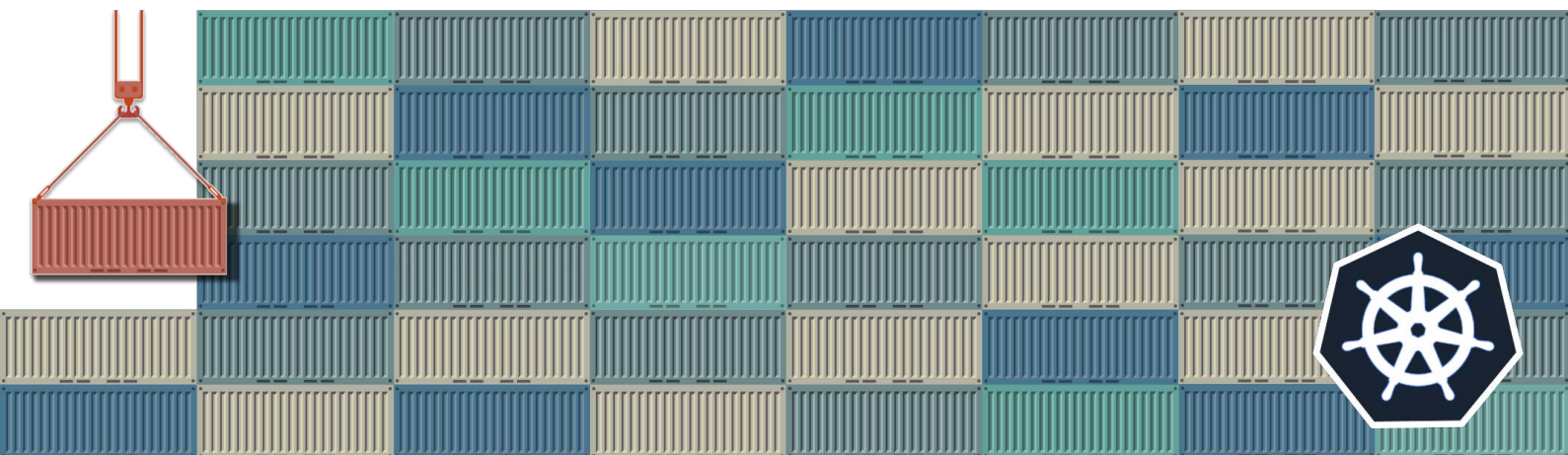


FUJITSU Enterprise Postgres 13 for Kubernetes



User's Guide

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 9.0: June 2023
Edition 8.0: October 2022
Edition 7.0: September 2022
Edition 6.0: June 2022

Edition 5.0: March 2022
Edition 4.0: December 2021
Edition 3.0: November 2021
Edition 2.0: October 2021
Edition 1.0: September 2021

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.....	1
Chapter 2 Overview of Operator Design.....	2
2.1 Design Task.....	2
2.2 System Configuration Design.....	2
2.2.1 Server Configuration.....	2
2.2.2 User Account.....	4
2.2.3 Basic Information of the Container.....	4
2.3 Design Perspective for Each Feature.....	6
2.3.1 Deployment.....	7
2.3.2 High Availability.....	7
2.3.3 Configurable Volume per Cluster.....	8
2.3.4 Deploying Pgpool-II and Connect to FEPCluster from Operator.....	10
2.3.5 Scheduling Backup from Operator.....	12
2.3.5.1 Important Setting Items.....	13
2.3.5.2 Parameters that cannot be Set.....	13
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.....	16
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.....	17
2.3.9.1 Change to FEPCluster CR - autoscale.....	17
Chapter 3 Operator Installation.....	18
3.1 Pre-requisite.....	18
3.2 Deploying Operator.....	19
3.3 Implement Collaborative Monitoring Tools.....	21
Chapter 4 Deployment Container.....	23
4.1 Deploying FEPCluster using Operator.....	23
4.2 Deploy a Highly Available FEPCluster.....	28
4.3 Deploying FEPEXporter.....	30
4.4 FEPEXporter in Standalone Mode.....	31
4.5 Configuration FEP to Perform MTLs.....	33
4.5.1 Manual Certificate Management.....	34
4.5.2 Automatic Certificate Management.....	37
4.5.3 Deploy FEPCluster with MTLs support.....	40
4.5.4 Configurable Parameters.....	47
4.6 Replication Slots.....	49
4.6.1 Setting Up Logical Replication using MTLs.....	49
Chapter 5 Post-Deployment Operations.....	53
5.1 Configuration Change.....	53
5.2 FEPCluster Resource Change.....	54
5.2.1 Changing CPU and Memory Allocation Resources.....	54
5.2.2 Resizing PVCs.....	54
5.3 FEPPGPool2 Configuration Change.....	55
5.4 Scheduling Backup from Operator.....	56

5.5 Configure MTLS Setting.....	57
5.5.1 Certification Rotation.....	57
5.6 Monitoring.....	57
5.6.1 Monitoring FEP Operator and Operands.....	58
5.6.2 Monitoring FEP Server.....	59
5.6.2.1 Architecture.....	59
5.6.2.2 Default Server Metrics Monitoring.....	60
5.6.2.3 Default Alerts.....	62
5.6.2.4 Graphical user interface.....	62
5.6.3 Monitoring FEP Backup.....	62
5.6.3.1 pgbackrest_info_backup view.....	63
5.6.4 Monitoring FEP PGPool2.....	63
5.6.4.1 pgpool2_stat_load_balance view.....	63
5.6.4.2 pgpool2_stat_conn_pool view.....	64
5.6.4.3 pgpool2_stat_sql_command view.....	64
5.7 Event Notification.....	64
5.7.1 Events raised.....	65
5.7.2 Viewing the custom events.....	65
5.8 Scaling Replicas.....	66
5.8.1 Auto Scale Out.....	66
5.8.2 Manual Scale In/Out.....	66
Chapter 6 Maintenance Operations.....	68
6.1 Minor Version Upgrade.....	68
6.2 Cluster Master Switchover.....	68
6.3 Perform PITR and the Latest Backup Restore from Operator.....	68
6.3.1 Setting Item.....	69
6.3.2 After Restore.....	69
6.4 Major Version Upgrade.....	69
6.4.1 Install a New Version of the Operator.....	69
6.4.2 Deploy a New Version of a Container.....	69
6.4.3 Migrating Data Between Containers.....	69
6.4.4 Removing Old Containers.....	70
6.4.5 Uninstalling Old Operators.....	70
6.5 Assigned Resources for Operator Containers.....	71
6.5.1 How to Change Assigned Resources.....	71
Chapter 7 Abnormality.....	73
7.1 Handling of Data Abnormalities.....	73
7.2 Handling when the Capacity of the Data Storage Destination or Transaction Log Storage Destination is Insufficient.....	73
7.3 What to do when the Capacity of the Backup Data Storage Area is Insufficient.....	73
7.4 Handling Access Abnormalities When Instance Shutdown Fails.....	73
7.5 Collection of Failure Investigation Information.....	73
Appendix A Quantitative Values and Limitations.....	75
A.1 Quantitative Values.....	75
A.2 Limitations.....	75
Appendix B Adding Custom Annotations to FEPCluster Pods using Operator.....	76
Appendix C Utilize Shared Storage.....	78
C.1 Creating a StorageClass.....	78
C.2 Creating a PersistentVolume.....	78
C.3 Creating FEPCluster.....	79

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	13.8	To provide server capabilities
3	Patroni	2.1.0	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

1.3 Supported Platform

It supports running on the following platforms.

