-sts	169.9 MB	0.024 cores	5 minutes ago
new-fep-sts-1	Running	2/2	0	new-fep-sts	63.0 MB	0.004 cores	4 minutes ago
new-fep-sts-2	Running	2/2	0	new-fep-sts	63.3 MB	0.002 cores	2 minutes ago

Information

You can determine whether the master or replica pod is the master or replica pod by issuing the following command:

```
$ 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

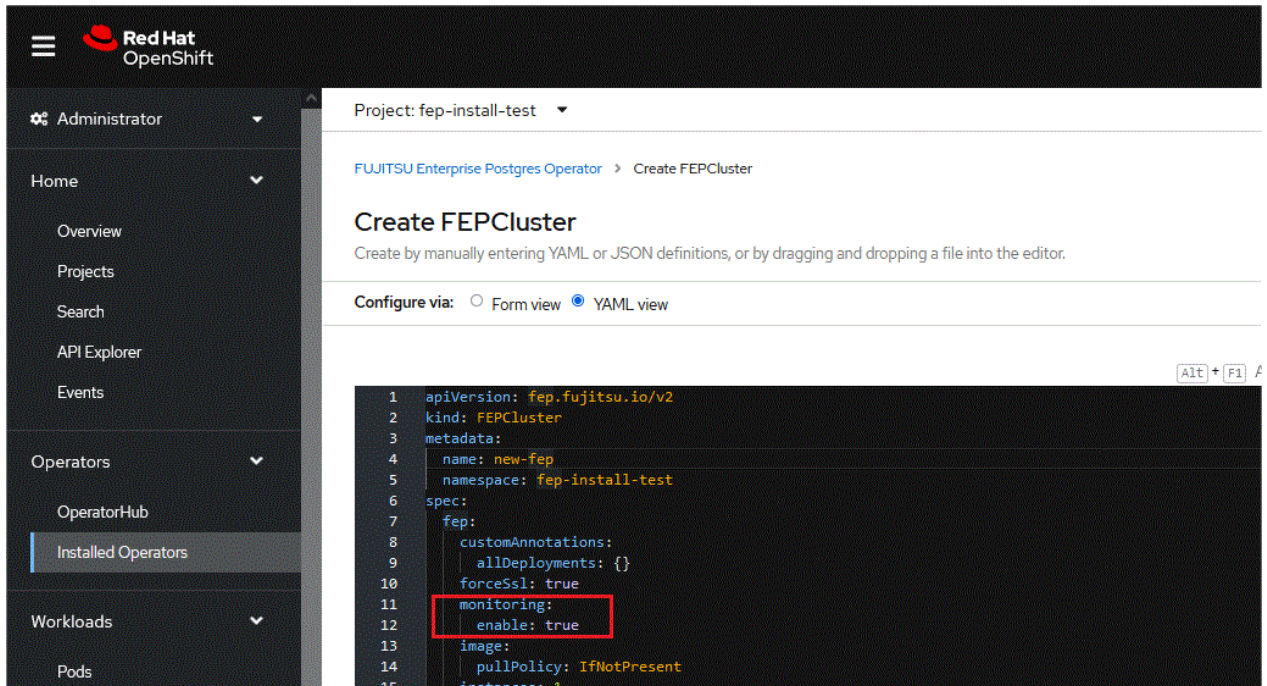
4.3 Deploying FEPEXporter

To deploy a FEPEXporter, follow these steps:

Note

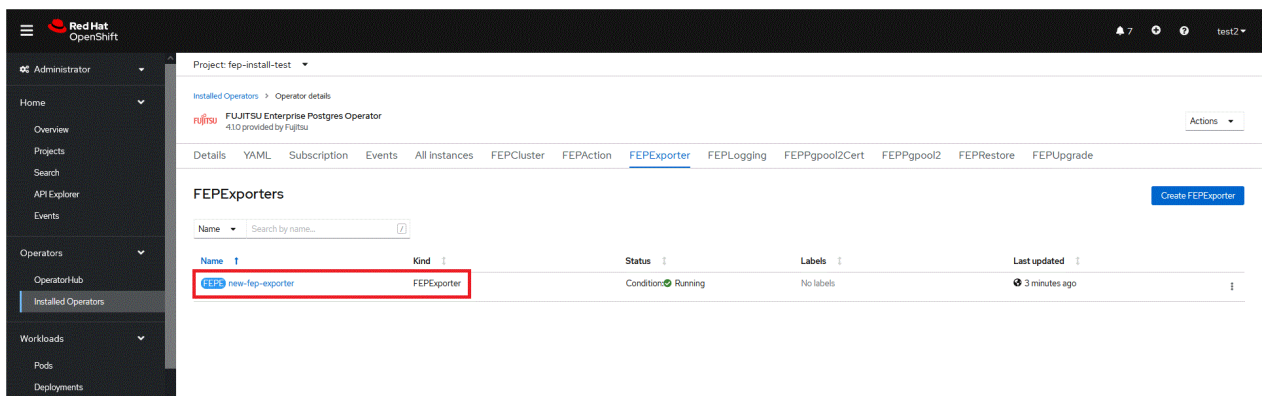
If you are deploying on a Kubernetes cluster, Refer to "Custom Resource Parameters" in the Reference to create and apply a yaml file.

1. In order to deploy FEPEXporter managed by Operator, it is as easy as setting `fep.monitoring.enable` to true in FEPCluster CR at the time of deployment.



```
1  apiVersion: fep.fujitsu.io/v2
2  kind: FEPCluster
3  metadata:
4    name: new-fep
5    namespace: fep-install-test
6  spec:
7    fep:
8      customAnnotations:
9        allDeployments: {}
10     forceSsl: true
11     monitoring:
12       enable: true
13     image:
14       pullPolicy: IfNotPresent
15     instances: 1
```

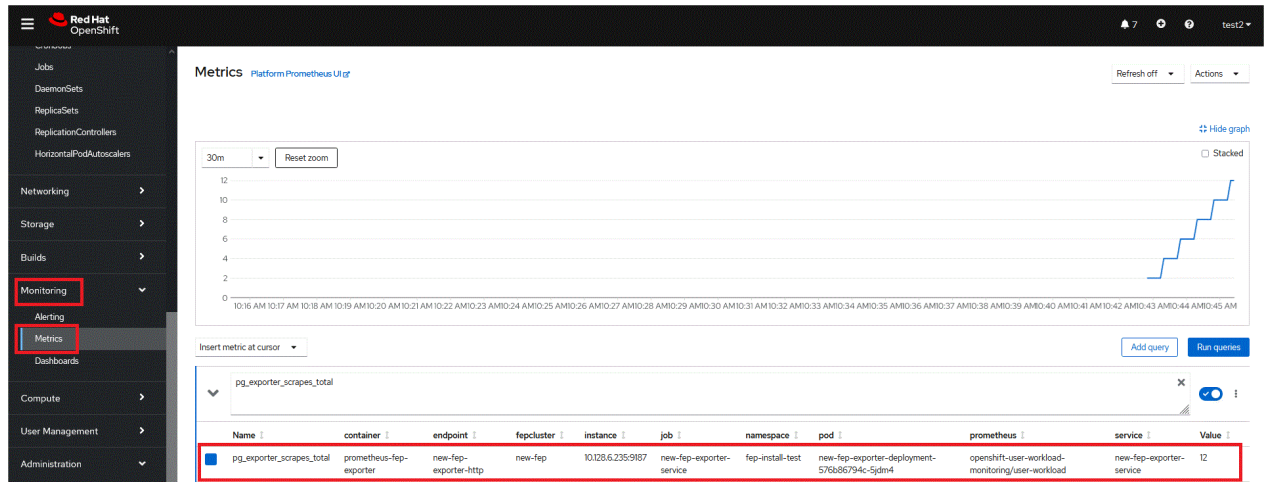
2. FEPEXporter will be created automatically under the name `<cluster-name>-fepexporter`. And it will list show all the database with statistics of specified FEPCluster.



Name	Kind	Status	Labels	Last updated
new-fep-exporter	FEPEXporter	Condition Running	No labels	3 minutes ago

3. FEPEXporter spawned by FEP Operator in aforementioned way will scrape metrics by default from the Master and standby instances and make it available to Prometheus.
4. User can configure MTLS to be used for HTTP endpoint used by Prometheus for metrics scraping as well as connection from FEP Exporter to database.
 - a. If `pgMetricsUser`, `pgMetricsPassword` and `pgMetricsUserTls` is defined in FEPCluster; FEPEXporter will hence use these for securing connection to the postgres instances. In absence of these parameters, FEPEXporter will use `pgAdminUser` (i.e. super user).
 - b. User can configure `Prometheus.tls` and `FEPEXporter.tls` to ensure that metrics end point (`/metrics`) by FEPEXporter is also used with MTLS (Refer to "FEPEXporter Custom Resource" in the Reference for details of fields)

5. User can also configure basic authentication by specifying a secret that contains username & password. (Refer to "FEPEXporter Custom Resource" in the Reference for details of fields)
6. Now user can see scrape FEPEXporter specific metrics on Openshift Platform in monitoring section area using PROMQL to specify a metrics of interest



Note

- User can set `fep.monitoring.enable` to true or false on an already instantiated cluster as well to achieve desired results
- `pgMetricsUser` can be defined later on a running FEPCluster with monitoring enabled and can force FEPEXporter to use `pgMetricsUser` by mere restarting it (refer `restartRequired`). However, MTLs can not be configured in this case and user is expected to grant specific permission to `pgMetricsUser` for all the database objects which are expected to be use while scraping information.
- For MTLs to be forced, ensure `usePodName` and `pg_hba.conf` is been set appropriately.
- FEPEXporter default metrics expects few following in `postgresql.conf`
 - `pg_stats_statements` library pre-loaded
 - `track_activities` and `track_counts` are turned on
 - Monitoring user needs permission on `pg_stat_*` views
- FEPEXporter pod specification related to CPU memory can be changed. After changing resources specification, set `restartRequired` flag to true. FEPEXporter will be restarted with new specifications
- FEP Monitoring is closely integrated with Prometheus available on platform. User should ensure that on openshift platform monitoring is enabled for user-defined projects (Refer: <https://docs.openshift.com/container-platform/4.6/monitoring/enabling-monitoring-for-user-defined-projects.html>). For platforms other than openshift, ensure Prometheus is installed before deployment of FEP operator

4.4 FEPEXporter in Standalone Mode

FEPEXporter is an independent CR; hence it does not necessarily depend on main FEPCluster CR. To deploy a FEPEXporter in given namespace follow the below step.

Note

If you are deploying on a Kubernetes cluster, Refer to "Custom Resource Parameters" in the Reference to create and apply a yaml file.

1. To create FEPEXporter CR, either
 - (1) Click on "**Create Instance**" under FEPEXporter.

OR

(2) Click on "**FEPExporter**" on top and then click on "**Create FEPExporter**" on the next page.

2. In Form View, one can change only the name of cluster being deployed. The default name is "new-fep-exporter". This name must be unique within a namespace.
3. FEPExporter scrapes metrics for FEPCluster within same namespace.

The screenshot shows the Red Hat OpenShift console interface. On the left is a sidebar with navigation menus: Administrator, Home, Overview, Projects, Search, API Explorer, Events, Operators (OperatorHub, Installed Operators), Workloads (Pods, Deployments). The main panel is titled 'Project: fep-install-test' and 'FUJITSU Enterprise Postgres Operator > Create FEPExporter'. Below this is the 'Create FEPExporter' section with a note: 'Create by completing the form. Default values may be provided by the Operator authors.' There are two tabs: 'Form view' (selected) and 'YAML view'. A note states: 'Note: Some fields may not be represented in this form. Please select "YAML View" for full control of object creation.' The 'Name' field is labeled 'Name *' and contains the value 'new-fep-exporter'. Below it is a 'Labels' field with the value 'app=frontend'. At the bottom are 'Create' and 'Cancel' buttons.

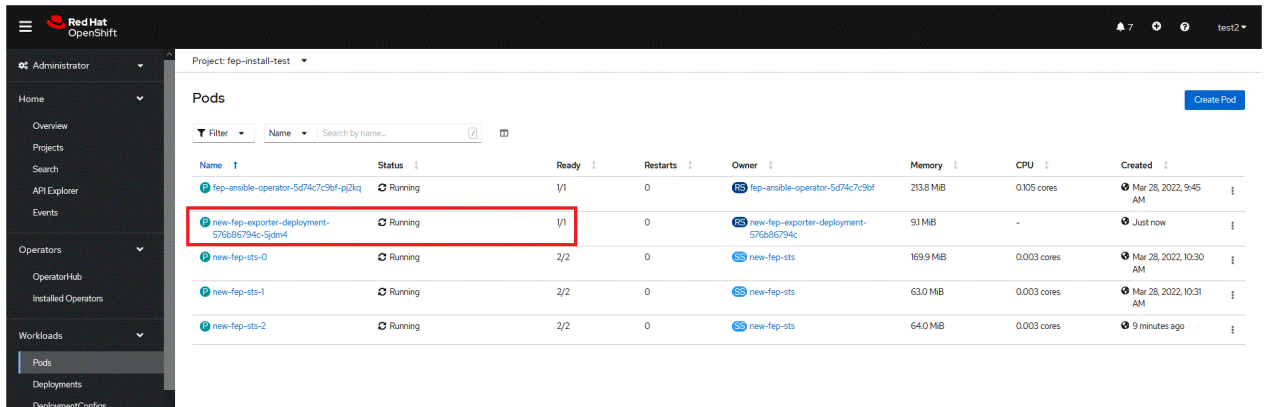
4. In YAML View, starting value of FEPExporter CR is visible and one can choose to modify parameters before creating CR. Refer to the Reference for details of parameters.

The screenshot shows the Red Hat OpenShift console interface in the 'YAML view' for 'Create FEPExporter'. The sidebar is the same as the previous screenshot. The main panel shows the 'YAML view' tab selected. Below the 'Create FEPExporter' section, there is a code editor displaying the following YAML configuration:

```
1 apiVersion: fep.fujitsu.io/v1
2 kind: FEPExporter
3 metadata:
4   name: new-fep-exporter
5   namespace: fep-install-test
6 spec:
7   fepExporter:
8     exporterLogLevel: error
9     fepClusterList:
10     - new-fep
11   image:
12     pullPolicy: IfNotPresent
13   mcSpec:
14     limits:
15       cpu: 500m
16       memory: 700Mi
17     requests:
18       cpu: 200m
19       memory: 512Mi
20   restartRequired: false
21   sysExtraLogging: false
22   userCustomQueries: |-
23     usr_example:
24       query: "SELECT EXTRACT(EPOCH FROM (now() - pg_last_xact_replay_timestamp())) as lag"
25       master: true
26       metrics:
27         - lag:
28           usage: "GAUGE"
29           description: "Replication lag behind master in seconds"
30
```

At the bottom of the form are 'Create' and 'Cancel' buttons, and a 'Download' button on the right.

- When clicked on the "Create" button. It will create FEPEXporter pod with other resource like secret, service, configmap for data source queries.



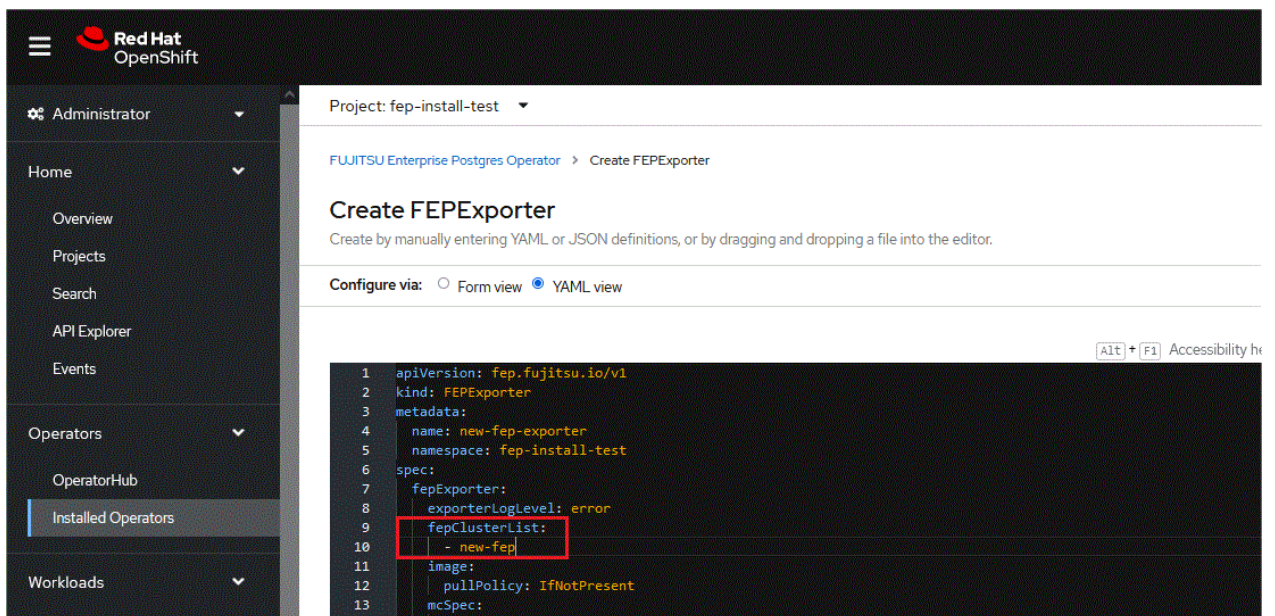
Project: fep-install-test

Pods

Filter Name Search by name...

Name	Status	Ready	Restarts	Owner	Memory	CPU	Created
fep-ansible-operator-5d74c7c9b4-g2kq	Running	1/1	0	fep-ansible-operator-5d74c7c9b4	213.8 MB	0.105 cores	Mar 28, 2022, 9:45 AM
new-fep-exporter-deployment-576b86794c-5gdm4	Running	1/1	0	new-fep-exporter-deployment-576b86794c	91 MB	-	Just now
new-fep-sts-0	Running	2/2	0	new-fep-sts	169.9 MB	0.003 cores	Mar 28, 2022, 10:30 AM
new-fep-sts-1	Running	2/2	0	new-fep-sts	63.0 MB	0.003 cores	Mar 28, 2022, 10:31 AM
new-fep-sts-2	Running	2/2	0	new-fep-sts	64.0 MB	0.003 cores	9 minutes ago

- Targeting the name of FEPCluster in FEPEXporter cluster list. Before targeting cluster, Check the FEPCluster status and FEP StatefulSet are in running condition.



Project: fep-install-test

FUJITSU Enterprise Postgres Operator > Create FEPEXporter

Create FEPEXporter

Create by manually entering YAML or JSON definitions, or by dragging and dropping a file into the editor.

Configure via: ☐ Form view ☒ YAML view