No	Platform	Version
1	OpenShift Container Platform	4.10, 4.11, 4.12

Supports storage supported by OpenShift.

However, backup and archive WAL volumes require shared storage, such as NFS.

1.4 Collaboration Tool

Supports integration with the following tools for monitoring and alerting.

No	Tool	Version	How to obtain
1	Prometheus	The version installed OpenShift	Preinstalled with OpenShift
2	AlertManager		
3	Grafana	Grafana v7.5.17 and later	Provided by OperatorHub (v4.7.1 and later)

Chapter 2 Overview of Operator Design

This chapter describes an overview of the operator design.

2.1 Design Task

This section describes the operation of FEP.

First, determine the configuration. You then design each feature and deploy the container. You can use FEP features immediately after deployment.

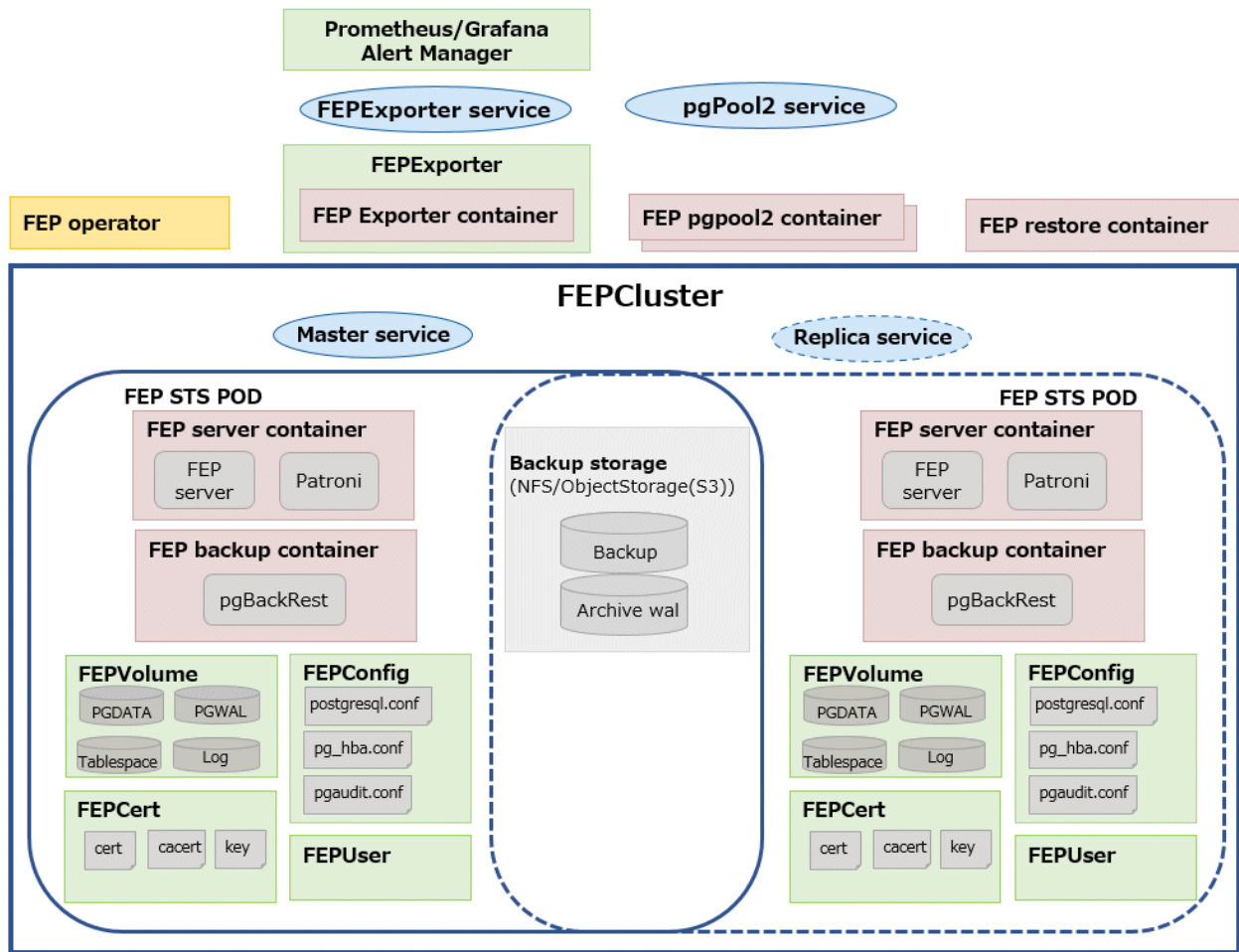
Task	Design required to operate FEP	Where to find
FEP setup	Required.	2.3.1 Deployment
High availability configuration	Optional. (When checking or changing the behavior of high availability. However, even by default, constant high availability operation is possible.)	2.3.2 High Availability
Volume settings	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 restore container	A container that performs the restore operation. Temporarily created during a restore operation.
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 Exporter container	A container that exposes http/https endpoint for monitoring stats scraping.
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.

Configuration server type	Description
FEP Cert	Child CR for system certificates.
FEP User	Child CR for database users.
FEP Action	CR for performing actions.
FEP Exporter	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.
Fep exporter service	A service to scrape metrics from all FEP Cluster 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-13-server:*OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	13	
<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-13-server:ubi8-13-1.0
 - fujitsu-enterprise-postgres-13-server:ubi8-13-1.0-amd64
 - fujitsu-enterprise-postgres-13-server:ubi8-13-1.0-s390x

FEP backup container

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

fujitsu-enterprise-postgres-13-backup: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	13	
<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-13-backup:ubi8-13-1.0
 - fujitsu-enterprise-postgres-13-backup:ubi8-13-1.0-amd64
 - fujitsu-enterprise-postgres-13-backup:ubi8-13-1.0-s390x

FEP restore container

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

fujitsu-enterprise-postgres-13-restore: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	13	
<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-13-restore:ubi8-13-1.0
 - fujitsu-enterprise-postgres-13-restore:ubi8-13-1.0-amd64
 - fujitsu-enterprise-postgres-13-restore:ubi8-13-1.0-s390x

FEP pgpool2 container

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

fujitsu-enterprise-postgres-13-pgpool2: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	13	
<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-13-pgpool2:ubi8-13-1.0
- fujitsu-enterprise-postgres-13-pgpool2:ubi8-13-1.0-amd64
- fujitsu-enterprise-postgres-13-pgpool2:ubi8-13-1.0-s390x

FEP Exporter container

FEP Exporter container as for FEP server containers.

fujitsu-enterprise-postgres-13-exporter: *OS-FEPBaseVersion-MajorVersion.MinorVersion-ARCH*

For each *Version*, specify the following:

Field	Values	Description
<i>OS</i>	ubi8	
<i>FEPBaseVersion</i>	13	
<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-13-exporter:ubi8-13-1.0
- fujitsu-enterprise-postgres-13-exporter:ubi8-13-1.0-amd64
- fujitsu-enterprise-postgres-13-exporter:ubi8-13-1.0-s390x

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	-

Spec	Example	Comment
	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 `fepChildCrVal` in the FEPCluster custom resource.