```

1  apiVersion: fep.fujitsu.io/v1
2  kind: FEPEXporter
3  metadata:
4    name: new-fep-exporter
5    namespace: fep-install-test
6  spec:
7    fepExporter:
8      exporterLogLevel: error
9    fepClusterList:
10     - new-fep
11    image:
12      pullPolicy: IfNotPresent
13    mcSpec:

```

- It will recreate FEPEXporter pod with a new dataresource secret. It will list down all the database with statistics of specified FEPCluster in monitoring section.
- If fepClusterList has more than one clusters listed, current exporter will collect metrics for all of those listed.
- Multiple FEPEXporters can be deployed within one namespace with their own cluster list to collect metrics from.

4.5 Configuration FEP to Perform MTLS

All three traffic can be secured by using TLS connection protected by certificates:

- Postgres traffic from Client Application to FEPCluster
- Patroni RESTAPI within FEPCluster
- Postgres traffic within FEPCluster (e.g. replication, rewind)

Here, we provide two methods to create certificates for securing the TLS connection and provide mutual authentication. The first method is to create and renew certificate manually. The second method is to use CertManager to create an automatically renew certificate.



Note

The following considerations apply to client connections to a database cluster in an MTLS configuration:.

- Distribute the Root certificate for server (validation) that you specified when you created the MTLS database cluster to the client machines.
- Create and use a new client certificate.
- If the server root certificate and the client root certificate are different, a server-side configuration update is required.

4.5.1 Manual Certificate Management

Overview of Procedures

The procedures to enable MTLS communication are listed below:

1. Create a self signed certificate as CA
2. Create Configmap to store CA certificate
3. Create a password for protecting FEP Server private key (optional)
4. Create FEP Server private key
5. Create FEP Server certificate signing request
6. Create FEP Server certificate signed by CA
7. Create TLS Secret to store FEP Server certificate and key
8. Create private key for Patroni
9. Create certificate signing request for Patroni
10. Create certificate signed by CA for Patroni
11. Create TLS secret to store Patroni certificate and key
12. Create private key for "postgres" user client certificate
13. Create certificate signing request for "postgres" user client certificate
14. Create client certificate for "postgres" user
15. Create TLS secret to store "postgres" certificate and key
16. Repeat step 12-15 for "repluser" and "rewinduser"



Note

- The information in the manual is only an example, and in operation, use a certificate signed by a certificate authority (CA) that the user can trust.
- When working on a Kubernetes cluster, replace the oc command with the kubectl command.

Creating a CA Certificate

1. Create a self signed certificate as CA

```
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. Create Configmap to store CA certificate

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

3. Create a password for protecting FEP Server private key (optional)

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

Creating a Server Certificate

4. Create FEP Server private key

```

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. Create FEP Server certificate signing request

```

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
```



The cluster name and namespace must be changed appropriately.

If you are connecting from outside the OCP cluster, you must also include the host name used for that connection.

6. Create FEP Server certificate signed by CA

```
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. Create TLS Secret to store FEP Server certificate and key

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

8. Create private key for Patroni

At the moment, FEP container does not support password protected private key for Patroni.

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

9. Create certificate signing request for Patroni

```
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
```



The cluster name and namespace must be changed appropriately.

If you are connecting from outside the OCP cluster, you must also include the host name used for that connection.

10. Create certificate signed by CA for Patroni

```
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. Create TLS secret to store Patroni certificate and key

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

Creating a User Certificate

12. Create private key for "postgres" user client certificate

At the moment, SQL client inside FEP server container does not support password protected certificate.

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

13. Create certificate signing request for "postgres" user client certificate

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

14. Create client certificate for "postgres" user

```
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. Create TLS secret to store "postgres" certificate and key

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

16. Repeat step 12-15 for "repluser" and "rewinduser"

4.5.2 Automatic Certificate Management

There are many Certificate Management tools available in the public. In this example, we will use cert-manager for the purpose.



Note

- Note that certificates created in this example are not password protected.
- When working on a Kubernetes cluster, replace the oc command with the kubectl command.

Install cert-manager

```
oc create namespace cert-manager

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

Create a Self Signed Issuer (This can be namespace specific or cluster wise)

This example creates an Issuer, that can create self signed certificate, in namespace my-namespace.

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

Create a Self Signed CA certificate using selfsigned-issuer

```
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
```

The above command will create a self signed Root certificate and private key stored in the Kubernetes secret "cacert" in namespace my-namespace.

Create a CA Issuer with above certificate

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



```
namespace: my-namespace
spec:
  ca:
    secretName: cacert
EOF
```

Create FEP Server certificate using above CA Issuer

Assuming FEPCluster name is mydb in namespace my-namespace.

```
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
```

Create Patroni certificate using above CA Issuer

Assuming FEPCluster name is mydb in namespace my-namespace.

```
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
```

Create postgres user client certificate

```
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
```

Create repluser user client certificate

```
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
```

Create FELogging(Fluentd) server certificate using above CA Issuer

Assuming FELogging name is **nfl** in namespace **feplogging-dev**.

```
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

```

Create FEPLogging client(prometheus) certificate

```

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

```

Create FEPLogging client(fluentbit) certificate

```

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

```

Create FEPExporter certificate using above CA Issuer

Assuming FEP Exporter name is **exp1** in namespace **my-namespace**.

```

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:
  - 'expl-service'
  - 'expl-service.fepexporter-dev'
  - 'expl-service.fepexporter-dev.svc'
  - 'expl-service.fepexporter-dev.svc.cluster.local'
  duration: 8760h
  usages:
  - server auth
  secretName: fepexporter-cert
  issuerRef:
    name: ca-issuer
EOF

```

Create FEPEXporter user client(prometheus) certificate

```

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

```

4.5.3 Deploy FEPCluster with MTLS support

Deploy FEPCluster with manual certificate management

Use the following yaml as an example to deploy a FEPCluster with Manual Certificate Management. MTLS related parameters are highlighted in **Red**.

```

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-14-server:ubi8-14-0.0'
        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-----
    MIIIEowIBAAKCAQEAO0DFkImha8CIJiVcwXbBP1L+/DmS9/ipRhQQHxf05x7jSONse
    IHdFd6+Qx2GX8KAiAhVykf6kfacwBYTATU1xDgwWTm82KVRPh+kZDIj2wPcJr14m
    mTP6I6a2mavUgDhezHc9F8/dchYj3cw8lX0kU6xamqrKQYlxQH48NkI0qcwh06sK
    AHF4eWfCr8Ot44xADIA1JcU2CS1RKSZEtURZ+30Py+j907Enjp1YR33ZKUHW30pU
    9dpIneyfXBN/pT6cX3MetYwtgmpV/pHqY8pbxqGfoYRhgQDsSRC14dtlecaZeZ4j
    uTOotcPkZELHP6eu8gaLtycG9lpbAMQ15w0r8QIDAQABaoIBACq213qPuoimExrQ
    fqXaNJmqNYK4fJqXCB6oUwf0Flu4ubkx5V532hLSPHwLs+a0lAWlbNozSoBVOu8G
    64VvrA9bv3/cJVqZZ6/UzUthHPU+Ogh24qhwF5QU8kXZEU11To3YsPoftalgjX9G
    Ff0fLcLVC8nL3K9RiaDXxXbEYpWrYu39M3FCpAXAzV2PrNxsP9PKyNWHnBPc08z5
    tFj45/bHn+j31AVVvgWtqz0pLks57hc4Q7yW/2RoRYq2md1KI709OLNwtkWEOVqb
    qnraorh2TWgnNaOB5oX5/lJvKtlq778fw96jGqykBr0+DKozj9rlr1OGgYOKDwld
    nsZJPAECgYEA+Oqf/fxtPdsNGiaL2Z/heewvtaxjw/WoEVBFEcb6/y4Ro7aux9nB
    16FcVi79CwfpoUTJ7cnZvYSmBk5GWEObEIAeo61lvM/QeltM5+usAPd5/TcHXLye
    92OnXmq7h3F4UXEkMayak8Lpu/TdmR5uOaL+m4aEu+XMY5tlxqDCnyECgYEA1h4X
    jCPi7Ja5CHK7a2Ud4TL2DNpIBE6GSK9iQ+0xFL6TsiK2Sfu6n8mx2sh+Jm0KHTie

```

```

/gWHdHQZSSWiuULfHoYEq3Rq8S6Av3GsGtRSp003j7BE8C20Vpt0FnNTjZmdzf2/
YZxc5KuYlH9qeY7Y7ceOsWA8JckDgMHPYzyLaTECgYBALD0TPgDr8Y1vMIDdmlqH
FF04eTk/TBYIYKltgJ81KqthibeFzp4q+W7UyUhzj5a4XQOySlfYhFpJReTc3JEd
r+o2SH3ymuEkqmUpZZjyptrMbWN4g3t4TDjaHqo6QQbD+GdcZyNy9M1Np9N5p17E
fUEm14dg6d3H0Ehs7QVAAQKBgQDRUx3mLxc9oKRINBiYDerGLJILQqLBQxtY181T
ZuFizGWL8w+PCIAMkpxDrVpWqqcGpiiuRi2ElbPapOaOg2epaY/LJscd/j5z6uc8
W3JoNljpKoRa4fO578Pv5tM6TYHOzLF5Veoiy/a8sI3hRNuiqkM/+TsUHY5FJDRh
aeDk4QKBgCOHievVR+MWuwakzD6lNCbb8H6fvZ3WRAT8BYZ3wW9YfnV4J4uh/B1
moWYgIK2UpkrhA8scMUC790FoybQeParQ35x7Jl9lbmTKkCqsX63fyqqYhx3SXRl
JSktmH4E2cGmosZisjB7COKHR32w0J5JCgaGInQxjldbGrwhZQpn
-----END RSA PRIVATE KEY-----

crt: |-
-----BEGIN CERTIFICATE-----
MIID2CCAsCgAwIBAgIQdFFYteD4kZj4Sko2iy1IJTANBgkqhkiG9w0BAQsFADBX
MRgwFgYDVQQKEw9NeSBPcmdbhml6YXRpb24xCzAJBgNVBAsTAkNBMS4wLAYDVQQD
EyVNeSBPcmdbhml6YXRpb24gQ2VydGlmawNhdGUgQXV0aG9yaXR5MB4XDTIxMDQy
MDAwMDQ1OV0XDTIxMDQyMDAxMDQ1OVowGDEWMBQGA1UEAwNKi5jaGctcHRjLnBv
ZDCCASIdQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBANAXZCJoWvAiCYlXMF2w
T5S/vw5kvf4qUYUEB8Xzuce40jp7HiB3RXevkMdh1/CgIgIVcpH+pH2nMAWEwE1N
cQ4MFk5vNilUT4fpGQyI9sD3Ca9eJpkz+iOmtpmr1IA4Xsx3PrfP3XIWI93MPNV9
JFOsWpqqykJGcUB+PDZCNKnMITurCgBxeHlnwq/DreOMQAYANSXFNgtUSkmRLVE
Wft9D8vo/dOxJ46dWEd92SlB8N9KVPXaSJ3snlwTf6U+nF9zHrWMLYJqVf6R6mPK
W8ahn6MkYEEA7EkQpEhbZxNxmEi7kzqLXD5GRCxz+nrvIGi7cnBvZaWwDEJecN
K/ECaWEAAaOB3jCB2zATBgNVHSUEDDAKBggrBgEFBQcDATAMBgNVHRMBAf8EAjAA
MIGlBgNVHREEga0wgaqCCWxvY2FsaG9zdIIbKi5jaGctcHRjLnBvZC5jbHVzdGVy
LmxvY2FsgMqLm15ZGItaGVhZGxl3Mtc3ZjghsqLm15ZGItaGVhZGxl3Mtc3Zj
LmNoZylwdGOCHyoubXlkYiIoZWFKbGVzcy1zdmMuY2hnLXB0Yy5zdmOCLSouBx1k
YiIoZWFKbGVzcy1zdmMuY2hnLXB0Yy5zdmMuY2xlc3Rlcis5b2NhbdANBgkqhkiG
9w0BAQsFAAOCAQEALnhliDflu+BHp5conq4dXBwD/Ti2YR5TWQixM/0a6OD4KecZ
MmaLl0T+OJJvA/j2IufZpc7dzEx5mZDKR2CRmoq10qZXqCRTrBZSXm6ARQWoYpeg
9c0l4f8roxrkMGUKVPTKUwAvbnNYhD2l6PlBPwMpkMUFqFaSEXMaPyQKhrtQxdpH
WjuS540P0lmoPeYu/yiaD98LtrTXnb6jch84SKf6Vii4HAVQyMeJaW+dpkqcI2+V
Q4fkWYSJy8BNcmXCwvHDLdy+s4EXWvHafhusuUhcp4HyMblA6hd5hJhgFSnEvLy
kLA0L9LaScxee6V756Vt9TN1NGjwmwyQDOhnQQ==
-----END CERTIFICATE-----

cacrt: |-
-----BEGIN CERTIFICATE-----
MIIDXCcAKSgAwIBAgIRAMPzF3BNFXT9HWE+NXlFQjQwDQYJKoZIhvcNAQELBQAw
VzEYMBYGA1UEChMPTXkgT3JnYW5pemF0aW9uMQswCQYDVQQLEwJDTQTEuMCA1UE
AxMlTXkgT3JnYW5pemF0aW9uIENlcnRzZmljYXRlIEF1dGhvcml0eTAeFw0yMTA0
MTkwNDQ0MjNaFw0yMTA0MTcwNDQ0MjNaMFcxGDAWBgNVBAoTD015IE9yZ2FuaXph
dGlvb3JELMAkGA1UEC3MCQ0ExLjAsBgNVBAMTJU15IE9yZ2FuaXphdGlvbiBDZXJ0
aWZpY2F0ZSB8dXRob3JpdHkwgGEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIB
AQc5t6CS23G1k65YMw5e4i4xH1dyxkCZS67w/6LWqeIlYKmfAae183Wwy8MHUpOb
4mahtUafEzDEOX6+URf72J8m0voldQ5FYrlAyUOyX8U90wGFqhbEgKRqt7vZEwIe
2961fwqHh6917zi4xmt5W6ZJ5dBQVtkhzB+Pf7O6KBYjHoCnBBkfNVzsfZQ/1hnR
0UzimfAc7Ze+UNwhXJhinFRJ3YuR+xiOTpPk1lGXPhLgFSQheKz4KepcbQEQKebj
jg0dumloBYIXZTSSbi09rNmFUVLB5DcV0vZbSrGxLjWLBt5U8N2xf2d1bvkQW+bw
Kklf9OG26bAi27tujurzn3r3AgMBAAGjIzAhMA4GA1UdDWEB/wQEAwICpDAPBgNV
HRMBAf8EBTADAQH/MA0GCSqGSIb3DQEBCwUAA4IBAQAAM0CN3n5C/KOT4uZ4ewwKK
rHmANBPVM9u6MJB08U62HcqLeoCuDFeU8zmUjLHjsQaPX64mJZlR7T5y52gEKO5A
0qsBz3pg/vJ5DJTtv0698+1Q1hB9k3smQdksAim19FZqysB7J4zK/+8aJ/q2kIFvs
Jk3ekwQdQ3xfggklBQVuf76gr1v0uYlPtPfPflfcGZ06Im6mqbajenXoR1PxPB0
+zyCS8DkgPtDulplrUwvXCFMYw9TPbzXKlt7t1sqRXogYLnXWJdZMlnOYCNd+rDm
qxenV9Ir8RqZ0XSyuYzRka5N4dhIhrzTAiNdeU5gzynXOz67u/Iefz1iK9ZcdE3
-----END CERTIFICATE-----

```

Deploy FEPCluster with automatic certificate management

Use the following yaml as an example to deploy a FEPCluster with Automatic Certificate Management. MTLS related parameters are highlighted in **Red**.

```

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-14-server:ubi8-14-0.0'
    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-----
MIIEowIBAAKCAQEADfKImha8CIJiVcwXbBPlL+/DmS9/ipRhQQHxf05x7jSONse
IHdFd6+Qx2GX8KAIaHvYkf6kfacwBYTATUlxDgwWTm82KVRPh+kZDIj2wPcJr14m
mTP6I6a2mavUgDhezHc9F8/dchYj3cw8lX0kU6xamqrKQYlXQH48NkI0qcwh06sK
AHF4eWfCr8Ot44xADIA1JcU2CS1RKSZEtURZ+30Py+j907Enjp1YR33ZKUHW30pU
9dpIneyfXBN/pT6cX3MetYwtgmpV/pHqY8pbxqGfoYRhGQDsSRC14dtlecaZeZ4j
uT0otcPkZELHP6eu8gaLtycG9lpbAMQ15w0r8QIDAQABaoIBACq213qPuoimExrQ
fqXaNJmqNYK4fJqXCB6oUwf0Flu4ubkx5V532hLSPHwLs+a0lAWlbNozSoBVOu8G
64VvrA9bv3/cJVqZZ6/UzUTbHPU+Ogh24qhwF5QU8kXZEUI1To3YsPofTalgjX9G
Ff0fLcLVC8nL3K9RiaDXxXbEYpWrYu39M3FCpAXAZV2PrNxsP9PKyNWHnBpc08z5
tFj45/bHn+j3lAVVvgWtqz0pLks57hc4Q7yW/2RoRYq2md1KI709OLNwtkWEOvqb
qnraorh2TWgnNaOB5OX5/lJvKtlq778fw96jGqykBr0+DKozj9rlr1OGgYOKDwLD
nsZJPAECgYEA+Oqf/fxtPdsNGiaL2Z/heewvtaxjw/WoEVBFCb6/y4Ro7aux9nB
16FcVi79CwfP0UTJ7cnZvYSmBk5GWEObEIAeo6llvm/QeltM5+usAPd5/TcHXLye
92OnXmq7h3F4UXEkMayak8Lpu/TdmR5uOaL+m4aEu+XMY5tlxqDCnyECgYEA1h4X
jCpi7Ja5CHK7a2Ud4TL2DNpIBE6GSK9iQ+0xFL6TsiK2Sfu6n8mx2sh+Jm0KHTiE
/gWHdHQZSSwiuULfHoYeq3Rq8S6Av3GsGtRSp003j7BE8C20Vpt0FnNTjZmdzf2/
YZxc5KuYlH9qeY7Y7ceOsWA8JckDgMHPYzyLaTECgYBALD0TPgDr8YlvMIDdmlqH
FF04eTk/TBYIYKltgJ8lKqthibeFzp4q+W7UyUhzj5a4XQOySlfYhFpJReTc3JEd
r+o2SH3ymuEkqmUpZZjyptrMbWN4g3t4TDjaHqo6QQbD+GdcZyNy9M1Np9N5p17E
fUEml4dg6d3H0Ehs7QVAAQKBgQDRUx3mLXc9oKRINBIyDerGLJILQqLBQxtYl81T
ZuFizGWL8w+PCIAMkpxDrVpWqqcGpiiuRi2ElbPapOaOg2epaY/LJscd/j5z6uc8
W3JoNljPkoRa4f0578Pv5tM6TYHOzlF5Veoiy/a8si3hRNuiqkM/+TsUHY5FJDRh
aeDk4QKBgCOHievvr+MWuwakzD6lNCbb8H6fvZ3WRAT8BYyz3wW9YfnV4J4uh/B1
moWYgIK2UpkrhA8scMUC790FoybQeParQ35x7Jl9lBmTKkCqsX63fyqqYhx3SXR1
JSktmH4E2cGmosZisjB7COKHR32w0J5JCgaGInQxjldBGrwhZQpn
-----END RSA PRIVATE KEY-----
```

```
crt: |-
-----BEGIN CERTIFICATE-----
MIID2CCAsCgAwIBAgIQDfFYteD4kZj4Sko2iy1IJTANBgkqhkiG9w0BAQsFADBx
MRgwFgYDVQQKEw9NeSBPcmdhbm16YXRpb24xZCZAJBgNVBAsTAkNBMS4wLAYDVQQD
EyVNeSBPcmdhbm16YXRpb24gQ2VydGlmawNhdGUgQXV0aG9yaXR5MB4XDTEyMDQy
MDAwMDQ1OV0xMDQ1MDQ1MDQ1OVowGDEWMBQGA1UEAwwNKi5jaGctcHRjLnBv
ZDCCASIdQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBANaxZCJoWvAiCYlXMF2w
T5S/vw5kvf4qUYUEB8Xzuce40jp7HiB3RXevkMdh1/CgIgIVcpH+pH2nMAWEwE1N
cQ4MFk5VnIlUT4fpgGyI9sD3Ca9eJpkz+iOmtpmr1IA4Xsx3PRfP3XIWI93MPNV9
JFOsWpqqyGJcUB+PDZCNKnMITurCgBxeHlnwq/DreOMQAYANSXFNgtUSkmRLVE
Wft9D8vo/dOxJ46dWED92SLB8N9KVPXaSJ3snlwTf6U+nF9zHrWMLYJqVf6R6mPK
W8ahn6MkYEEA7EkQpeHbZXnGmXmeI7kzqLXD5GRCxz+nrvIGi7cnBvZaWwDEJecN
K/ECaWEAAOB3jCB2zATBgNVHSUEDDAKBggrBgEFBQcDATAMBGNVHRMBAf8EAjAA
MIGlBgNHRREEga0wgaqCCWxvY2FsaG9zdIIbKi5jaGctcHRjLnBvZC5jbHVzdGVy
LmxvY2FsgHMQlml5ZGItaGVhZGxlc3Mtc3ZjghsqLml5ZGItaGVhZGxlc3Mtc3Zj
LmNoZyldGOCHyoubXlkYiIoZWFKbGVzcy1zdmMuY2hnLXB0Yy5zdmOCLSoubXlk
YiIoZWFKbGVzcy1zdmMuY2hnLXB0Yy5zdmMuY2xlc3Rlcj5sb2NhbDANBgkqhkiG
9w0BAQsFAAOCAQEALnhliDflu+BHp5conq4dXBwD/Ti2YR5TWQixM/0a6OD4KecZ
MmaLl0T+OJJvA/j2IufZpc7dzEx5mZDKr2CRmoq10qZXqCRTrBZSXm6ARQWoYpeg
9c0l4f8roxrkMGUKVPTKUwAvbnNYhD2l6PlBPwMpkMUFqFaSEXMaPyQKhrtQxdpH
WjuS54QP0lm0PeYu/yiad98LtrTXnb6jch84SKf6Vii4HAVQYMeJaW+dpkqcI2+V
Q4fkWYSJy8BncmXCwvHDLdy+s4EXWvHafhusuUhcp4HyMblA6hd5hJhgFSnEvLy
kLA0L9LaScxee6V756Vt9TN1NGjwmwyQDOhnQQ==
-----END CERTIFICATE-----
```

```
ca crt: |-
-----BEGIN CERTIFICATE-----
MIIDXCcAkSgAwIBAgIRAMPzF3BNFxt9HWE+NXlFQjQwDQYJKoZIhvcNAQELBQAw
VzEYMBYGA1UEChMPTXkgT3JnYW5pemF0aW9uMQswCQYDVQQLLEwJJDQTEuMwGA1UE
AxMlTXkgT3JnYW5pemF0aW9uIENlcnRpZmljYXRlIEF1dGhvcml0eTAeFw0yMTA0
MTkwNDQ0MjNaFw0zMTA0MTcwNDQ0MjNaMFcxGDAWBgNVBAoTD015IE9yZ2FuaXph
dGlvbWJELMAKGA1UECXMCMQ0EXLjAsBgNVBAMTJU15IE9yZ2FuaXphdGlvbiBDZXJ0
aWZpY2F0ZSBBDXRob3JpdHkwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIB
AQc5t6CS23G1k65Ymw5e4i4xH1dyxkCZS67w/6LWqeIlYKmfAae183Wwy8MHUpOb
4mahtUafEzDEOX6+URf72J8m0voldQ5FYr1AyUOyX8U90wGFqhbEgKRqt7vZEwIe
2961fwgHh6917zi4xmt5W6ZJ5dBQVtkhzB+Pf7O6KBYjHoCnBBkfnVzsfZQ/1hnr
```

```

0UzimfAc7Ze+UNwhXJhinFRJ3YuR+xiOTpPk1lGXPhLgFSQheKz4KepcbQEKejb
jg0dumloBYIXZTSSbi09rNmFUVLB5DcV0vZbSrGxLjWLBt5U8N2xf2d1bvkQW+bw
Kklf9OG26bAi27tuJurzN3r3AgMBAAGjIzAhMA4GA1UdDwEB/wQEAwICpDAPBgNV
HRMBAf8EBTADAQH/MA0GCSqGSIb3DQEBCwUAA4IBAQAAM0CN3n5C/KOT4uZ4ewWKK
rHmANBPVM9u6MJB08U62HcqLeoCuDFeU8zmUjLHjsQaPX64mJZlR7T5y52gEKO5A
0qsBz3pg/vJ5DJTtV0698+1Q1hB9k3smQdksAim19FZqysB7J4zK/+8aJ/q2kIFvs
Jk3ekwQdQ3xfggklBQVuf76gr1v0uYlPtFpfPlfcGZ06Im6mqbajenXoR1PxPB0
+zyCS8DkgPtDulplrwwXCFMYw9TPbzXKlt7tIsqRXogYLnXWJDzMlnOYCnD+rDm
qxenV9Ir8RqZ0XSYuUyzRka5N4dhIhrzTAiNdeU5gzynXOz67u/Iefz1iK9ZcdE3
-----END CERTIFICATE-----

```

4.5.4 Configurable Parameters

To enable MTLS, make changes to the following parameters.

Key	Value	Details
spec.fep.usePodName	True	For MTLS, this key must be defined and set to true. For TLS connection without MTLS, it can be omitted. However, it is recommended to set this to true as well.
spec.fep.patroni.tls.certificateName	<secret-name>	Name of Kubernetes secret that contains the certificate in tls.crt and private key in tls.key for Patroni REST API. For MTLS Patroni REST API communication, this key must be defined. The private key cannot be password protected. When using cert-manager, the secret also contains the CA bundle in ca.crt.
spec.fep.patroni.tls.caName	<configmap-name>	Name of Kubernetes configmap that contains the CA bundle. If using cert-manager, the ca.crt is already included in the secret above. In this situation, this key can be omitted.
spec.fep.postgres.tls.certificateName	<secret-name>	Name of Kubernetes secret that contains the certificate in tls.crt and private key in tls.key for Postgres server. For MTLS Postgres communication, this key must be defined. The private key can be password protected. When using cert-manager, the secret also contains the CA bundle in ca.crt.
spec.fep.postgres.tls.caName	<configmap-name>	Name of Kubernetes configmap that contains the CA bundle. If using cert-manager, the ca.crt is already included in the secret above. In this situation, this key can be omitted.

Key	Value	Details
spec.fep.postgres.tls.privateKeyPassword	<secret-name>	Name of Kubernetes secret that contains the password for the private key for Postgres Server.
spec.fepChildCrVal.sysUsers.pgAdminTls.certificateName	<secret-name>	Name of Kubernetes secret that contains the certificate in tls.crt and private key in tls.key for "postgres" user. For MTLS Postgres communication, this key must be defined. The private key cannot be password protected. When using cert-manager, the secret also contains the CA bundle in ca.crt.
spec.fepChildCrVal.sysUsers.pgAdminTls.caName	<configmap-name>	Name of Kubernetes configmap that contains the CA bundle. If using cert-manager, the ca.crt is already included in the secret above. In this situation, this key can be omitted.
spec.fepChildCrVal.sysUsers.pgAdminTls.sslMode	verify-full	For MTLS, this value must be set to verify-full. If only TLS is required, this can be set to verify-ca or prefer.
spec.fepChildCrVal.sysUsers.pgrepluserTls.certificateName	<secret-name>	Name of Kubernetes secret that contains the certificate in tls.crt and private key in tls.key for "repluser" user. For MTLS Postgres communication, this key must be defined. The private key cannot be password protected. When using cert-manager, the secret also contains the CA bundle in ca.crt.
spec.fepChildCrVal.sysUsers.pgrepluserTls.caName	<configmap-name>	Name of Kubernetes configmap that contains the CA bundle. If using cert-manager, the ca.crt is already included in the secret above. In this situation, this key can be omitted.
spec.fepChildCrVal.sysUsers.pgrepluserTls.sslMode	verify-full	For MTLS, this value must be set to verify-full. If only TLS is required, this can be set to verify-ca or prefer.
spec.fepChildCrVal.sysUsers.pgRewindUserTls.certificateName	<secret-name>	Name of Kubernetes secret that contains the certificate in tls.crt and private key in tls.key for "rewinduser" user. For MTLS Postgres communication, this key must be defined. The private key cannot be password protected. When using cert-manager, the secret also contains the CA bundle in ca.crt.
spec.fepChildCrVal.sysUsers.pgRewindUserTls.caName	<configmap-name>	Name of Kubernetes configmap that contains the CA bundle. If

Key	Value	Details
		using cert-manager, the ca.crt is already included in the secret above. In this situation, this key can be omitted.
spec.fepChildCrVal.sysUsers.pgRewindUserTls.sslMode	verify-full	For MTLS, this value must be set to verify-full. If only TLS is required, this can be set to verify-ca or prefer.

It is also required to customize pg_hba.conf to perform MTLS. Below are two possible settings.

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

The above setting will force FEP server to perform certification authentication. At the same time verify the authenticity of client certificate.

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
----------------------	---

The above setting will force FEP server to perform md5 authentication as well as verifying the authenticity of client certificate.

4.6 Replication Slots

4.6.1 Setting Up Logical Replication using MTLS

This section describes setup of logical replication.

To setup logical replication using MTLS, follow these steps:

1. Create two FEPClusters - to act as Publisher and Subscriber) and ensure that they can communicate with each other. You can see the creation of FEPCluster in the ["4.1 Deploying FEPCluster using Operator"](#).
2. To setup Publisher, make following changes to the FEPCluster yaml of the cluster that you want to use as publisher:
 - a. Add section replicationSlots under spec.fep to create replication slots.

The "**database**" should be the name of the database for which we are setting up logical replication.

```

158 spec:
159   fep:
160     forceSsl: true
161     replicationSlots: |
162       myslot1:
163         type: logical
164         database: db1
165         plugin: pgoutput
166       myslot2:
167         type: logical
168         database: db1
169         plugin: pgoutput
170     podAntiAffinity: false

```

- b. Add section postgres under spec.fep as shown below.
caName = enter the name of configmap created for the CA

certificateName = secret created by the end user that contains server certificate

```
78 |         memory: 512Mi
79 |         customAnnotations:
80 |           allDeployments: {}
81 |         servicePort: 27500
82 |         postgres:
83 |           tls:
84 |             caName: cacert
85 |             certificateName: my-fep-cert
86 |         image:
```

- c. Change the value of wal_level parameter under spec.fepChildCrVal.customPgParams from replica to logical.

```
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. Add entry under spec.fepChildCrVal.customPgHba as shown below.

This requires the client to present a certificate and only certificate authentication is allowed.

Replace "SubClusterName" and "SubNamespace" with the appropriate values as per the Subscriber FEPCluster.

```
[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. To setup Subscriber, make following changes to the FEPCluster yaml of the cluster that you want to use as subscriber:

- a. Add customCertificates under spec.fepChildCrVal as shown below.

caName = enter the name of configmap created for the CA (i.e. The CA certificate which is used to sign/authenticate the server/client certificates is mounted as a configMap called 'cacert')

certificateName = secret created by end user that contains a client certificate which can be verified by the server

username = name of the role created on publisher cluster for logical replication

```
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. Connect to the pod terminal of the Publisher FEPCluster and then connect to the postgres database as shown below.

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

postgres=#
```

5. Next, on the publisher side, connect to the database that contains the tables you want to replicate and create a role e.g., logicalrepluser and give the required permissions to this role.

Consider the below image as example only, the privileges to grant may differ as per the requirements.

```
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. At the Publisher side, create a publication and alter the publication to add the tables that need to be replicated.

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

7. At the subscriber side, the custom certificates added in the above step 3.a will be mounted at the path /tmp/custom_certs/ as shown:

```
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. The structure of the table to be replicated should be present in the subscriber cluster since logical replication only replicates the data and not the table structure.

Create a subscription as shown below:

```
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=verify-full dbname=db1 user=logicalrepluser' PUBLICATION my_publication WITH (slot_name=myslot1, create_slot=false);
CREATE SUBSCRIPTION
```

The command in the above example is :


```
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

Where

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 and used to establish logical replication connection fromSubscriber to Publisher
dbname= database which contains the tables to be replicated

4.7 FEP Logging

FEPCluster generates log files and auditlog files, if configured, over the lifetime of execution. These log files can be useful for understanding cluster healthness and debugging purpose. By default, the log files are stored on persistent volume of the container. User can enable log monitoring feature by forwarding those log files and auditlog files to a analytics platform such as Elasticsearch.

There are two steps to enable monitoring and forwarding.

1. FEPLogging Configuration - Creating FEP Logging instance
2. FEPCluster configuration - Enabling logging in FEPCluster

The FEP Logging instance is a standalone container running fluentd. It accepts log forwarded from FEP Clusters and aggregate data according to log entries severity and present that to Prometheus for monitoring and alerting purpose. It can optionally be configured to forward those logs to an Elasticsearch instance for detail analysis.

When logging is enabled on FEPCluster, a sidecar, containing fluentbit, will be deployed alongside the FEP server container. This fluentbit sidecar will monitor the FEP server log files and auditlog files on persistent volume and forward to the FEP Logging instance.

Multiple FEPClusters can forward logs to single FEPLogging instance.

User can have two types of connection between FEPCluster & FEPLogging

- Insecure connection: Without TLS/MTLS certificates
- Secure connection: With TLS/MTLS certificates

For the secure connections between the components, User have two options:

- User can use their own certificates
- User can generate self signed certificates (see "[4.5.2 Automatic Certificate Management](#)")

The FEP Logging instance can run standalone without additional component. For detail log analysis, the user can configure the FEP Logging instance to forward logs to Elastic Stack or Elastic Cloud. Please consult the [Elastic Document](#) on how to deploy a Elastic Stack or sign up to [Elastic Cloud](#).

4.7.1 FEPLogging Configuration

This section describes how to deploy and configure FEP Logging instance via the FEPLogging custom resource. FEPLogging is a separate CR which will accept logs sent from FEPCluster and forwards them to Elasticsearch or Prometheus for raising alarm. User must create FEPLogging CR before enabling FEPCluster logging feature.

4.7.1.1 FEPLogging Custom Resources - spec

The fepllogging section needs to be added under spec to define required parameters for FEPLogging configuration.