- 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
archivalVol	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

Field	Mandatory	Sub-Field	Default	Description
				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
		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

Field	Mandatory	Sub-Field	Default	Description
		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.

Field	Default	Details
apiVersion	fep.fujitsu.io/v1	Fixed
kind	FEPPgpool2	Fixed
metadata.name	-	List the name of the FEP pgpool2 container.
metadata.namespace	-	Specify the namespace of the environment where you want to deploy the operator.
spec.image	-	Specifies the FEP pgpool2 container image to provide.
spec.count	2	List the number of FEP pgpool2 containers to create.
spec.serviceport	9999	Describes the TCP port for connecting to the FEP pgpool2 container.
spec.statusport	9898	Identifies the TCP port for connecting to the PCP process.
spec.limits.cpu	400m	List the number of CPUs (restriction) to allocate to resources.limits.cpu.
spec.limits.memory	512Mi	Specifies the memory size (restriction) to allocate to resources.limits.memory.
spec.requests.cpu	200m	List the number of CPUs (request) to allocate to resources.requests.cpu.
spec.requests.memory	256Mi	Specifies the memory size (request) to allocate to resources.requests.memory.
spec.fepclustername	new-fep	Enter the FEPCluster name to connect to.
spec.customhba	-	If you want to use pool_hba.conf, describe what pool_hba.conf should contain from the line below.
spec.customparams	" "	" " and the Pgpool-II parameters. Refer to " Pgpool-II parameters " for detail.
spec.custompcp	" "	If you use the pcp command, " " and the contents of pcp.conf from the line below.
spec.customsslkey	" "	If you want to do it, " " and the Beethoven key content in the line below.
spec.customsslcert	" "	If you want to do it, " " and the contents of the public x 509 certificate from the line below.
spec.customsslca	" "	If you want to do it, " " and the following lines describe the contents of the CA root certificate in PEM format.
spec.customlogsize	100 Mi	Specifies the persistent volume size for log output.

Field	Default	Details
spec.storageclassname	-	Specifies the storage class for log output.

Pgpool-II parameters

The parameters that can be specified are shown in the table below. For details on the parameters, refer to the Pgpool-II manual.

Category	Parameter name (Specified format)	Restart required after change
Connection settings	listen_addresses (string)	Y
	pcp_listen_addresses (string)	Y
	num_init_children (integer)	Y
	reserved_connections (integer)	Y
Authentication settings	enable_pool_hba (boolean)	
	allow_clear_text_frontend_auth (boolean)	
	authentication_timeout (integer)	
Backend settings	backend_weight0 (floating point)	
	backend_weight1 (floating point)	
	backend_flag0	
	backend_flag1	
Connection pooling	connection_cache (boolean)	Y
	max_pool (integer)	Y
	listen_backlog_multiplier (integer)	Y
	serialize_accept (boolean)	Y
	child_life_time (integer)	Y
	client_idle_limit (integer)	
	child_max_connections (integer)	Y
	connection_life_time (integer)	Y
reset_query_list (string)		
Error reporting and log acquisition	client_min_messages (enum)	
	log_min_messages (enum)	
	log_statement (boolean)	
	log_per_node_statement (boolean)	
	log_client_messages (boolean)	
	log_hostname (boolean)	
	log_connections (boolean)	
	log_error_verbosity (enum)	
log_line_prefix (string)		
Load sharing settings	load_balance_mode (boolean)	Y
	ignore_leading_white_space (boolean)	
	white_function_list (string)	
	black_function_list (string)	

Category	Parameter name (Specified format)	Restart required after change
	black_query_pattern_list (string)	
	database_redirect_preference_list (string)	
	app_name_redirect_preference_list (string)	
	allow_sql_comments (boolean)	
	disable_load_balance_on_write (string)	Y
	statement_level_load_balance (boolean)	
Health check	connect_timeout (integer)	
Streaming replication check	sr_check_period (integer)	
	sr_check_user (string)	
	sr_check_password (string)	
	sr_check_database (string)	
	delay_threshold (integer)	
	log_standby_delay (string)	
Secure Socket Layer (SSL)	ssl (boolean)	Y
	ssl_ciphers (string)	Y
	ssl_prefer_server_ciphers (boolean)	Y
	ssl_ecdh_curve (string)	Y
	ssl_dh_params_file (string)	Y
Other parameters	relcache_expire (integer)	Y
	relcache_size (integer)	Y
	enable_shared_relcache (boolean)	Y
	relcache_query_target (enum)	
	check_temp_table (enum)	
	check_unlogged_table (boolean)	