Following is a sample template :

```
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:
      ...
```

Below is the list of all parameters defined in the fepLogging section, along with their brief description

Custom Resource spec	Required/ Optional	Change Effect	Updating value allowed
spec.fepLogging.image.image	Optional	Fluentd Image of FEPLogging	Yes
spec.fepLogging.image.pullPolicy	Required	Fluentd Image pull policy of FEPLogging	Yes
spec.fepLogging.mcSpec.limits.cpu	Required	Max CPU allocated to fluentd container	Yes
spec.fepLogging.mcSpec.limits.memory	Required	Max memory allocated to fluentd container	Yes
spec.fepLogging.mcSpec.requests.cpu	Required	CPU allocation at start for fluentd container	Yes
spec.fepLogging.mcSpec.requests.memory	Required	Memory allocation at start for fluentd container	Yes
spec.fepLogging.sysExtraLogging	Required	To turn on extra debugging messages for operator, set value to true. It can be turned on/off at any time	Yes
spec.fepLogging.restartRequired	Required	To restart FEPLogging instance for applying any new configuration for example after certificate rotation	Yes
spec.fepLogging.scrapeInterval	Optional	Scrape interval for Prometheus to fetch metrics from FEPLogging instance	Yes

Custom Resource spec	Required/ Optional	Change Effect	Updating value allowed
spec.fepLogging.scrapeTimeout	Optional	Scrape Timeout for Prometheus to fetch metrics from FEPLogging instance	Yes
spec.fepLogging.elastic.host	Optional	Target Elasticsearch host name	Yes
spec.fepLogging.elastic.port	Optional	Target Elasticsearch port number	Yes
spec.fepLogging.elastic.authSecret.secretName	Optional	Secret name which contains Elasticsearch authentication username & password	Yes
spec.fepLogging.elastic.authSecret.userKey	Optional	Username key specified in Elasticsearch authentication secret	Yes
spec.fepLogging.elastic.authSecret.passwordKey	Optional	Password key specified in Elasticsearch authentication secret	Yes
spec.fepLogging.elastic.logstashPrefix	Optional	Logstash prefix to differentiate index pattern in elastic search. Default value is postgres	Yes
spec.fepLogging.elastic.auditLogstashPrefix	Optional	Logstash prefix to differentiate index pattern in elastic search for auditlog. If not specified, it will default to the same value as 'logstashPrefix'.	Yes
spec.fepLogging.elastic.scheme	Optional	Connection scheme between FEPLogging & Elasticsearch. Possible options http & https	Yes
spec.fepLogging.elastic.sslVerify	Optional	Set to true if you want to verify ssl certificate. If set to false then will not consider TLS certificate	Yes
spec.fepLogging.elastic.tls.certificateName	Optional	Kubernetes secret name which holds fluentd certificate	Yes
spec.fepLogging.elastic.tls.caName	Optional	Kubernetes configmap which holds cacert of Elasticsearch to verify Elasticsearch TLS connection	Yes
spec.fepLogging.tls.certificateName	Optional	Kubernetes secret name which holds Fluentd certificate	Yes
spec.fepLogging.tls.caName	Optional	Kubernetes configmap which holds cacert of Fluentd to configure MTLS between FEPLogging & Prometheus	Yes
spec.prometheus.tls.certificateName	Optional	Kubernetes secret name which holds Prometheus certificate	Yes
spec.prometheus.tls.caName	Optional	Kubernetes configmap which holds cacert of Fluentd to configure MTLS between FEPLogging & Prometheus	Yes

4.7.1.1.1 Define fepLogging image

The image property is used to specify other than default Fluentd image and its pullPolicy from FEPLogging CR.

If not specified it will use default image provided by Operator.

Example)

```
spec:
  fepLogging:
    image:
      image: 'quay.io/fujitsu/fujitsu-enterprise-postgres-fluentbit:ubi8-14-0.0'
      pullPolicy: IfNotPresent
```

4.7.1.1.2 Define fepLogging mcSpec

FEPLogging container Memory & CPU configuration can be provided by mcSpec properties.

Example)

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

4.7.1.1.3 Define fepLogging restartRequired

If FEPLogging required to be restarted to apply any new change, for example, after certificate rotation, FEPLogging container can be restarted by setting restartRequired flag as true. Default value of this flag is False. This flag will change back to false once the pod is restarted

Example)

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

4.7.1.1.4 Define fepLogging scrapeInterval and scrapeTimeout

scrapeInterval and scrapeTimeout properties of FEPLogging are optional. These properties are used by Prometheus Servicemonitor to configure metrics fetching interval(scrapeInterval) and timeout of request.

Example)

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

4.7.1.1.5 Define fepLogging elastic

To forward logs from FEPLogging(Fluentd) to Elasticsearch, need to configure elastic property. This is optional property. Elasticsearch server and certificates will be configured by user.

To configure log forwarding to Elasticsearch, the following properties are required.

- authSecret
- host
- port
- logstashPrefix
- auditLogstashPrefix
- scheme
- sslVerify
- tls(if sslVerify set to true)

Configure Elasticsearch server and use it's host name and port.

Here `tls` property is optional and works with `sslVerify` flag. To enable secure connection and `tls` verification set `sslVerify` true and provide valid `certificateName` & `caName`.

Elasticsearch `caName` is mandatory which holds CA cert of elastic search server.

Example)

```
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
```

4.7.1.1.6 Define `authSecret` for elastic

`authSecret` is the secret which contains username & password in base64 format for elastic search authentication

Example)

```
kind: Secret
apiVersion: v1
metadata:
  name: elastic-auth
  namespace: my-namespace
data:
  password: OFBobzlyRUJWOGg1Mk0xcXdaMUQ5bzQ0
  username: ZWxhc3RpYw==
type: kubernetes.io/basic-auth
```

4.7.1.1.7 Define `fepLogging` TLS

FEPLogging has optional `TLS` property. If user wants to forward logs from FEPCluster to FEPLogging instance over a secure connection, the `TLS` configuration for FEPCluster(`remoteLogging` section) and the `TLS` configuration for FEPLogging and Prometheus are mandatory. Configuring `TLS` configuration on just `fepLogging` or Prometheus will not work.

When a self signed certificate is used, `caName` can be skipped.

Example)

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

4.7.1.1.8 Define Prometheus TLS

If secured connection between FEPLogging and FEPCluster is required, then `TLS` configuration for FEPLogging and Prometheus are mandatory. Configuring `TLS` on just `fepLogging` or Prometheus will not work.

When a self signed certificate is used, caName can be skipped.

Example)

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

4.7.2 FEPCluster Configuration

This section describes how to enable logging in FEPCluster. FEP cluster provides a feature to forward logs to remote Fluentd(FEPLogging) and FEPLogging instance will forward the same logs to Elasticsearch(Optional) & Prometheus.

4.7.2.1 FEP Custom Resources - spec.fep.remoteLogging

The remoteLogging section needs to be added under fep to define required parameters for remoteLogging configuration.

Following is a sample template:

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

Below is the list of all parameters defined in the remoteLogging section, along with their brief description:

Custom Resource spec	Required/ Optional	Change Effect	Updating value allowed
remoteLogging.enable	Required	The 'enable' is set to true for enabling Logging feature	No
remoteLogging.fluentdName	Required	The 'fluentdName' is the name of the FEPLogging CR where logs will be forwarded	Yes
remoteLogging.tls.secretName	Optional	Secret name which contains MTLS certs of fluentbit	No
remoteLogging.tls.caName	Optional	Cacert of Fluentd for ssl verification	No
remoteLogging.image	Optional	Fluentbit image for remoteLogging	Yes
remoteLogging.pullPolicy	Optional	Fluentbit image pull policy	Yes

4.7.2.1.1 Define remoteLogging enable and fluentdName

The enable flag is used to describe that FEPCluster will enable log monitoring feature if set as true.

If enable flag set as true then fluentdName is the mandatory field. It will describe the FEPLogging CR name to which FEPCluster will forwards the logs.

If the enable flag is set as false, the FEPCluster will not enable logging feature.

Example)

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

If user wants to update existing FEPCluster with log monitoring feature then FEPCluster log_destination configuration must be set as **csvlogs**. For new cluster it will be already set.

Example)

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

fepChildCrVal:
  customPgParams:
    ...
    log_destination = csvlog
    ...
```

4.7.2.1.2 Define remoteLogging tls

When FEPCluster uses secure connection for remoteLogging, then TLS section is mandatory.

In the TLS section, provide the secret name that contains certificate and private key that is used for ssl verification.

For MTLS connection caName is required to mutually validate certificate.

Example)

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



Note

The Elasticsearch server is configured by user and it is NOT part of FEPLogging deployment by operator.

4.7.2.1.3 Define remoteLogging image

The image property is used to specify other than default Fluentbit image and it's pullPolicy.

If not specified it will use default image provided by Operator.

Example)

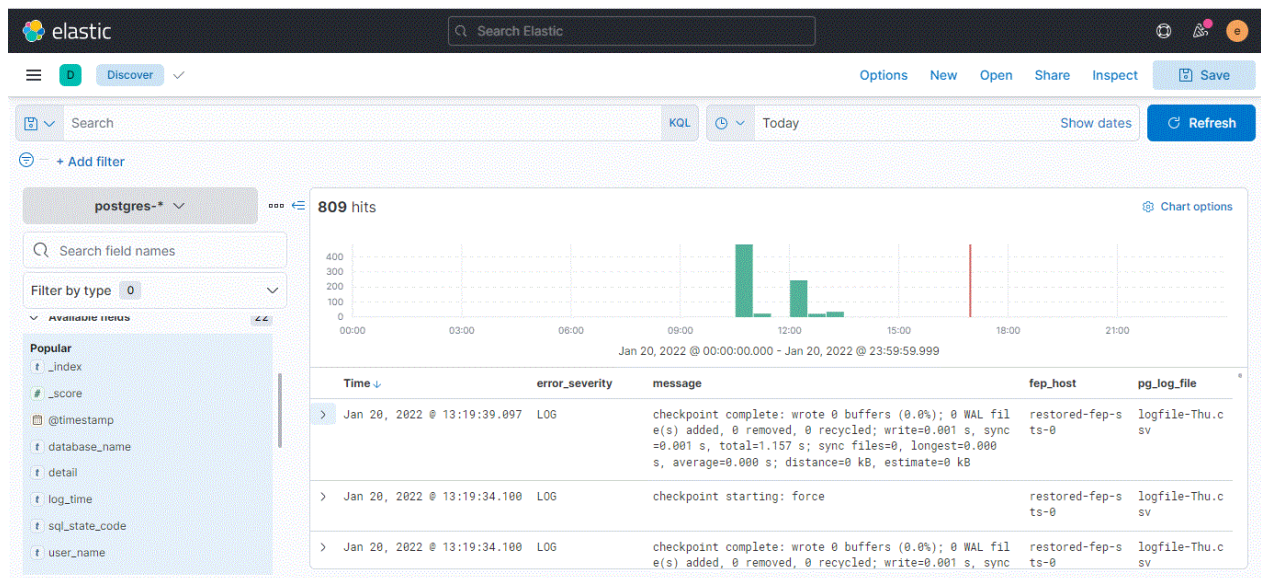
```
spec:
  fep:
    remoteLogging:
      image: 'quay.io/fujitsu/fujitsu-enterprise-postgres-fluentbit:ubi8-14-1.3'
      pullPolicy: IfNotPresent
```

4.7.3 FEPLoggging Operations

4.7.3.1 Log Forwarding to Elasticsearch

If the user has provided Elasticsearch configuration in the FEPLoggging CR, and FEPCluster is configured to send server log files and auditlog files to that FEPLoggging instance, those logs will be visible on Elasticsearch stack or Elastic Cloud. Assuming Elasticsearch has been configured with Kibana then logs will be visible in Kibana Dashbord. User can use `fep log csv` fields to create various Dashbord in Kiabana as well. LogstashPrefix and auditLogstashPrefix will be used to filter logs of specific FEPLoggging instance.

User can verify if FEPLoggging feature is configured properly or not by checking real time FEP logs are populating to the destination.

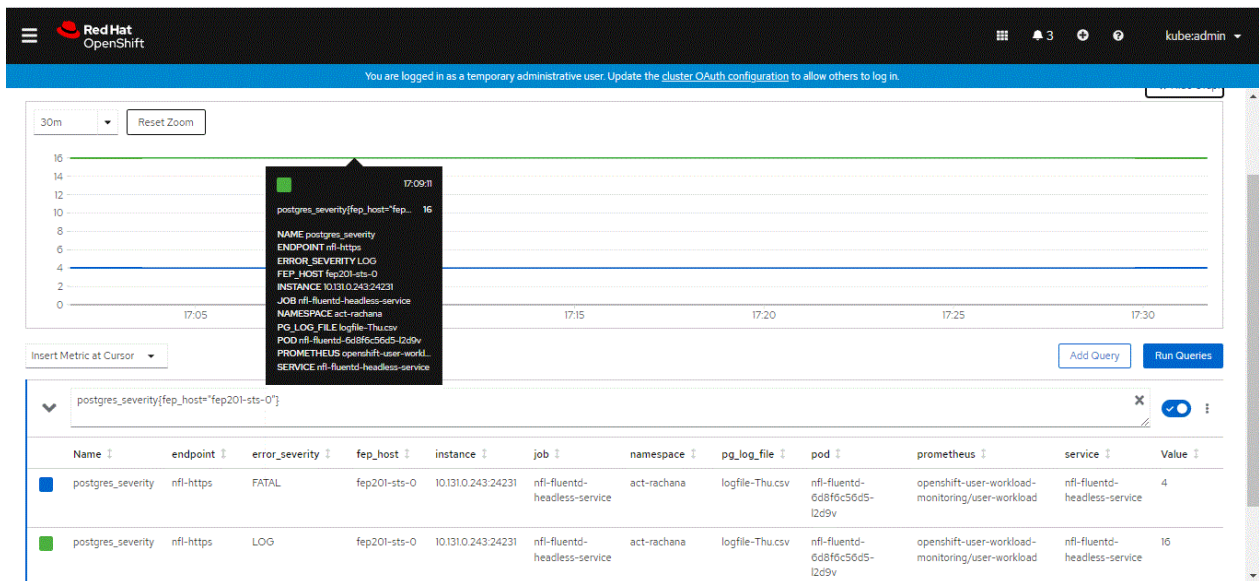


4.7.3.2 Log severity based Alarms/Metrics

FEPLoggging feature is used for raising alarm/alert based on postgres severity counts as well. While user creates FEPLoggging CR, Operator will forward real time counts of various postgres seviry metrics to Openshift managed Prometheus. Openshift managed Alertmanager can access this metrics counters and user can use them to create alerts/alarms. There are 4 default alert rules already created as part of FEPLoggging implementation as listed below:

- FEPLogErrorMessage
- FEPLogFatalMessage
- FEPLogPanicMessage
- FEPLogWarningMessage

Prometheus will scrape postgres_severity counter at every 30s as default scrape interval is 30s. User can modify this scrape interval from FEPLoggging CR. After each scrape interval, if any change/increment found in postgres_severity counter then alert rule will be fired. User can check counts of postgres_severity metrics anytime from Prometheus dashboard as well.



4.7.3.3 Forwarding auditlog to Elasticsearch

In order to forward auditlog to Elasticsearch, update the FEPCluster to enable creating auditlog.

Example)

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

4.7.4 Limitations

- Only postgres_severity including ERROR, PANIC, FATAL and WARNING are monitored.
- External fluentd can not be used for log monitoring and log forwarding.
- External Elasticsearch is required for log forwarding.
- User must decide at deployment time whether secured connection between FEPCluster and FEPLogging is required or not. After deployment, one can switch connection from insecure to secure but can not switch from secure to insecure connection.
- User must configure FEPLogging CR first then only FEPCluster can forward logs to particular FEPLogging otherwise Logging feature will not work.
- User must set log_destination in FEPCluster CR.

4.8 Configuring pgBadger

This section describes how to configure pgBadger. FEP cluster provides a feature to create pgbadger report on defined schedule and upload the report to a web server outside.

4.8.1 FEP Custom Resources - spec.fep.pgBadger

Custom Resource spec	Change Effect
pgBadger.schedules.create	The 'create' schedule to create report and upload it to endpoint
pgBadger.schedules.cleanup	The 'cleanup' schedule to delete the report left in container
pgBadger.options.incremental	Default: false; When set to True: create incremental report in pgbadger
pgBadger.endpoint.authentication	a secret to contain authentication info to access endpoint support basic auth only
pgBadger.endpoint.customCertificateName	Client certificate reference in customCertificate CR
pgBadger.endpoint.fileUploadParameter	The file upload parameter defined by the web server Default: 'file'
pgBadger.endpoint.insecure	equivalent to curl -insecure option, default to false
pgBadger.endpoint.url	Web server url to upload the report file

4.8.2 Define pgBdager Schedules

The schedules are used to create and run a job periodically, written in Cron format.

If the schedule format is invalid, the cronjob will not be created, so no pgBadger report will be created and uploaded.

Example)

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

4.8.3 Define pgBdager Options

When the incremental option is set to false, pgbadger will create normal html report and upload the html file to the web server.

When the incremental option is set to true, pgbadger will create incremental report and upload a zip file to the web server.

Example)

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

4.8.4 Define Endpoint for Uploading Report

Web server url

Both http and https are supported.

Example)

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

Web Server authentication

Only basic auth is supported

To configure web server authentication:

Create a base64 encoded text from username:password

Example)

```
$ echo -ne "myuser:mypass" | base64  
  
amFzb253Omphc29udw==
```

Wrap the output with base64 for creating a secret

Example)

```
$ echo -ne "amFzb253Omphc29udw==" | base64  
  
YWlGemIyNTNPbXB0YzI5dWR3PT0=
```

Create a secret by using the wrapped text. The key must be 'basic_auth'.

Example)

```
kind: Secret  
apiVersion: v1  
metadata:  
  name: pgbadger-endpoint-auth  
  namespace: fep-container-ct  
data:  
  basic_auth: YWlGemIyNTNPbXB0YzI5dWR3PT0=  
type: Opaque
```

Add the secret name in the endpoint definition.

Example)

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

Web Server certificates

When certificate files are required by the web server, FEP cluster provides customCertificate CR to mount the certificates files in container.

To use certificates for web server.

Create a secret based on the cert and key files.

Example)

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

The webserver.pem and webserver.key are certificate files for accessing web server

Create a configmap based on the CA cert.

Example)

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

The webca.pem is the CA certificate file for accessing web server.

Define custom certificates in FEPCluster CR.

Example)

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

The userName is a reference in the pgBadger endpoint.

The certificateName is the secret created above.

The caName is the configmap created above.

Refer the custom certificate name in pgbadger endpoint.

Example)

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

Insecure access to web server

The pgbadger CR provides an option to the web server endpoint when secure connection is not required:

Example)

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

File upload parameter

This parameter specify the request parameter for uploading a file to a web server. The value of this parameter is depended on the web server implementation.

Example)

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

curl command and parameters

FEP cluster uses curl command to upload the generated report to a web server endpoint. The CR in enpoint section will be converted to curl command parameters. The following table shows the mapping:

curl command parameter	User configuration
[URL]	Endpoint url
--cert	webserver.pem included in the secret referred in customCertificateName
--key	webserver.key included in the secret referred in customCertificateName
--cacert	webca.pem included in the configmap referred in customCertificateName
--form "uploadfile=@/path/to/report"	Endpoint fileUploadParameter
--header "Authorization: Basic passxxx"	Endpoint authentication configmap

curl command parameter	User configuration
--insecure	When endpoint.insecure is set to true

4.8.5 Uploaded File on Web Server

The FEP cluster uploads the pgbadger report according to the incremental mode:

incremental mode	Uploaded file name	Example
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

The zip file contains a folder of pgbadger incremental report.

Example)

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



Note

- The web server is NOT included in the FEP cluster solution.
- The web server is responsible to the uploaded files according to the customer's business logic.

4.9 Transparent Data Encryption Using a Key Management System

Describes how to configure transparent data encryption using a key management system.

Transparent data encryption using a key management system can only be configured when the FEPCluster is first created. Users cannot configure an existing FEPCluster for transparent data encryption using a key management system.

4.9.1 Certificate Registration

Save the certificate used for TLS communication between key management systems in Secret or ConfigMap.

The Secret or ConfigMap you created gives the FEPCluster custom resource a resource name and mounts it in the FEP container.

Create a Secret to store the client certificate and private key for connecting to your key management system.

Also, optionally create a ConfigMap to store the root certificate.

An example of registering credentials using the credentials file below is explained.

```
kmip.pem # Client certificate for connecting to key management system
kmip.key # Private key
myca.pem # Root certificate
```

Create a Secret to store the client certificate and private key.

Specify `tls.crt` and `tls.key` as file names when mounting the client certificate and private key, respectively.

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

Optionally create a `ConfigMap` to store your root certificates.

Specify `ca.crt` as the file name to be mounted.

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

4.9.2 Configuring FEPCluster Custom Resources

To enable TDE using a key management system, you need to set “`spec.fepChildCrVal.customPgParams`” and “`spec.fepChildCrVal.sysTde`”.

4.9.2.1 Define `spec.fepChildCrVal.customPgParams`

The `fepChildCrVal.customPgParams` section must define the following parameters:

`shared_preload_libraries`

Add the `'tde_kms'` library to the list of libraries in `shared_preload_libraries`.

Example)

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

Do not remove `'tde_kms'` library from `'shared_preload_libraries'` list after cluster creation.

4.9.2.2 Define `spec.fepChildCrVal.sysTde`

Add a `sysTde` section under `spec.fepChildCrVal` to define the parameters required to connect to your key management system. Under `sysTde` there are two parameters defined:

- `tdeType`
- `tdek`

Define `spec.fepChildCrVal.sysTde.tdeType`

`sysTde` itself is an optional parameter (if `sysTde` is not defined, TDE with a passphrase is implemented). However, if `sysTde` is defined by the user, `sysTde.tdeType` must also be defined.

If configuring TDE with a key management system, set `sysTde.tdeType` to `"tdek"`.

Example)

```
sysTde:
  tdeType: tdek
```

Define `spec.fepChildCrVal.sysTde.tdek`

If you set `sysTde.tdeType` to `"tdek"`, you must also define `sysTde.tdek`.

Define the connection information of the key management system in `sysTde.tdek.kmsDefinition`. Based on the information defined here, the operator creates the key management system connection information file used by FUJITSU Enterprise Postgres.

Example)

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

Refer to the Reference for details of each parameter.

For the parameter under cert of `fepChildCrVal.sysTde.tdek.kmsDefinition`, specify the Secret or ConfigMap name created in "[4.9.1 Certificate Registration](#)".

Example)

```
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
```

Chapter 5 Post-Deployment Operations

This chapter describes the operation after deploying the container.

5.1 How to Connect to a FEP Cluster

When connecting from within the same project of the OpenShift system

Service resources are used to connect to FEPCluster and FEPPgpool2 from within the same project.

A service resource provides a single endpoint for communicating with containers.

Service resources are created with the following naming conventions.

FEPCluster service

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

FEPPGPool2 service

- <FEPPgpool2 name>-feppgpool2-svc

Example of checking service resources of FEPCluster container and FEPPgpool2 container

```
$ oc get all
```

Check where the resource type is Service (Begin with "svc/").

You can also check with the oc get svc command. The following is an example.

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

Example of accessing FEPPgpool2 container

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

```

When connecting from outside the OpenShift system

Automatically creating a service with ClusterIP to connect to the deployed container. You can connect to FEP or FEP pgpool2 services from the OpenShift system's internal network. To access from outside the OpenShift system, you need to know the address of the OpenShift node.

For example, "Access the FEP pgpool2 container from an application server that is running outside the OpenShift system but is part of the Internal network".

An example of how to check the node IP in OpenShift.

```
$ 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
```

An example of verifying the service resource for the FEP pgpool2 container.

```
$ oc get all
```

Check where the resource type is Service (Begin with "svc/").

You can also see this with the oc get svc command. The following is an example.

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

This is an example of accessing the FEP pgpool2 container.

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

5.2 Configuration Change

This section describes changes to the FEPCluster configuration.

List FEPCluster

Equivalent Kubernetes command: `kubectl get FEPClusters (-A)`

This operation will list all FEPClusters in a namespace, or if the `-A` option is specified, will list all FEPClusters in all namespace.

Default output format:

Field	Value	Details
NAME	.metadata.name	Name of Cluster
AGE	Elapsed time	Indicates the amount of time that has elapsed since the cluster was created

Example)

```
# kubectl get fepclusters -A

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

Update FEPCluster

Equivalent Kubernetes command: `kubectl apply -f <new_spec>`

Operations that can be performed here.

Custom Resource spec	Change effect
.spec.fep.instances: <i>n</i>	Increase the number of nodes in the cluster to <i>n</i> .
.spec.fep.image.image: 'quay.io/fujitsu/fujitsu-enterprise-postgres-14-server:ubi8-14-1.1'	Minor upgrade of FEP image to ubi8-14-1.1.
spec.fepChildCrVal.backup.image.image: 'quay.io/fujitsu/fujitsu-enterprise-postgres-14-backup:ubi8-14-1.1'	Minor upgrade of Backup image to ubi8-14-1.1.

This will impact behaviour for values in `feh` section only.
All parameters can be updated from the `FEPCluster` custom resource.

Delete FEPCluster

Equivalent Kubernetes command: `kubectl delete FEPCluster <cluster_name>`

This operation will remove the `FEPCluster` by the `cluster_name` and all Child CRs (`FEPVolume`, `FEPConfig`, `FEPcert` & `FEPUser`) & resources associated with it.



Note

Deleting a `FEPCluster` will delete all PV associated with the cluster, including backup and archived WAL volumes (except when using pre-made PV or AWS S3). This is an unrecoverable action.

5.3 FEPCluster Resource Change

5.3.1 Changing CPU and Memory Allocation Resources

Describes how to change the CPU and memory resources assigned to a pod created by a `FEPCluster`.

This allows you to scale the pod vertically through custom resources.

To modify CPU and memory resources, modify the `spec.feh.mcSpec` section(*1) of the `FEPCluster` custom resource and apply your changes.

When the changes are applied, restart the replica server with the new resource settings. If there are multiple replica servers, restart them one at a time. When all replica servers are restarted, one of them is promoted to the new master server due to a switchover. Then restart the container image on the original master server. This allows you to change resource settings for all servers with minimal disruption.

*1) Modifying this section scales up the FEP server container. For information about other container resource sections, refer to "FEPCluster Parameters" in the Reference.

5.3.2 Resizing PVCs

Describes how to resize a PVC assigned to a pod created by a `FEPCluster`.

This allows you to increase the size of the volume allocated to the pod through custom resources.

To change the PVC size, modify the size of each volume in the `spec.fehChildCrVal.storage` section of the `FEPCluster` custom resource and apply the change. These changes apply to all PVCs assigned to the pod created by the `FEPCluster`.



Note

- PVC resizing is extensible only.
- You can resize a PVC only if the `StorageClass` supports dynamic resizing.
- If the `StorageClass` does not support resizing PVCs, use the `FEPRestore` custom resource to create a new `FEPCluster` to resize the PVC. For more information, refer to "FEPRestore Custom Resource Parameters" in the Reference.

5.4 FEPPGPool2 Configuration Change

This section describes changes to the `FEPPGPool2` configuration.

List FEPPGPool2

Equivalent Kubernetes command: `kubectl get FEPPGPool2 (-A)`

This operation will list all FEPPGPool2 in a namespace, or if the -A option is specified, will list all FEPPGPool2 in all namespace.

Default output format:

Field	Value	Details
Name	.metadata.name	Name of pgpool2

Example)

```
# kubectl get feppgpool2 -A

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

Delete FEPPGPool2

Equivalent Kubernetes command: kubectl delete FEPPGPool2 <pgpool2_name>

This operation will remove the FEPPGPool2 by the pgpool2_name.

Update FEPPGPool2

Equivalent Kubernetes command: kubectl apply -f <new_spec>

Specify updated parameters in the format described in "2.3.4 Deploying Pgpool-II and Connect to FEPCluster from Operator". Only following parameters would change for Operations that can be performed here.

Custom Resource spec	Change Effect
.spec.count: n	Increase the number of nodes in the cluster to n.
.spec.serviceport	Change the TCP port for connecting to the Pgpool-II.
.spec.statusport	Change the TCP port for connecting to the PCP process.
.spec.limits.cpu	Change limits of cpus.
.spec.limits.memory	Change limits of memory.
.spec.requests.cpu	Change requests of cpus.
.spec.requests.memory	Change requests of memory.
.spec.fepclustername	Change fepcluster to connect.
.spec.customhba	Change pool_hba.conf file.
.spec.customparams	Change pgpool2 parameters
.spec.custompcp	Change pcp.conf file.
.spec.customsslkey	Change key content
.spec.customsslcert	Change the contents of the public x 509 certificate.
.spec.customsslcert	Change the contents of the CA root certificate in PEM format.

Some of the customparams parameters, customhba and custompcp, require a restart of pgpool2.

Equivalent Kubernetes command: Kubectl apply -f <new_spec>

"pgpool2_restart" action type expects users to specify the name of the pgpool2 that they want to restart from.

Specify the metadata.Name of the FEPPGPool2 CR in the targetPgpool2Name section of the FEPACTION CR, as below:

```
spec:
  targetPgpool2Name: fep1-pgpool2
```

```
fepAction:
  type: pgpool2_restart
```

Note

When updating FEPPGPool2, the Pod of FEPPGPool2 is restarted. If configured with more than one FEPPGpool2, they are rebooted sequentially. The application should be designed to reconnect the connection because the connection being connected is broken.

5.5 Scheduling Backup from Operator

Operational status confirm

Information about the backup can be found by running the command in the FEP backup container, as shown in the example below.

```
$ oc exec pod/fepserver-XXXXX -c FEPbackup -- pgbackrest info
stanza: fepbackup
  status: ok
  cipher: none

db (current)
  wal archive min/max (12-1): 00000001000000000000000001/000000010000000000000005

  full backup: 20201125-025043F
    timestamp start/stop: 2020-11-25 02:50:43 / 2020-11-25 02:50:52
    wal start/stop: 00000001000000000000000003 / 00000001000000000000000003
    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: 00000001000000000000000005 / 00000001000000000000000005
    database size: 31.7MB, backup size: 24.3KB
    repository size: 3.9MB, repository backup size: 619B
    backup reference list: 20201125-025043F
```

Update FEPBackup

Equivalent Kubernetes command: `kubectl apply -f <new_spec>`

Specify updated parameters in the format described in "[2.3.5 Scheduling Backup from Operator](#)". Only following parameters would change for Operations that can be performed here.

Custom Resource spec	Change Effect
spec.schedule.num	Change the Number of Registered Backup Schedules
spec.scheduleN.schedule	Change the scheduled backup time
spec.scheduleN.type	Change the scheduled backup type
spec.pgBackrestParams	Change pgBackRest parameters
spec.scheduleN.repo	If you specified more than one repository for spec.pgBackrestParams, select the repository in which to store the backup data. The default is 1.

Note

- Changes made during the backup are reflected from the next backup.

- Changes to the backup schedule do not affect the application.
- If you perform any of the following update operations, be sure to obtain a backup after the update.
 - When the master encryption key is updated with `pgx_set_master_key`
 - When the encryption passphrase for transparent data encryption is updated (can be updated by the `tdeppassphrase` parameter of FEPCluster CR)

5.6 Configure MTLS Setting

5.6.1 Certification Rotation

All certificates are bounded by the time limit. At certain time, it needs to be renewed. We recommend to renew the certificate when it reaches 3/4 of its life cycle or as soon as possible if it is compromised. When a certificate is renewed, we need to rotate it inside the FEP server container. At the moment, FEP server container does not support automatic certificate rotation. Depending on which certificate has renewed, there are different procedures to handle that.

Patroni Certificate Rotation

When Patroni certificate is renewed, we have to re-deploy each and every Pod for FEP server container to pick up the new certificate. There is a down time on FEPCluster.

FEP Server Certificate Rotation

When FEP Server certificate is renewed, we can use FEPAction CR to trigger a reload of the database and FEP server will pick up the new certificate with no interruption to service.

Client certification Rotation

When any of the client certificate is renewed, FEP server container internally will use the new certificate next time it establishes a connection to FEP server. However, to avoid any unexpected interruption to service, it is recommended to re-deploy each and every Pod as soon as possible.

5.7 Monitoring

Monitoring is collecting historic data points that you then use to generate alerts (for any anomalies), to optimize databases and lastly to be proactive in case something goes wrong (for example, a failing database).

There are five key reasons to monitor FEP database.

1. Availability

It is a very simple equation that if you do not have a database in running, your application will not work. If the application is critical, it directly effects on users and the organization.

2. System Optimization

Monitoring helps to identify the system bottlenecks and according to the user can make changes to your system to see if it resolves the problem or not. To put this into perspective, there may be a situation where users see a very high load on the system. And figured out that there is a host parameter that can be set to a better value.

3. Identify Performance Problems

Proactive monitoring can help you to identify future performance problems. From the database side, it could be related to bloating, slow running queries, table and index statistics, or the vacuum being unable to catch up.

4. Business Process Improvement

Every database user has a different need and priority. Knowing the system (load, user activity, etc.) helps you to prioritize customer tasks, reporting, or downtime. Monitoring helps to make business process improvement.

5. Capacity Planning

More user or application growth means more system resources. It leads to key questions: Do you need more disk space? Do you need a new read replica? Do you need to scale your database system vertically? Monitoring helps you to understand your current system utilization—and if you have data, points spread over a few weeks or months, it helps to forecast system scaling needs.

This article describes monitoring and alerting operations using OpenShift's standard Pod alive monitoring, resource monitoring and database statistics provided by the FEP Exporter.

5.7.1 Monitoring FEP Operator and Operands

The monitoring of FEP operators and operands are achieved by Prometheus' standard alive and resource monitoring.

Metrics name	Details
Alive monitoring	Can monitor Pod status
Resource monitoring	<p>You can monitor the following resource status</p> <ul style="list-style-type: none">- CPU Usage- CPU Quota- Memory Usage- Memory Quota- Current Network Usage- Receive Bandwidth- Transmit Bandwidth- Rate of Received Packets- Rate of Transmitted Packets- Rate of Received Packets Dropped- Rate of Transmitted Packets Dropped

By setting alert rules based on these monitoring items, operators and operands can be monitored. For the setting method, refer to the appendix in the Reference.

If an error is detected by monitoring the operator's alive, it can be dealt with by recreating the Pod.

If resource monitoring detects an error, consider allocating more resources to the Operator or Operands.

Check the Operator Hub or Red Hat Operator Catlog page to see which version you are currently using, which can be updated, and to check for security vulnerabilities.

5.7.2 Monitoring FEP Server

Monitoring and alerts system leverages standard GAP stack (Grafana, Alert manager, Prometheus) deployed on OCP and Kubernetes. GAP stack must be there before FEP operator & FEPCluster can be deployed.

Prometheus is a condensed way to store time-series metrics. Grafana provides a flexible and visually pleasing interface to view graphs of FEP metrics stored in Prometheus.

Together they let store large amounts of metrics that user can slice and break down to see how the FEP database is behaving. They also have a strong community around them to help deal with any usage and setup issues.