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: FEPBackup
metadata:
  name: fepcluster-backup
spec:
  pgBackrestParams: |
    # define custom pgbackrest.conf parameters below to override defaults.
    [global]
    repo-retention-full = 30
    repo-retention-full-type = time
  ...

```

2.3.5.2 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
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)
Delta Option (--delta)	Restore or Backup with Checksum	For new restores only
Lock Path Option (--lock-path)	Path where the lock file is stored	To use internal fixed values

Parameter	Overview of parameters	Reason	
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	
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	
Azure Repository Account Option (--repo-azure-account)	Azure account used to store the repository	Azure storage is not supported	
Azure Repository TLS CA File Option (--repo-azure-ca-file)	Use a non-default CA file for the Azure Repository TLS CA file system		
Azure Respository TLS CA Path Option (--repo-azure-ca-path)	Use non-default CA path for Azure Repository TLS CA path system		
Azure Repository Container Option (--repo-azure-container)	Azure repository container.Azure container used to store the repository.		
Azure Repository Host Option (--repo-azure-host)	Azure Repository Host		
Azure Repository Key Option (--repo-azure-key)	Azure Repository Shared Key or Shared Access Signature		
Azure Repository Key Type Option (--repo-azure-key-type)	Azure Repository Key Type		
Azure Repository Server Port Option (--repo-azure-port)	Azure Repository Server Port		
Azure Repository Server Certificate Verify Option (--repo-azure-verify-tls)	Validate Azure Repository Server Certificate.		
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		
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		

Parameter	Overview of parameters	Reason
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
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 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
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

Parameter	Overview of parameters	Reason
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
Repository Type Option (--repo-type)	Type of storage to use for the repository	posix/s3

2.3.5.4 About Sections in the Config File

In FEPBackup 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 metrics 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 metrics 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 CPU utilization of the DB container exceeds the threshold.

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



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

2.3.9.1 Change to FEPCluster CR - autoscale

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

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

Chapter 3 Operator Installation

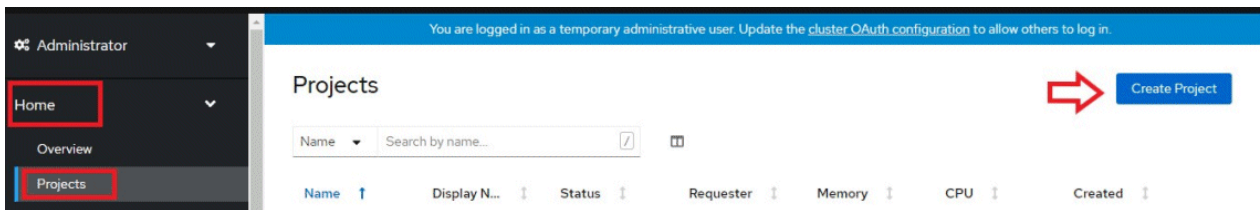
This chapter describes how to install FEP operator in a new namespace on openshift.

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.

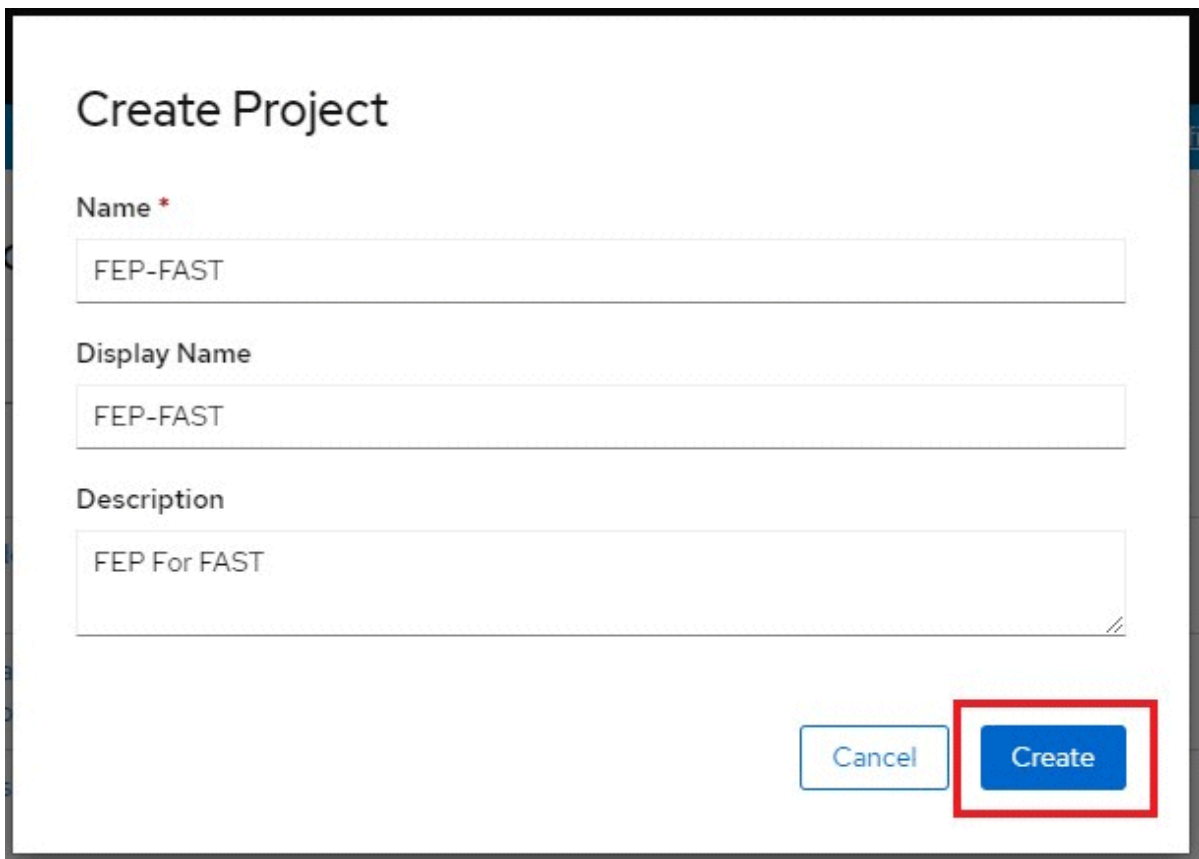
3.1 Pre-requisite

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.



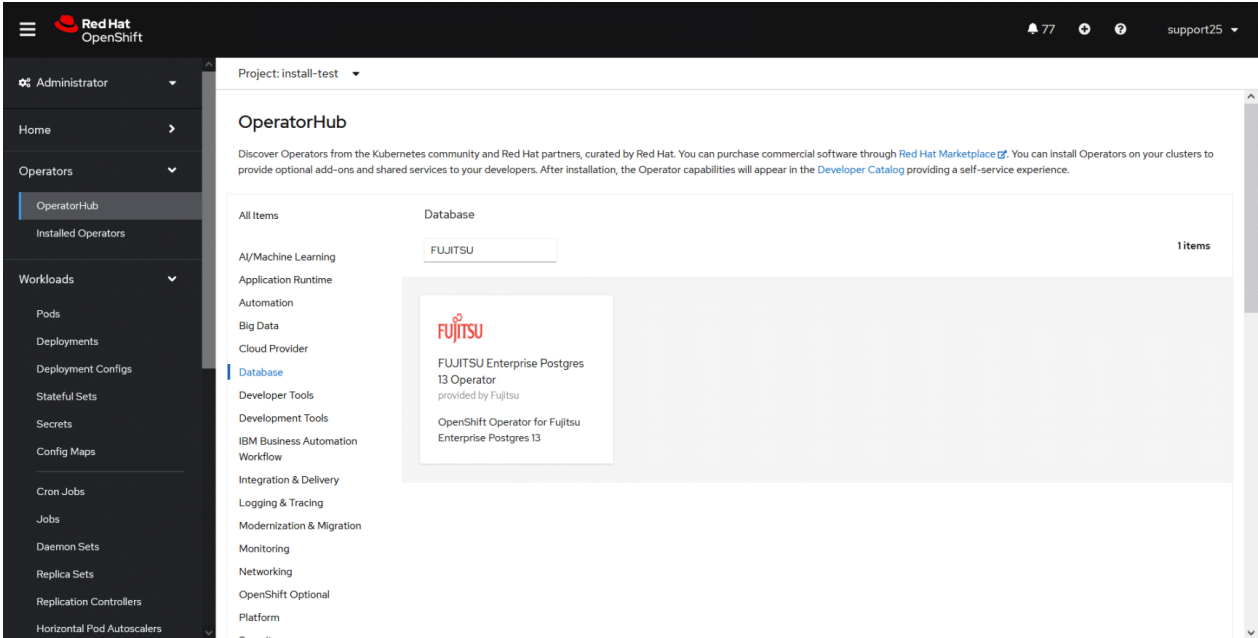
Note

FEP13 Operator installation needs Prometheus to be pre-installed in the Openshift cluster.

3.2 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 to find FUJITSU Enterprise Postgres 13 Operator.



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



FUJITSU Enterprise Postgres 13 Operator

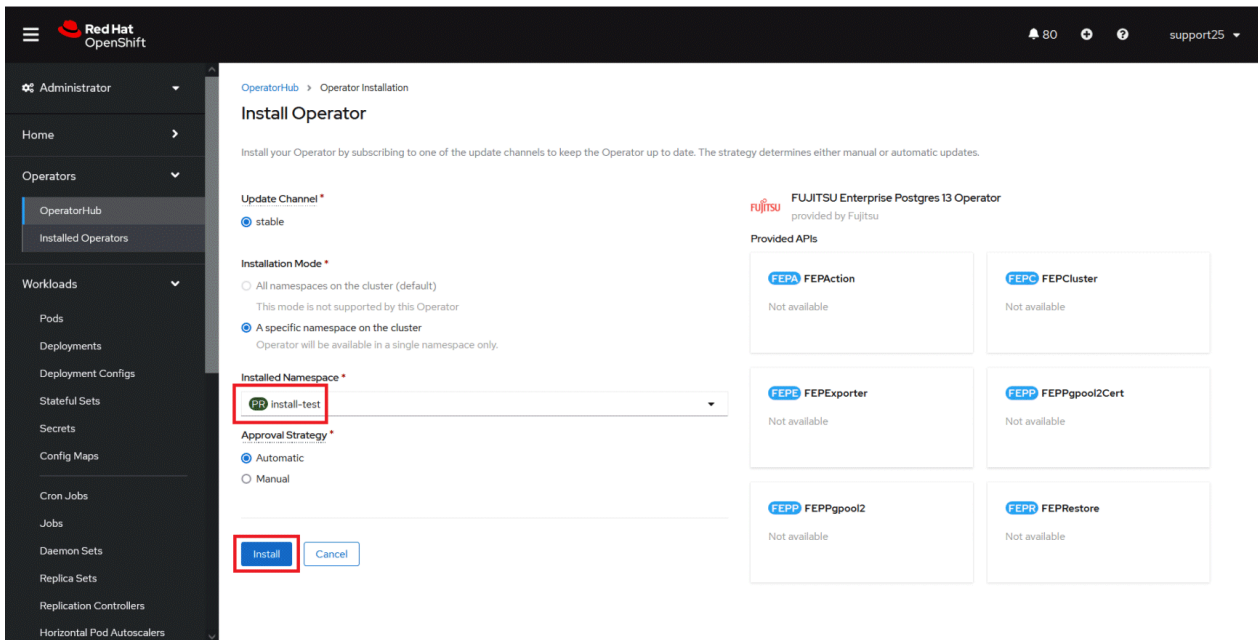


3.0.0 provided by Fujitsu

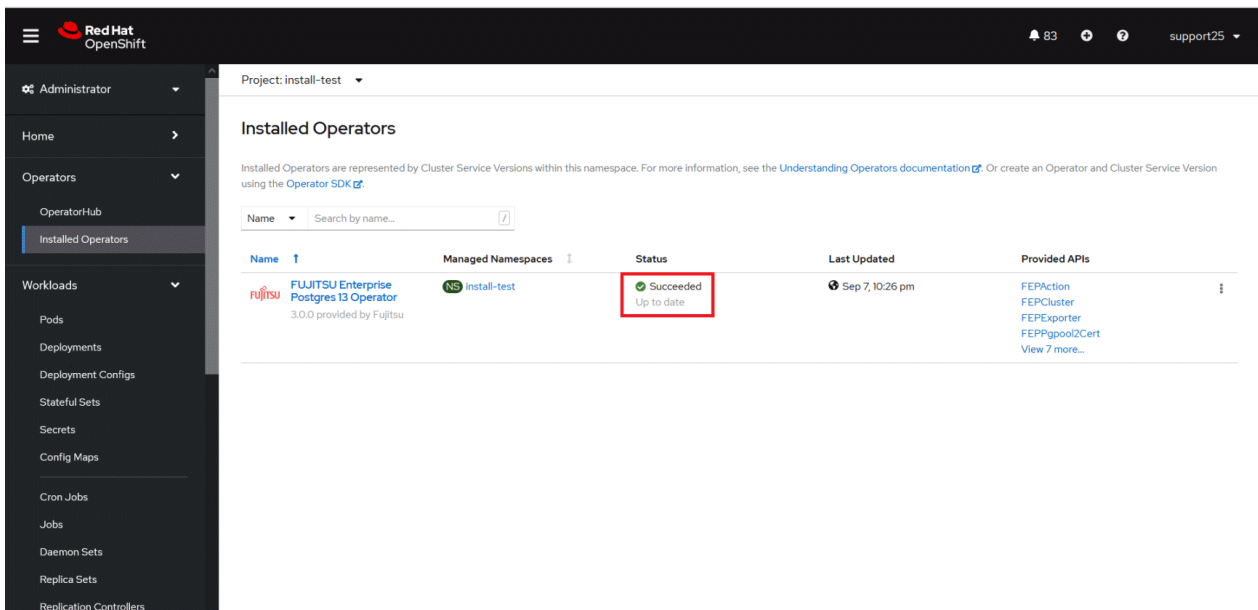


Latest Version 3.0.0	FUJITSU Enterprise Postgres 13 delivers an enterprise-grade PostgreSQL on OpenShift Container Platform.