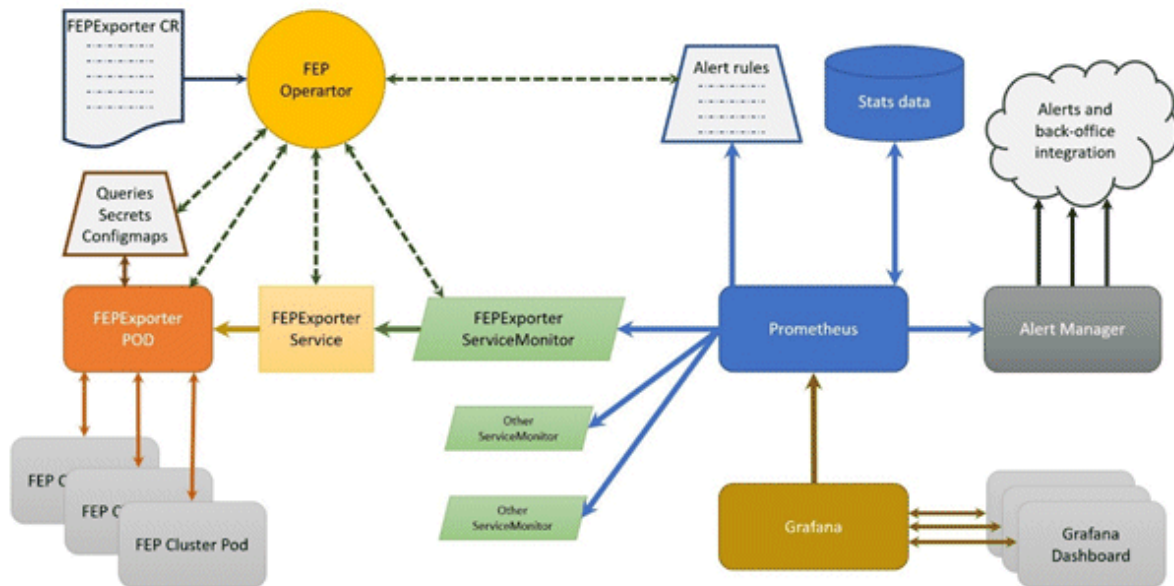
The Prometheus acts as storage and a polling consumer for the time-series data of FEP container. Grafana queries Prometheus to displaying informative and very pretty graphs.

If Prometheus rules are defined, it also evaluates rules periodically to fire alerts to Alert manager if conditions are met. Further Alert manager can be integrated with external systems like email, slack, SMS or back-office to take action on alerts raised.

Metrics from FEP Cluster(s) is collected by Prometheus through optional components deployed using FEP Exporter with default set of metrics and corresponding Prometheus rules to raise alerts. User may extend or overwrite metrics by defining their custom metrics queries and define their custom Prometheus rules for alerting.

5.7.2.1 Architecture

Block diagram of monitoring FEP server is as follows.



- FEPExporter CR is managed by FEP Operator
- When FEPExporter CR is created, FEP operator creates following kubernetes objects:
 - ConfigMap that contains default and custom queries to collect metrics from database cluster from each node
 - Secret containing JDBC URL for all FEPCluster nodes to connect and request metrics. This string contains authentication details as well to make JDBC connection.
 - Prometheus rules corresponding to default alert rules
 - ServiceMonitor for Prometheus to discover FEPExporter service
 - FEPExporter container using FEPExporter image to scrape metrics from all FEPCluster nodes

Note

- Alert Manager integration to back-office to send mail / message / raising ticket is done by user based on their environment
- Grafana installation and integration is done by user. Use the Grafana Operator provided by OperatorHub.
- Grafana dashboard is created by user based on their requirements and design.

5.7.2.2 Default Server Metrics Monitoring

By default FEPExporter scrapes some useful metrics for server.

Once FEPExporter is running, user can check the collected metrics under Openshift->Monitoring->Metrics submenu.

There are 2 levels of default server metrics defined by FEP Exporter

Type	Details
Default mandatory	Are collected by FEP Exporter. These are kept enabled by default by FEP Exporter and can not be disabled by end user.
Default useful	Useful focused metrics for health and performance metrics. Can be disabled by end user.

Default mandatory metrics

These metrics are either from basic statistics view of the database or FEP Exporter own metrics;

Various metrics under this category are

Metrics name	Details
pg_stat_bgwriter_*	Maps to view in Statistic Collector
pg_stat_database_*	Maps to view in Statistic Collector
pg_stat_database_conflicts_*	Maps to view in Statistic Collector
pg_stat_archiver_*	Maps to view in Statistic Collector
pg_stat_activity_*	Maps to view in Statistic Collector
pg_stat_replication_*	Maps to view in Statistic Collector
pg_replication_slots_*	Maps to System Catalog pg_replication_slots
pg_settings_*	Maps to System Catalog pg_settings
pg_locks_*	Maps to System Catalog pg_locks
pg_exporter_*	Exposes exporter metrics: <ul style="list-style-type: none"> - last_scrape_duration_seconds (Duration of the last scrape of metrics from PostgreSQL) - scrapes_total (Total number of times PostgreSQL was scraped for metrics) last_scrape_error (Whether the last scrape of metrics from PostgreSQL resulted in an error; 1 for error & 0 for success)
pg_*	Exposes exporter metrics <ul style="list-style-type: none"> - pg_up (set to 1 if the connection to service is success, 0 otherwise) - pg_static (can be used to fetch label short_version / version containing postgres server version information)

Default useful metrics

There are certain useful queries which are additionally added to evaluate the health of the Database system.

Metrics name	Details
pg_capacity_connection_*	Metrics on connections e.g. txns running for 1 hour
pg_capacity_schema_*	Metrics on disk space of schema
pg_capacity_tblspace_*	Metrics on disk space of tablespace
pg_capacity_tblvacuum_*	Metrics on tables without vacuum for days
pg_capacity_longtx_*	Number of transactions running longer than 5 minutes Review the information and consider SQL tuning and resource enhancements.
pg_performance_locking_detail_*	Details of processes in blocked state
pg_performance_locking_*	Number of processes in blocked state

Metrics name	Details
pg_replication_*	Replication lag behind master in seconds Provides the ability to check for the most current data in a reference replica To solve the problem, it is necessary to consider measures such as increasing network resources and reducing the load
pg_postmaster_*	Time at which postmaster started
pg_stat_user_tables_*	Important statistics from pg_stat_user_tables
pg_statio_user_tables_*	Important statistics from pg_statio_user_tables
pg_database_*	Database size If the database runs out of space, database restore is required
pg_stat_statements_*	Statistics of SQL statements executed by server
pg_capacity_ttblbloat_*	Fetches bloat in tables



Note

You can tune the intervals and thresholds at which information is gathered by changing the values specified in the information gathering query. For more information, refer to the queries in the appendix of the Reference Guide, and make your own settings.

Refer an example below.

Alert rule	Alert Level	Condition persistence	Description
pg_stat_activity_count	Warning	5 mins	FEP server container/Pod CPU usage is exceeding 80% of the resource limits
pg_stat_activity_count	Warning	30 mins	FEP server container/Pod memory usage is exceeding 80% of the resource limits
PVCLowDiskSpace	Warning	5 mins	A FEP PVC (volume) has less than 10% disk available

5.7.2.3 Default Alerts

There are few basic alert rules which are setup by the FEP Operator as below

Alert rule	Alert Level	Condition persistence	Description
ContainerHighCPUUsage	Warning	5 mins	FEP server container/Pod CPU usage is exceeding 80% of the resource limits
ContainerHighRAMUsage	Warning	30 mins	FEP server container/Pod memory usage is exceeding 80% of the resource limits
PVCLowDiskSpace	Warning	5 mins	A FEP PVC (volume) has less than 10% disk available

Alert rule	Alert Level	Condition persistence	Description
ContainerDisappeared	Warning	60 seconds	FEP server container/Pod has disappeared since last 60 seconds
PostgresqlDown	Error	-	FEP server apparently went down or not accessible
PostgresqlTooManyConnections	Warning	-	FEP server container/Pod connection usage is beyond 90% of its available capacity

** The alerts are based on statistics/metrics. If a platform statistics are incorrect, it may raise an incorrect alarm.

e.g. if the Storage Driver is not showing correct metrics for bytes usage for a PV, system may end up raising incorrect alarm of PVCLowDiskSpace. This behaviour can be seen with NFS storage.

You can configure any alert by adding alert rules to other monitoring items.

5.7.2.4 Graphical user interface

User can build their custom dashboard using default and custom metrics.

An example Grafana dashboard screenshot is shown below



5.7.3 Monitoring FEP Backup

You can view information about the backed-up data and the status of the backup process in the FEP server tables and system views.

Backup information is updated when the automatic backup process completes or when backup data is deleted as specified by retention.

The following tables and views are added. The tables and views to be added are created under the `feep_exporter` schema in the postgres database on the FEP server.

Table/View name	Details
<code>pgbackrest_info_backup</code>	Backup Processing Status

5.7.3.1 pgbackrest_info_backup view

Contains one line per backup for information about the state of the backup.

Column	Type	Description
<code>label</code>	text	Information identifying the backup

Column	Type	Description
type	text	full: full backup, incr: incremental backup
prior	text	Label of the backup that should be applied first (For incremental backups only)
database_size	bigint	Database size
database_size_comp	bigint	Database size (After Compression)
backup_size	bigint	Backup size
backup_size_comp	bigint	Backup size (After Compression)
archive_start	text	Range of WALs required for restore (Start)
archive_stop	text	Range of WALs required for restore (End)
backup_start	timestamp with timezon	Backup Start Time
backup_stop	timestamp with timezone	Backup End Time
backup_exec_time	interval	The duration of the backup

5.7.4 Monitoring FEP PGPool2

Information about pgpool2 activity and replication status can be found in the FEP server table and in the system view.

The pgpool2 statistics are updated according to the schedule specified in the parameter.

The tables and views that have been added are described below. The tables and views to be added are created under the `fepegpoolschema` in the postgres database on the FEP server.

Table/View name	Details
pgpool2_stat_load_balance	Load Balance Information in pgpool2
pgcluster_stat_replication	Replication State
pgpool2_stat_conn_pool	Connection Pool State for pgpool2
pgpool2_stat_sql_command	SQL Command Statistics

5.7.4.1 pgpool2_stat_load_balance view

Contains one row for MasterService and one row for ReplicaService.

Column	Type	Description
node_id	integer	database node id (0 or 1)
status	text	status (up or down)
lb_weight	double precision	load-balancing weight
role	text	role (primary or standby)
last_status_change	timestamp with time zone	last status change time

5.7.4.2 pgpool2_stat_conn_pool view

Indicates the state of the connection pool. Contains connection pool information for each pgpool2 instance.

Column	Type	Description
pgpool2_node_id	integer	pgpool2 node id (0 - the number of pgpool2 instance -1)
pool_pid	integer	The PID of the displayed Pgpool-II process

Column	Type	Description
start_time	timestamp with timezone	The timestamp of when this process was launched
pool_id	integer	The pool identifier (should be between 0 and max_pool - 1)
backend_id	integer	The backend identifier (should be between 0 and the number of configured backends minus one)
role	text	role (primary or standby)
database	text	The database name for this process's pool id connection
username	text	The user name for this process's pool id connection
create_time	timestamp with timezo	The creation time and date of the connection
majorversion	integer	The protocol version numbers used in this connection
minorversion	integer	The protocol version numbers used in this connection
pool_counter	integer	Counts the number of times this pool of connections (process) has been used by clients
pool_connected	boolean	True (1) if a frontend is currently using this backend

5.7.4.3 pgpool2_stat_sql_command view

Represents SQL command statistics.

Column	Type	Description
node_id	integer	The backend identifier (should be between 0 and the number of configured backends minus one)
role	text	role (primary or standby)
select_cnt	integer	The numbers of SQL command: SELECT
insert_cnt	integer	The numbers of SQL command: INSERT
update_cnt	integer	The numbers of SQL command: UPDATE
delete_cnt	integer	The numbers of SQL command: DELETE
ddl_cnt	integer	The numbers of SQL command: DDL
other_cnt	integer	The numbers of SQL command: others
panic_cnt	integer	The numbers of failed commands
fatal_cnt	integer	The numbers of failed commands
error_cnt	integer	The numbers of failed commands

5.8 Event Notification

The eventing mechanism introduced, is to enable operator to raise customized Kubernetes events. The custom events will be raised during the creation of custom resources. Currently following events are raised.

5.8.1 Events raised

- fecluster - During FEPCluster CR creation
 - Event is raised when FEPCluster CR creation is initiated and when FEPCluster CR creation initiation fails.
 - Event is raised when FEPCluster CR creation is initiated and when FEPCluster CR creation initiation fails.
 - Event is raised when FEPCluster CR creation is initiated and when FEPCluster CR creation initiation fails.

- Event is raised when FEPCert CR creation is initiated and when FEPCert CR creation initiation fails.
- Event is raised when Statefulset creation is successful and Statefulset creation fails.
- Event is raised when PDB creation is successful and when PDB creation fails.
- Event is raised when FEPBackup CR creation is initiated and when FEPBackup CR creation initiation fails.

Please note the following child CR events are raised as part of Create FEP Cluster

- fepcert - During FEPCert CR creation
 - Event is raised when FEPCert CR creation is successful, when FEPCert CR fails annotating FEPCluster and when FEPCert CR creation fails.
- feconfig - During FEPConfig CR creation
 - Event is raised when FEPConfig CR creation is successful, when FEPConfig CR fails annotating FEPCluster and when FEPConfig CR creation fails.
- fepvolume - During FEPVolume CR creation
 - Event is raised when FEPVolume CR creation is successful, when FEPVolume CR fails annotating FEPCluster and when FEPVolume CR creation fails.
- febackup - During FEPBackup CR creation
 - Event is raised when FEPBackup cronjob1 creation is successful and when FEPBackup cronjob1 creation fails.
 - Event is raised when FEPBackup cronjob2 creation is successful and when FEPBackup cronjob2 creation fails.
 - Event is raised when FEPBackup cronjob3 creation is successful and when FEPBackup cronjob3 creation fails.
 - Event is raised when FEPBackup cronjob4 creation is successful and when FEPBackup cronjob4 creation fails.
 - Event is raised when FEPBackup cronjob5 creation is successful and when FEPBackup cronjob5 creation fails.
- feppgpool2- During FEPPgPool2 CR creation
 - Event is raised when FEPPgPool2 CR creation is successful and when FEPPgPool2 CR creation fails.
 - Event is raised when FEPPgPool2Cert CR creation is initiated and when FEPPgPool2Cert CR creation initiation fails.

Please note the following child CR event are raised as part of Create FEP PgPool2

- feppgpool2cert- During FEPPgPool2Cert CR creation
 - Event is raised when FEPPgPool2Cert CR creation is successful, when FEPPgPool2Cert CR fails annotating FEPPgPool2 and when FEPPgPool2Cert CR creation fails
- feprestore - During FEPRestore CR creation
 - Event is raised when FEPRestore CR creation is successful and when FEPRestore CR creation fails.

5.8.2 Viewing the custom events

The custom events can be viewed on CLI as well as the Openshift console

On cli

Executing the command

kubectrl get events

OR

oc get events

Following is a snippet of the events output is ==shown when the above command is executed,

14m	Normal	InitiatedChildCRCreate	fecluster/new-fep-hg-12-08-21	playground-hg, Started FEP Volume CR creation
13m	Normal	InitiatedChildCRCreate	fecluster/new-fep-hg-12-08-21	playground-hg, Started FEP User CR creation
13m	Normal	InitiatedChildCRCreate	fecluster/new-fep-hg-12-08-21	playground-hg, Started FEP Cert CR creation
13m	Normal	InitiatedChildCRCreate	fecluster/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

On openshift console

For the specific project/ namespace the custom events can be viewed along with Kubernetes events under the events as shown in the following screenshot.

Project: playground-hg ▾

Events

Resources 1 ▾ Normal ▾ Filter Events by name or message... /

Resource All x x

Streaming events... Showing 46 events

FEPB new-fep-hg-12-08-21 **NS** playground-hg Aug 12, 5:49 pm

Generated from fepbackups

playground-hg, Successfully created FEP Backup Cronjob2

FEPB new-fep-hg-12-08-21 **NS** playground-hg Aug 12, 5:49 pm

Generated from fepbackups

playground-hg, Successfully created FEP Backup Cronjob1

5.9 Scaling Replicas

5.9.1 Auto Scale Out

Auto scale out occurs when the average CPU utilization or number of connections of the DB container exceeds the threshold.

The maximum number of replica containers, excluding the master container, is 15.

If the load decreases after the number of replicas increases due to a temporary increase in load, the number of replicas will remain increased. Perform manual scale in if necessary.

Specify `spec.fepChildCrVal.autoscale.scaleout` in `FEPClusterCR` when you want to perform Auto scale out. Refer to "FEPCluster Parameters" in the Reference for information about the values to specify.

```
$ oc edit fecluster <FEPClusterCR name>
```

5.9.2 Manual Scale In/Out

To manually scale in or out of a `FEPCluster`, edit the "`spec.fep.instances`" in `FEPClusterCR`.

The value must be between 1 and 16. (Number of instances with one master)

```
$ oc edit fecluster <FEPClusterCR name>
```



Note

- Do not scale in from two to one replica instance when the `syncMode` is 'on'. Update SQL cannot be executed.
- Any database connections to the replica Pod that are deleted during a scale in will be forced to disconnect.

5.10 Backing Up to Object Storage

Describes how to store backup data in object storage.

5.10.1 Pre-creation of Resources

5.10.1.1 Storing CA Files (Root Certificates)

If you want to use a non-default root certificate for object storage connections, register it in ConfigMap.

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

5.10.1.2 Storing Repository Key

When using the parameter (repo-gcs-key) of pgBackRest, register the GCS repository key in Secret.

```
$ oc create secret generic storage-key-secret --from-file=key.json=storage-key.json -n my-namespace
```

5.10.2 Defining a FEPCluster Custom Resource

List the backup settings under spec.fepChildCrVal.backup in the FEPCluster custom resource.

Specify the object storage for the backup data in pgbackrestParams. Refer to "[2.3.5 Scheduling Backup from Operator](#)" for possible values for pgbackrestParams.

Specify the ConfigMap name created in "[5.10.1.1 Storing CA Files \(Root Certificates\)](#)" for caName.

FEPCluster Custom Resource Example: Only Object Storage Used for Backup Repository

```
apiVersion: fep.fujitsu.io/v2
kind: FEPCluster
metadata:
  ...
spec:
  fepChildCrVal:
    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
  ...
```

If the persistent volume and object storage specified in spec.fepChildCrVal.storage.backupVol are to be used together in the backup repository, specify the object storage setting after "repo2".

If "repo1" is not defined, a permanent volume is automatically designated as the storage destination for the backup volume.

FEPCluster Custom Resource Example: When using object storage and PV

```
...
spec:
  fepChildCrVal:
    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
...

```

When using object storage GCS as a backup repository, specify as follows.

For repoKeySecretName, specify the Secret created in "[5.10.1.2 Storing Repository Key](#)". Also, specify service for gcs-key-type.

FEPCluster Custom Resource Example: When using GCS as a backup repository

```

apiVersion: fep.fujitsu.io/v1
kind: FEPCluster
metadata:
  ...
spec:
  fepChildCrVal:
    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 Disaster Recovery

By using OSS (pgBackRest) functionality to store backup data in object storage, data can be migrated to a database cluster in a different OCP environment.

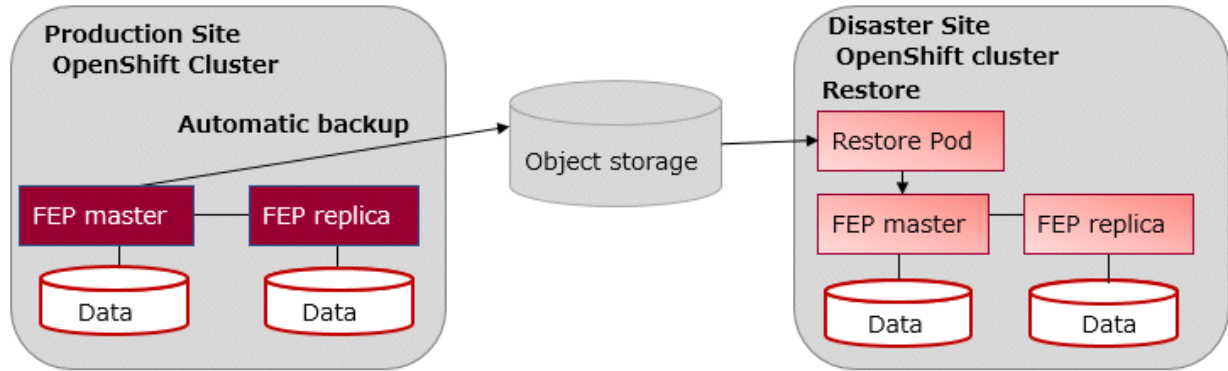
Even if it is difficult to operate in an OCP environment with a database cluster due to a disaster, it is possible to continue operating in a different OCP environment.

5.11.1 Disaster Recovery Prerequisites

The configuration diagram of the pod placement and backup repository, which are prerequisites for the backup feature for performing disaster recovery, is shown below.

In FEPCluster to get a backup, specify the object storage as the backup data storage destination with spec.fepChildCrVal.backup.pgbackrestParams.

Specify object storage that is in an area that is considered safe for the scope of the expected disaster.



Note

The definition of the FEPCluster custom resource is not inherited when performing disaster recovery.

We recommend that you save your production site FEPCluster custom resource definitions in case of a disaster.

5.11.2 Performing Disaster Recovery

Describes the procedure for restoring to an OCP environment different from the restore source using the backup data stored in the object storage.

5.11.2.1 Pre-creation of Resources

5.11.2.1.1 Storing CA Files (Root Certificates)

If you want to use a non-default root certificate for object storage connections, register it in ConfigMap.

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

5.11.2.1.2 Storing Repository Key

When using the parameter (repo-gcs-key) of pgBackRest, register the GCS repository key in Secret.

```
$ oc create secret generic storage-key-secret --from-file=key.json=storage-key.json -n my-namespace
```

5.11.2.2 Defining a FEPCluster Custom Resource

In addition to the FEPCluster settings, specify the Restore settings below.

FEPCluster Custom Resource Example

```
apiVersion: fep.fujitsu.io/v1
kind: FEPCluster
metadata:
  ...
spec:
  fepChildCrVal:
    restore:
      pgbackrestParams: |
        repol-type=s3
        repol-path=/backup/cluster1
        repol-s3-bucket=sample-bucket
        repol-s3-endpoint=s3.ap-northeast-1.amazonaws.com
        repol-s3-region=ap-northeast-1
        repol-storage-ca-file=/pgbackrest/storage-certs/ca.crt
      pgbackrestKeyParams: |
```

```

    repol-s3-key=SAMPLEKEY
    repol-s3-key-secret=SAMPLESECRET
  caName:
  - storage-cacert

```

...

When using object storage GCS as a backup repository, specify as follows.

For repoKeySecretName, specify the Secret created in "[5.11.2.1.2 Storing Repository Key](#)". Also, specify service for gcs-key-type.

```

apiVersion: fep.fujitsu.io/v1
kind: FEPCluster
metadata:
  ...
spec:
  fepChildCrVal:
    backup:
      pgbackrestParams: |
        repol-type=gcs
        repol-path=/backup-ct/test2
        repol-gcs-bucket=dbaas-gcs
        repol-gcs-endpoint=localhost
        repol-storage-ca-file=/pgbackrest/storage-certs/ca.crt
        repol-gcs-key=/pgbackrest/storage-key/key.json
        repol-gcs-key-type=service
      caName:
      - storage-cacert
      repoKeySecretName:
      - storage-key-secret
    ...

```

Setting value

Field	Default	Details
spec.fepChildCrVal.restore		Define when restoring by specifying the backup data stored in the object storage.
spec.fepChildCrVal.restore.pgbackrestParams		Optional " " is fixed, and the following lines specify the parameters to set in pgbackrest.conf. Specify the object storage where the backup data is stored. If you want to use a root certificate other than the default, specify the following: repol-storage-ca-path=/pgbackrest/storage-certs/<file name> Register the CA file in ConfigMap and specify the ConfigMap name in spec.fepChildCrVal.restore.caName.
spec.fepChildCrVal.restore.pgbackrestKeyParams		Optional " " is fixed, and the following lines specify the parameters to set in pgbackrest.conf. The value described by this parameter is masked with *****. Specify the parameter you want to mask, such as a password.
spec.fepChildCrVal.restore.caName		Optional Specify when you use a CA file other than the system default. Specify the name of the created ConfigMap in list format.

Field	Default	Details
		The specified ConfigMap will be mounted in /pgbackrest/storage-certs.
spec.fepChildCrVal.restore.mcSpec.limits	cpu: 200m memory: 300Mi	Optional CPU and memory allocated to the container performing the restore.
spec.fepChildCrVal.restore.mcSpec.requests	cpu: 100m memory: 200Mi	Optional CPU and memory allocated to the container performing the restore.
spec.fepChildCrVal.restore.restoreType	latest	Optional Restore Type (latest or PITR)
spec.fepChildCrVal.restore.restoreRedate		Optional Specify the date to restore when spec.fepChildCrVal.restore.restoreType is "PITR".
spec.fepChildCrVal.restore.restoreRetime		Optional Specify the time to restore when spec.fepChildCrVal.restore.restoreType is "PITR".
spec.fepChildCrVal.restore.image		Optional Image of the container to perform the restore. It is omitted by default. In this case, the URL for image is obtained from the operator container environment.
spec.fepChildCrVal.restore.imagePullPolicy	IfNotPresent	Optional

5.12 Operation of Transparent Data Encryption Using Key Management System

5.12.1 Updating Custom Resource Parameters

When using a newly generated master encryption key in your key management system, update the FEPCluster custom resource `fepChildCrVal.sysTde.tdek.targetKeyId` to the ID of the new master encryption key. The operator will automatically re-enable TDE when this value is updated.

Also, if the credentials for connecting to the key management system are updated and the passphrase of the private key is updated, update the `sslpassphrase` value under `fepChildCrVal.sysTde.tdek.kmsDefinition` of the FEPCluster custom resource. The operator automatically performs a keystore open when this value is updated.

When re-enabling TDE or opening the keystore is completed, the following event will be notified.

```
# When re-enabling TDE
$ kubectl get event
LAST SEEN   TYPE      REASON              OBJECT                                  MESSAGE
164m        Normal    SuccessfulTdeSetMasterKey  fepconfig/<FEPClusterCR名> <namespace>, Successfully
set TDE masterKey

# When re-enabling TDE fails
$ kubectl get event
LAST SEEN   TYPE      REASON              OBJECT                                  MESSAGE
164m        Warning   FailedTdeSetMasterKey  fepconfig/<FEPClusterCR名> <namespace>, Error/Failure
set TDE masterKey
```

If the process fails, review the parameters defined in the FEPCluster custom resource and re-enter the correct values.

If there is an error in the Secret or ConfigMap that stores the private key for TLS communication with the key management system, and there is no correction for the custom resource, use the FEPAAction custom resource described in "5.12.2 Update Credentials", to open the keystore.

5.12.2 Update Credentials

If the credentials of the key management system are updated, update the Secret/ConfigMap values specified under cert in sysTde.tdek.kmsDefinition and apply the FEPAAction custom resource to update the credentials.

Example) Definition example of FEPAAction custom resource

```
apiVersion: fep.fujitsu.io/v1
kind: FEPAAction
metadata:
  name: new-fep-action
spec:
  sysExtraLogging: false
  targetClusterName: nf-131851
  fepAction:
    type: open_tde_masterkey
```

5.12.3 Encrypting a Tablespace

If you create an encrypted tablespace, configure the encryption algorithm in runtime parameters. For example, to create a tablespace named secure_tablespace using AES with a 256-bit key length as the encryption algorithm, define:

```
-- Specify the encryption algorithm for the tablespace to be created below
SET tablespace_encryption_algorithm = 'AES256';
CREATE TABLESPACE secure_tablespace LOCATION '/database/tablespaces/tbspacel';
-- Specify that the tablespace to be created below is not to be encrypted
SET tablespace_encryption_algorithm = 'none';
```

Or

```
CREATE TABLESPACE tbs_tst_new LOCATION '/database/tablespaces/tbspacel' WITH
(tablespace_encryption_algorithm = 'AES256' );
```

Checking for encrypted tablespaces

You can check which tablespaces are encrypted by executing the following SQL.

```
SELECT spcname, spcencalgo FROM pg_tablespace ts, pgx_tablespaces tsx WHERE ts.oid =
tsx.sptablespace;
```

5.12.4 Backup/Restore

In case the FEP cluster is damaged or lost, backups should be made at the following times:

- When the cluster is first created
- When the master encryption key is changed

When you use the FEPRestore custom resource to create a cluster restored from backup, the restored cluster is restored with the master encryption key at the time the backup was taken on the source cluster (where the backup was created from).

When a master encryption key newer than the time of backup is specified in sysTde.tdek.targetKeyId newer than the backup time of the source FEPCluster custom resource, the value is inherited to the restore destination FEPCluster custom resource, and the operator automatically restores the data after the data is restored. Temporarily re-enable TDE with the new master encryption key.

Also, before executing the restore, update the authentication information (sslpasphrase value of sysTde.tdek.kmsDefinition or the value held in the Secret/ConfigMap specified under cert) to the key management system of the source FEPCluster custom resource. please give me. If your authentication information is not updated, you will not be able to connect to the key management service and restore your data.

If parameters other than "sysTde.tdek.targetKeyId" and "sysTde.tdek.kmsDefinition sslpassphrase" under sysTde.tdek are incorrectly updated after building FEPCluster, the key management system cannot be referenced when restoring data. Before executing the restore process, confirm that the correct values are described in the FEPCluster custom resource.

Chapter 6 Maintenance Operations

This chapter describes the maintenance operation after deploying the container.

6.1 Minor Version Upgrade

Minor FEP version upgrade is done by replacing the image in FEPCluster customer resource with a new one. For the procedure, refer to "Minor Version Upgrade" in the Overview.

Update information can be found in the Red Hat catalog to see if a new FEP database server container has been released.

Upgrades are rolling updated, so you can localize downtime, but it is recommended that you avoid running during business hours as connected applications will result in connection errors.



Note

The upgrade process will cause an outage on the cluster for the duration to upgrade both Master and Sync Replica. If there is no Sync Replica in the cluster, the outage is limited to the length of time to upgrade the Master (or actually the failover time required to take another replica been promoted by patroni).

6.2 Cluster Master Switchover

You can switch a master instance to a replica instance in the event of a master instance performance failure or planned node maintenance.

Specify "switchover" for the action type of the FEPAAction CR to update FEPAAction CR.

Equivalent Kubernetes command: `kubectl apply -f <new_spec>`

"switchover" action type expects users to specify the name of the current leader/primary pod that they want to switchover from. Specify the name in the args section under the FEPAAction CR spec as below:

```
spec:
  fepAction:
    args:
      - new-fep-sts-2
    type: switchover
    targetClusterName: new-fep
```

Here, new-fep-sts-2 is the current primary.

Refer to "FEPAAction Custom Resource Parameters" in the Reference for more information on parameters.

6.3 Perform PITR and the Latest Backup Restore from Operator

It can be used to restore a database to a specific location due to an application failure or to prepare a duplicate database for production.

Restore process can restore data by creating a CR (FEPRestore CR) for the restore as follows:

`oc create -f [Custom Resource Files]`

Example)

```
$oc create -f config/samples/postgres_v1_restore.yaml
```

There are two methods of restoring: restoring data to an existing FEPCluster or restoring data to a new FEPCluster.

When restoring to an existing FEPCluster, information such as the FEPCluster name, IP address, and various settings remain the same.

If you restore to a new FEPCluster, the FEPCluster name is the one you specified in CR and the new IP address is also given. If the setting value is not specified, the new cluster will inherit the settings from the restore source cluster, but you can change the settings to create a new cluster by specifying them in CR.

6.3.1 Setting Item

Refer to "FEP Restore Custom Resource Parameters" in the Reference for the items to be set in a custom resource file.

6.3.2 After Restore

Switching connections to the new cluster

The restore creates a new FEPCluster. If necessary, you need to set up Pgpool-II and change the access point of the application to the new cluster or the new Pgpool-II.

Backup data of the destination cluster

PITR restores to the pre-restore time are not possible, because the backup of the destination cluster begins after the restore completes.

6.4 Major Version Upgrade

Describes the procedure for upgrading the major version of the operator and FEP container.

A major version upgrade of a FEP builds a new major version of the FEP in the same Namespace as the previous major version of the FEP. At this time, by defining the "spec.fepChildCrVal.upgrade" field in FEPClusterCR, the operator creates the upgrade execution container. The upgrade execution container uses the previous version of FEP Cluster specified in "spec.fepChildCrVal.upgrade.sourceCluster" as the data source FEPCluster and migrates the data to the newly created FEPCluster.

6.4.1 Pre-work on the Data Source FEP Cluster

Stop the running business application before executing the major version upgrade.

Next, edit "spec.fepChildCrVal.customPgHba" of the data source FEPCluster Custom Resource to allow the connection of the upgrade execution container.

The addresses that are allowed to connect are specified as follows:

```
<fep>-upgrade-pod.<fep>- upgrade-headless-svc.<namespace>.svc.cluster.local
```

<fep> specifies the name of the newly created FEPCluster Custom Resource.

The authentication method can be either trust/md5/cert.

Example of Editing a FEPCluster Custom Resource in a Data Source:

```
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 Operator Upgrade

Describes the instructions for upgrading the operator.



Note

After an operator upgrade, any custom resource configuration changes you defined in the previous version are not reflected in the container.

6.4.2.1 Uninstalling the Old Operator

Uninstall the old operator.

Select "Uninstall Operator" from "Operators">"Installed Operators">"FUJITSU Enterprise Postgres <Old version> Operator"> Actions.

6.4.2.2 Installing a New Version of the Operator

Refer to "[Chapter 3 Operator Installation](#)" to install the new version of the operator.

6.4.3 Major Version Upgrade of FEP

6.4.3.1 Creating a New FEPCluster CR

Refer to the Reference to define a new major version of the FEPCluster custom resource. At this time, allow the running upgrade container to connect as you did in "[6.4.1 Pre-work on the Data Source FEP Cluster](#)".

In addition, a major version upgrade of FEP is performed by defining the "spec.fepChildCrVal.upgrade" field, as in the following example of defining a FEPCluster custom resource.

The upgrade execution container uses PV to store dump files retrieved from the FEPCluster of the data source.

If you have not enabled the automatic PV provisioning feature in your Kubernetes environment, create a PV for the upgrade in addition to the new PV for the FEPCluster before creating the FEPCluster custom resource.

Also, edit "spec.fepChildCrVal.customPgHba" to allow the connection of the upgrade execution container, as in "[6.4.1 Pre-work on the Data Source FEP Cluster](#)".