Capability Level <input checked="" type="checkbox"/> Basic Install <input checked="" type="checkbox"/> Seamless Upgrades <input checked="" type="checkbox"/> Full Lifecycle <input checked="" type="checkbox"/> Deep Insights <input type="checkbox"/> 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. Use of the product is subject to Fujitsu evaluation license located at: https://www.fast.fujitsu.com/fujitsu-enterprise-postgres-trial-version-software-evaluation-license-agreement and the license period is 90 days after the download
Provider Type fujitsu-enterprise-postgres-13-registry-sp25	
Provider Fujitsu	
Repository N/A	
Container Image quay.io/fj-dbaas-dev/fujitsu-enterprise-postgres-operator-st-fj:v2.2.0-registry-test	

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



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



3.3 Implement Collaborative Monitoring Tools

There is a pre-requisite for running FEPEXporter.

- GAP(Grafana, AlertManager, Prometheus) stack is installed on host OpenShift cluster
- FEPCluster that needs to be scrapped 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 with OpenShift. Please refer to the following for deployment information.

(Refer: <https://docs.openshift.com/container-platform/4.6/monitoring/understanding-the-monitoring-stack.html>)

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

(Refer: <https://fast.fujitsu.com/knowledge-base/knowledge-articles-installationsetup/setting-up-grafana-on-ibm-linuxone>)

(Refer: <https://fast.fujitsu.com/knowledge-base/knowledge-articles-installationsetup/setting-up-grafana-on-amd64-ocp>)

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

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:

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.

Name	Managed Namespaces	Status	Last Updated	Provided APIs
FUJITSU Enterprise Postgres 13 Operator 3.0.0 provided by Fujitsu	NS install-test	Succeeded Up to date	Sep 7, 10:26 pm	FEPAAction FEPCluster FEPEXporter FEPPgpool2Cert View 7 more...

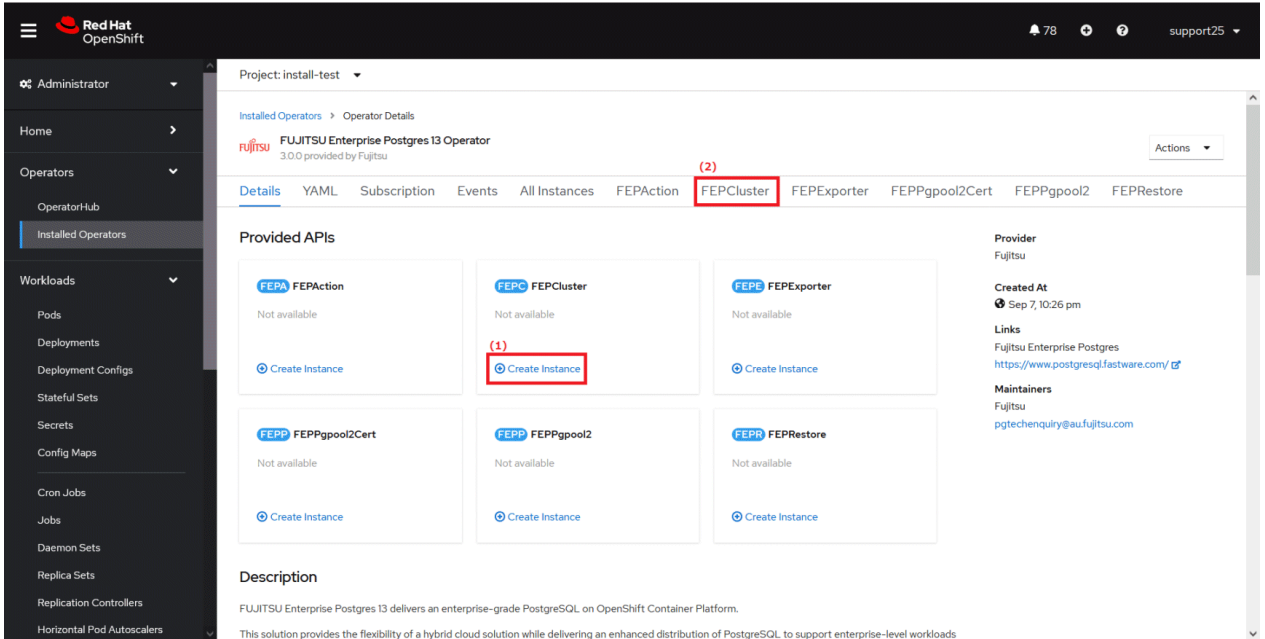
2. It will display a page with all CRs this operator supports. FEPCluster is the main CR and all others are child CR. We would create the main CR and all other CRs will be created automatically by Operator.

To create Cluster CR, either

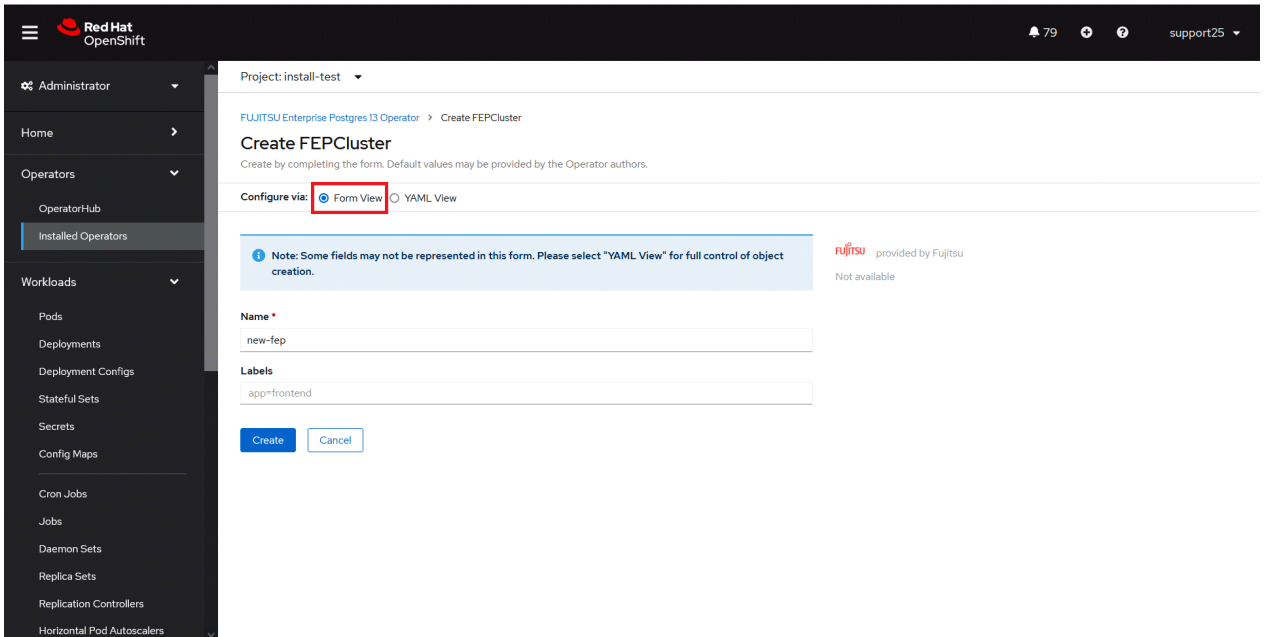
- (1) Click on "**Create Instance**" under FEPCluster.

OR

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



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.



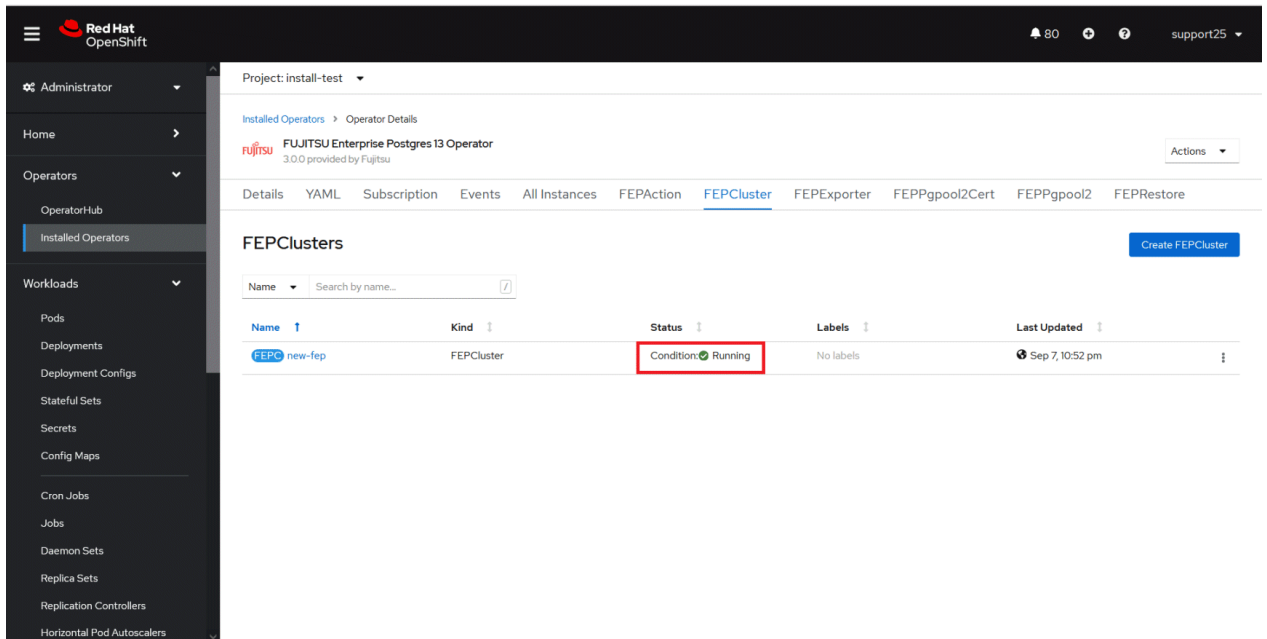
- In YAML View, starting value of CR is visible and one can choose to modify parameters before creating CR. Refer to the Reference for details of parameters.

The screenshot shows the Red Hat OpenShift console interface. On the left is a navigation sidebar with categories like Administrator, Home, Operators, Workloads, and Pods. The main content area is titled 'Project: install-test' and 'FUJITSU Enterprise Postgres 13 Operator > Create FEPCluster'. Below the title, it says '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. Below this is a code editor showing a YAML configuration for an FEPCluster. The configuration includes metadata (name: new-fep, namespace: install-test) and a spec section for the 'fep' resource, detailing custom annotations, forceSsl, image, pullPolicy, instances, mcSpec (with limits and requests for CPU and memory), podAntiAffinity, podDisruptionBudget, servicePort, and svcMode. At the bottom of the editor are 'Create', 'Cancel', and 'Download' buttons.

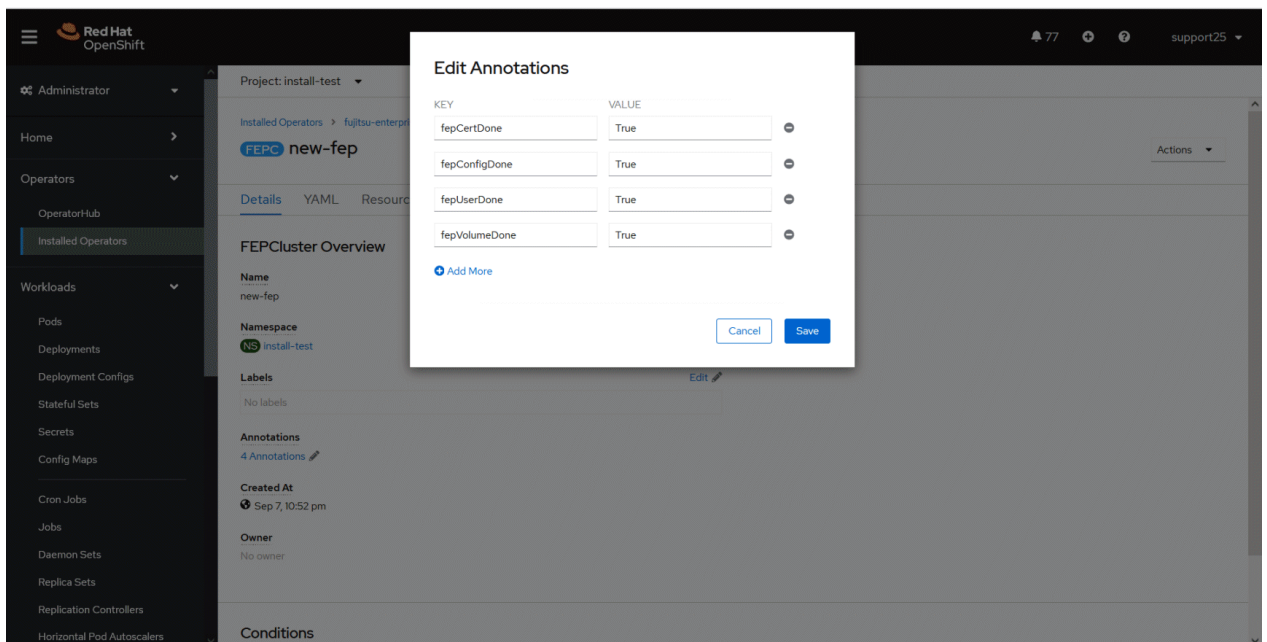
```
1 apiVersion: fep.fujitsu.io/v2
2 kind: FEPCluster
3 metadata:
4   name: new-fep
5   namespace: install-test
6 spec:
7   fep:
8     customAnnotations:
9       allDeployments: {}
10    forceSsl: true
11    image:
12      pullPolicy: IfNotPresent
13    instances: 1
14    mcSpec:
15      limits:
16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