Example of Defining a FEPCluster Custom Resource to Perform an Upgrade:

```
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 Custom Resource Fields "spec.fepChildCrVal.upgrade"

Field	Default	Details
spec.fepChildCrVal.upgrade		Optional When this field is defined, a major version upgrade is performed.

Field	Default	Details
		However, if spec.fepChildCrVal.restore is defined, the FEPCluster build stops.
spec.fepChildCrVal.upgrade.sourceCluster		Specify the FEPCluster CR name of the data migration source. Be sure to specify spec.fepChildCrVal.upgrade when defining it.
spec.fepChildCrVal.upgrade.mcSpec.limits	cpu: 200m memory: 300Mi	Optional Specify the maximum number of resources allocated to the upgrade execution container.
spec.fepChildCrVal.upgrade.mcSpec.requests	cpu: 100m memory: 200Mi	Optional Specify the lower limit of resources allocated to the upgrade execution container.
spec.fepChildCrVal.upgrade.image		Optional If omitted, the URL of the image is obtained from the operator container environment.
spec.fepChildCrVal.upgrade.imagePullPolicy	IfNotPresent	Optional Specify the pull policy for the container image. - Always - IfNotPresent - Never
spec.fepChildCrVal.upgrade.source.pgAdminTls.certificateName		Optional If the data source FEPCluster used "cert" as the authentication method for the Upgrade Execution Container, use the secret certificate that defines spec.fepChildCrVal.sysUsers.pgAdminTls.certificateName for the data source FEPCluster. If the above parameter is not defined, it points to the Kubernetes TLS secret containing the certificate of the Postgres user "postgres" in the data source. Refer to "4.5.1 Manual Certificate Management" for information about creating secrets.
spec.fepChildCrVal.upgrade.destination.pgAdminTls.certificateName		Optional

Field	Default	Details
		<p>If the newly created FEPCluster used the "cert" authentication method for the running upgrade container, use the secret certificate that defines the spec.fepChildCrVal.sysUsers.pgAdminTls.certificateName of the newly created FEPCluster.</p> <p>If the above parameter is not defined, it points to the Kubernetes TLS secret containing the certificate of the newly created Postgres user "postgres".</p> <p>Refer to "4.5.1 Manual Certificate Management" for information about creating secrets.</p>
spec.fepChildCrVal.upgrade.storage		<p>Optional</p> <p>Defines storage for storing dump files.</p>
spec.fepChildCrVal.upgrade.storage.storageClass		<p>Optional</p> <p>If omitted, the default storage class of the operating environment will be used.</p>
spec.fepChildCrVal.upgrade.storage.size	2Gi	<p>Optional</p> <p>Specify the size of the storage to store the dump file.</p>
spec.fepChildCrVal.upgrade.storage.accessModes	ReadWriteOnce	<p>Optional</p> <p>Storage access mode for storing dump files</p> <p>As an array of access modes.</p> <p>e.g. [ReadWriteMany]</p> <p>If omitted, it is treated as [ReadWriteOnce].</p>



Note

Connect to the database and run the following SQL to check the size of the database in advance:

```
$ SELECT pg_size_pretty(sum(pg_database_size(datname))) AS dbsize FROM pg_database;
```

Since the pg_dumpall command used in the upgrade execution container outputs the database data as an SQL command, the file actually created is as follows.

For example, the integer type 2147483647 is 4 bytes for database data.

However, this is 10 bytes because SQL commands output them as strings. Therefore, make sure that the storage (PV) for dump files has sufficient disk space.

6.4.3.2 Verifying FEP Major Upgrade Complete

If you migrate your data to the new FEPCluster and the FEP major version upgrade is successful, the following event will be output:

```
$ kubectl get event
LAST SEEN   TYPE      REASON              OBJECT                                          MESSAGE
164m       Normal    SuccessfulFepUpgrade fepupgrade/<Name of the new FEPClusterCR> <namespace>,
Successfully FEP Upgrade
```

In addition, the following annotation will be added to YAML in FEPClusterCR:

```
apiVersion: fep.fujitsu.io/v2
kind: FEPCluster
metadata:
  annotations:
    FEPUpgradeDone: true
...
name: destination-fep-cluster
namespace: my-namespace
spec:
...
```



Note

When a major upgrade of FEP fails, an event similar to the following is output:

```
$ kubectl get event
LAST SEEN   TYPE      REASON              OBJECT                                          MESSAGE
164m       Warning    FailedFepUpgrade    fepupgrade/<Name of the new FEPClusterCR> <namespace>, Error/
Failure in FEP Upgrade
```

Obtain the Kubernetes resource information listed in the OBJECT column, review the output messages, and then recreate the new FEPCluster custom resource.

```
$ kubectl describe fepupgrade/<Name of the new FEPClusterCR>
```

6.4.4 Updating Each Custom Resource

Describes the procedures for each custom resource used to operate the FEPCluster for the data source after the major FEP upgrade is complete.

After this process is complete, resume the suspended business applications.

6.4.4.1 Removing a FEPClusterCR for a Data Source

Delete the FEPCluster for the data source.

For the Openshift GUI console:

From "Operators" > "Installed Operators" > "FUJITSU Enterprise Postgres < New version > Operator" > "FEPCluster" > "FEPCluster name to delete" > Actions, select "Delete FEPCluster".

6.4.4.2 FEPPgpool2

Re-create FEPPgpool2 to match the version of the client with the version of the upgraded FEP.

6.4.4.3 FEPExporter Built in Standalone Mode

Edit the FEPExporter custom resource "spec.fepExporter.fepClusterList" to specify the new version of the FEPCluster custom resource.

Refer to "FEPExporter Custom Resource" in the Reference for more information about the parameters.

6.5 Assigned Resources for Operator Containers

The following resources are allocated by default to the operator containers provided by this product.

```
resources:
limits:
  cpu: 2
  memory: 1536Mi
requests:
  cpu: 500m
  memory: 768Mi
```

If there is only one FEPCluster custom resource managed by an operator, it can be operated with the resource assigned by default. However, when deploying and operating multiple FEPCluster custom resources, change the assigned resource of the operator container.



Note

If you have changed the resource, the resource value will revert to the default value after the operator version upgrade. Therefore, change the resource again after upgrading the operator.

6.5.1 How to Change Assigned Resources

Describes how to change the resources assigned to an operator container.

When updating resources assigned to an operator container, the operator container is recreated. At this time, the operation of already built containers such as FEPCluster will not stop.

How you change the allocated resources depends on how the operator was installed.

6.5.1.1 When installing using OperatorHub

If you are using an operator installed from OperatorHub To change the resources assigned to the operator container, edit the ClusterServiceVersion (CSV).

Editing the CSV "spec.install.spec.deployments[0].spec.template.spec.containers[0].resources" will recreate the operator container and apply the specified resources.

When editing CSV from the OCP GUI console

Click [Installed Operators] in the menu item under Operators and select the installed operator. On the [YAML] tab, edit the specified part of the allocation resource and click [Save].

```
675 vendor: Fujitsu
676 spec:
677   containers:
678     - resources:
679       limits:
680         cpu: '2'
681         memory: 3072Mi
682       requests:
683         cpu: 500m
684         memory: 768Mi
685       name: fep-ansible-operator
686       livenessProbe:
687         failureThreshold: 10
```

When editing CSV from the CUI console using the OC client

Check the CSV name of the installed operator with the "oc get" command.

```
$ oc get csv
NAME                                     DISPLAY                                VERSION  REPLACES  PHASE
fujitsu-enterprise-postgres-operator.v4.1.5  FUJITSU Enterprise Postgres Operator
4.1.5                                     Succeeded
```

Edit the CSV with the "oc edit" command.

```
$ oc edit csv fujitsu-enterprise-postgres-operator.v4.1.5
```

6.5.1.2 When installing using Helm Chart or RancherUI

If the operator is installed using Helm Chart or RancherUI, edit the deployment of the operator container to change the resources assigned to the operator container.

Editing the Deployment's "spec.template.spec.containers[0].resources" will recreate the operator container and apply the specified resources.

Edit the Deployment "fep-ansible-operator" with the "kubectl edit" command.

```
$ 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
```

Chapter 7 Abnormality

This chapter describes the actions to take when an error occurs in the database or an application, while FEP is operating.

Depending on the type of error, recover from the backed-up material, reserve capacity, check the operator log, and check the FEP log.

7.1 Handling of Data Abnormalities

Recover the database cluster from the backup immediately prior to failure in any of the following cases:

- A hardware failure occurs on the data storage disk or the backup data storage disk.
- If the data on the disk is logically corrupted and the database does not work correctly
- Data corruption caused by user error

Refer to "[6.3 Perform PITR and the Latest Backup Restore from Operator](#)" for restore instructions.

7.2 Handling when the Capacity of the Data Storage Destination or Transaction Log Storage Destination is Insufficient

If you run out of space in the data storage location, first check if there are any unnecessary files on the disk, and then delete them so that you can continue working.

If deleting unnecessary files does not solve the problem, you may need to migrate the data to a larger disk.

Use a backup restore to migrate data.

7.3 What to do when the Capacity of the Backup Data Storage Area is Insufficient

If you run out of space in the backup data destination, first check the disk for unnecessary files, and then delete the unnecessary files. Or reduce the backup retention generation.

7.4 Handling Access Abnormalities When Instance Shutdown Fails

If an instance fails to start or stop, refer to the Operator log and the FEP log to determine the cause.

For checking the operator log and the FEP log, refer to "[7.5 Collection of Failure Investigation Information](#)".

7.5 Collection of Failure Investigation Information

If the cause of the trouble that occurred during the construction or operation of the environment is not identified, information for the initial investigation is collected.

I will explain how to collect information for the initial investigation.

- Product log
- Operator log

Product log

FEP log

Get into the container and collect the log.

The log location is specified by log_directory in the custom resource FEP Clusterspec.startupValues.customPgParam parameter. The default is/database/log.

Pgpool-II log

Get into the container and collect the log.

The log location is /var/log/pgpool/pool.log.

Operator log

Check the operator log as follows.

Verification Example

```
$oc get po
NAME                                READY   STATUS    RESTARTS   AGE
fep-ansible-operator-7dc5fd9bf7-4  1/1     Running   0           20m
```

How to check the log

```
$oc logs pod fep-ansible-operator-7dc5fd9bf7-4 smzk -c manager
```

The log will be output to the console. Please check the file output by redirection.

Appendix A Quantitative Values and Limitations

A.1 Quantitative Values

Refer to the FUJITSU Software Enterprise Postgres Installation and Setup Guide for Server.

A.2 Limitations

Note

If you log in to a container and edit the configuration file directly, restarting the container may undo your changes.

If you want to change the settings, modify the custom resource files as described in "[5.2 Configuration Change](#)" and reapply. Depending on the parameters to be changed, the container may be redeployed. Refer to "[5.2 Configuration Change](#)" for details of the parameters.

Unavailable FEP features

Since FEP server container is based on other components (like UBI and Patroni), there are certain limitations that doesn't allow it to be 100% functionally capable to VM based server instance. The known limitations are as below.

No	Limitation	Reason for Limitation	Description
1	No Support for JIT	Since UBI8 is not having requisite LLVM libraries	It is not possible to enable JIT in postgresql.conf. Impact for the customer is that they are not able to achieve maximum performance capabilities on given CPU and memory
2	FEP parallelism improvements	Since UBI8 is not hosting dstat binaries	FEP parallelism improvement is to restrict number of parallel workers in case the CPU is already busy because of other tasks/processes. It is unlikely to have too much impact on FEP container, since container is running only one process.
3	Crypto Express cards are not supported	IBM LinuxOne doesn't support CryptoExpress cards in Openshift container platform at this stage.	FEP TDEz extension cannot be used on LinuxOne Openshift environment. However, User can still use TDE on both LinuxOne Openshift environment as well as Azure (x86) Openshift environment.
4	No Support for Oracle foreign data wrapper	Oracle foreign data wrapper has dependency on Instant Client package, which is not available.	Oracle InstantClient package is not redistributed by FUJITSU Enterprise Postgres leading to this limitation. The functionality of Oracle Foreign data wrapper is not available to FUJITSU Enterprise Postgres on Openshift environment.

Fixed parameter

Some parameters cannot be changed. Refer to "[2.3.5.2 Parameters that cannot be Set](#)".

FEP features that needs to be set when using

Refer to "[2.3.7 FEP Unique Feature Enabled by Default](#)".

Appendix B Adding Custom Annotations to FEPCluster Pods using Operator

This section describes instructions for adding custom annotations to a FEPCluster pod.

1. In YAML view of the Create FEPCluster section, add custom annotations as below and then click on Create.

Project: fep14-install-test

FUJITSU Enterprise Postgres 14 Operator > Create FEPCluster

Create FEPCluster

Create by manually entering YAML or JSON definitions, or by dragging and dropping a file into the editor.

Configure via: ☐ Form view ☒ YAML view

```
1 apiVersion: fep.fujitsu.io/v2
2 kind: FEPCluster
3 metadata:
4   name: new-fep
5   namespace: fep14-install-test
6 spec:
7   fep:
8     customAnnotations:
9       allDeployments:
10        annotation1: value1
11        annotation2: value2
12   forceSsl: true
13   image:
14     pullPolicy: IfNotPresent
15   instances: 1
16   mcSpec:
17     limits:
18       cpu: 500m
19       memory: 700Mi
20     requests:
21       cpu: 200m
22       memory: 512Mi
23   podAntiAffinity: false
24   podDisruptionBudget: false
25   servicePort: 27500
26   syncNode: 'off'
27   sysExtraLogging: false
28   fepChildCrVal:
29     backup:
30       image:
31         pullPolicy: IfNotPresent
32       mcSpec:
33         limits:
34           cpu: 0.2
```

[Alt + F1 Accessibility help](#) | [View shortcuts](#)

[Create](#) [Cancel](#) [Download](#)

- Both the Statefulset and its resulting pods will be annotated with your provided annotations: archivalVol and backupVol must be ReadWriteMany.

The screenshot shows the Red Hat OpenShift console interface. On the left is a sidebar with navigation menus: Administrator, Home, Operators, Workloads, and others. The 'Stateful Sets' menu item is highlighted. The main panel displays the 'Stateful Set Details' for 'new-fep-with-cust-anno-sts' in the 'install-test' project. The 'YAML' tab is active, showing the following configuration:

```
1 kind: StatefulSet
2 apiVersion: apps/v1
3 metadata:
4   annotations:
5     annotation1: value1
6     annotation2: value2
7   statusCheckAt: 'Tue Sep 7 15:23:31 UTC 2021'
8   selflink: >-
9     /apis/apps/v1/namespaces/install-test/statefulsets/new-fep-with-cust-anno-sts
10  resourceVersion: '147317819'
11  name: new-fep-with-cust-anno-sts
12  uid: 269c6888-434d-4bde-b1d4-832c16ad521c
13  creationTimestamp: '2021-09-07T15:20:55Z'
14  generation: 1
15  managedFields:
16    - manager: OpenAPI-Generator
17      operation: Update
18      apiVersion: apps/v1
19      time: '2021-09-07T15:20:55Z'
20      fieldsType: FieldsV1
21      fieldsV1:
22        'f:metadata':
23          'f:annotations':
```

At the bottom of the editor are 'Save', 'Reload', and 'Cancel' buttons, and a 'Download' button on the right.

This screenshot shows the same Red Hat OpenShift console interface, but with a different YAML configuration for the 'new-fep-with-cust-anno-sts' StatefulSet. The 'YAML' tab is active, showing the following configuration:

```
535   name: new-fep-with-cust-anno
536   uid: 27037431-46a9-49eb-a723-3b8c2e8aab49
537   labels:
538     app: new-fep-with-cust-anno-sts
539     fepclustername: new-fep-with-cust-anno
540   spec:
541     replicas: 1
542     selector:
543       matchLabels:
544         app: new-fep-with-cust-anno-sts
545         fepclustername: new-fep-with-cust-anno
546     template:
547       metadata:
548         creationTimestamp: null
549       labels:
550         app: new-fep-with-cust-anno-sts
551         fepclustername: new-fep-with-cust-anno
552       annotations:
553         annotation1: value1
554         annotation2: value2
555       spec:
556         restartPolicy: Always
557         serviceAccountName: new-fep-with-cust-anno-sa
```

The interface includes the same sidebar and top navigation as the previous screenshot, with 'Stateful Sets' selected in the sidebar.

Appendix C Utilize Shared Storage

Explains how to build a FEPCluster when using shared storage.

Use a disk where PV accessModes can specify ReadWriteMany.

This chapter shows an example of using NFS as PV in static provisioning.

C.1 Creating a StorageClass

Create a StorageClass.

In the OCP WebGUI screen, click "StorageClass" in the main menu "Storage", then press "Create Storage Class" > "Edit YAML" and edit YAML to create the StorageClass.

If you are using the CLI, create a yaml file and create a StorageClass with the following command:

```
$ oc create -f <file_name>.yaml
```

YAML definitions are created with reference to the following samples.

Example)

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: < StorageClass Name >
provisioner: kubernetes.io/no-provisioner
reclaimPolicy: Delete
volumeBindingMode: WaitForFirstConsumer
```

C.2 Creating a PersistentVolume

Create as many PersistentVolumes (PV) as you need.

On the Web GUI screen, click "PersistentVolumes" in the main menu "Storage", click "Create PersistentVolume", and edit YAML to create PV.

If you are using the CLI, create a yaml file and create a PV using the following command:

```
$ oc create -f <file_name>.yaml
```

YAML definitions are created with reference to the following samples.

The StorageClass name specifies the StorageClass created in "[C.1 Creating a StorageClass](#)".

Assign a different NFS directory for each PV.

In addition, accessModes is ReadWriteMany.

Example)

```
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: < StorageClass name created in "C.1 Creating a StorageClass">
```

C.3 Creating FEPCluster

Specifies that ReadWriteMany PV is used in the YAML definition in step 4 of "[4.1 Deploying FEPCluster using Operator](#)".

In spec.fepChildCRVal.storage, specify the StorageClass and AccessModes of the PV created in "[C.2 Creating a PersistentVolume](#)".

The "spec.fepChildCRVal.storage.<Volume Type>.size" should be less than or equal to the PV allocated.

Example) Using PV created by archivewalVol and backupVol

```
apiVersion: fep.fujitsu.io/v2
kind: FEPCluster
metadata:
  name: t3-fep
spec:
  ~ Suppress ~
  fepChildCrVal:
    storage:
      archivewalVol:
        size: < Capacity Required ex. 8Gi >
        storageClass: <StorageClass name created in C.1 Creating a StorageClass >
        accessModes:
          - "ReadWriteMany"
      backupVol:
        size: < Capacity Required ex. 8Gi >
        storageClass: <StorageClass name created in C.1 Creating a StorageClass >
        accessModes:
          - "ReadWriteMany"
  ~ Suppress ~
```