- 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 FEP Cert 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

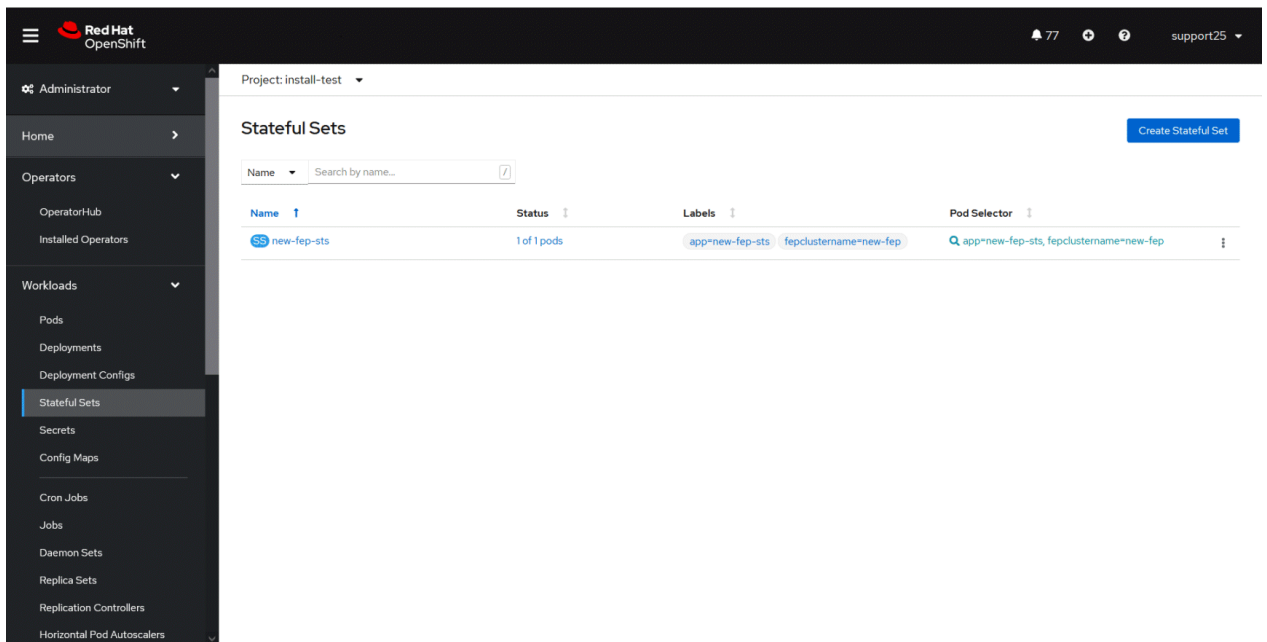
changes are reflected in child CRs. Child CRs are marked internal objects and hence will not be visible on the OCP console. However, you can check child CRs using command-line tools.



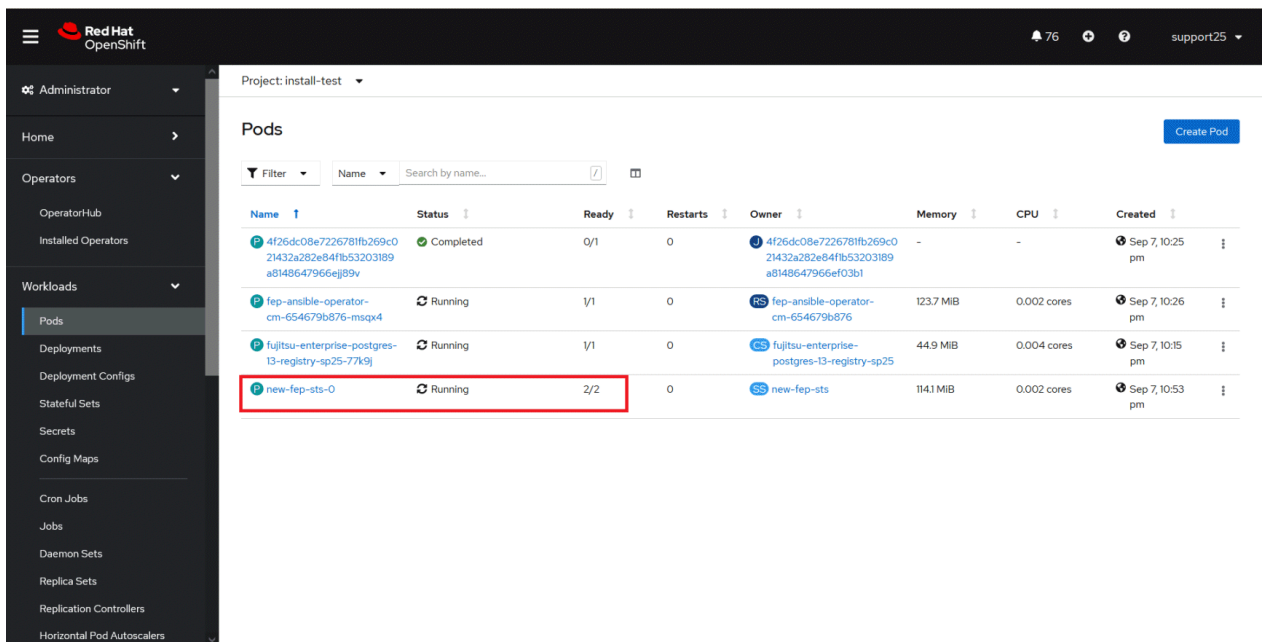
6. In FEPCluster CR, annotations are added to indicate that child CRs are created successfully and has initialised properly. It may take some time to complete.



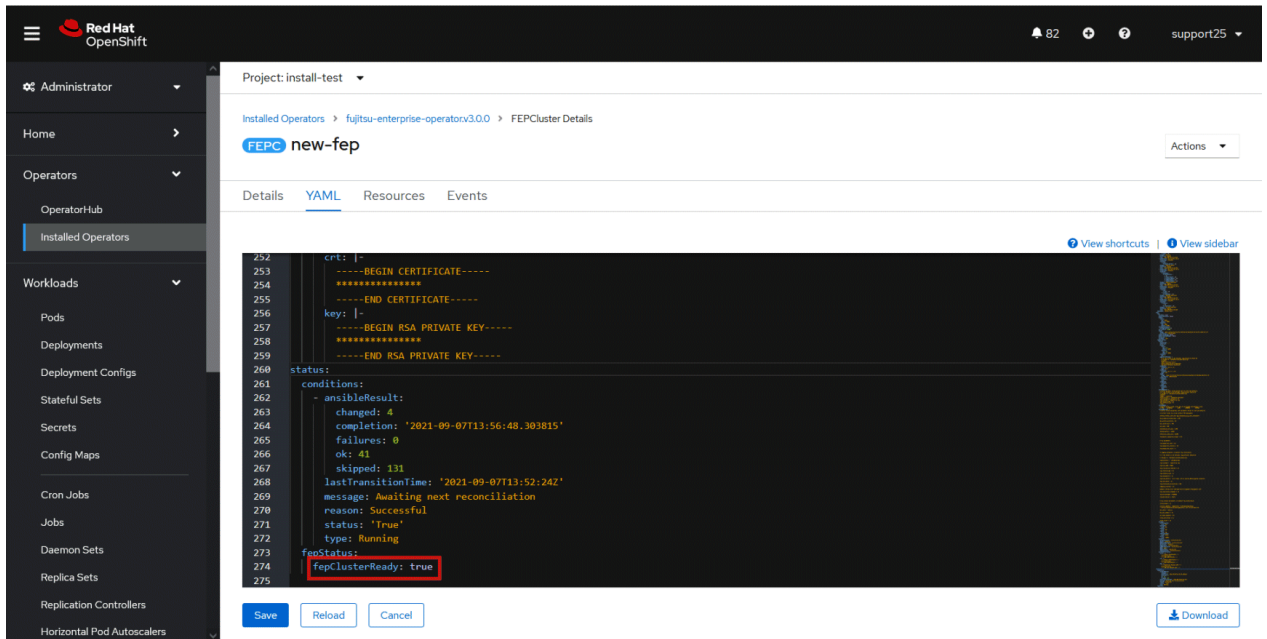
7. Once all four child CRs are marked done in annotations, operator creates StatefulSet for the cluster.



8. 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:



- 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:

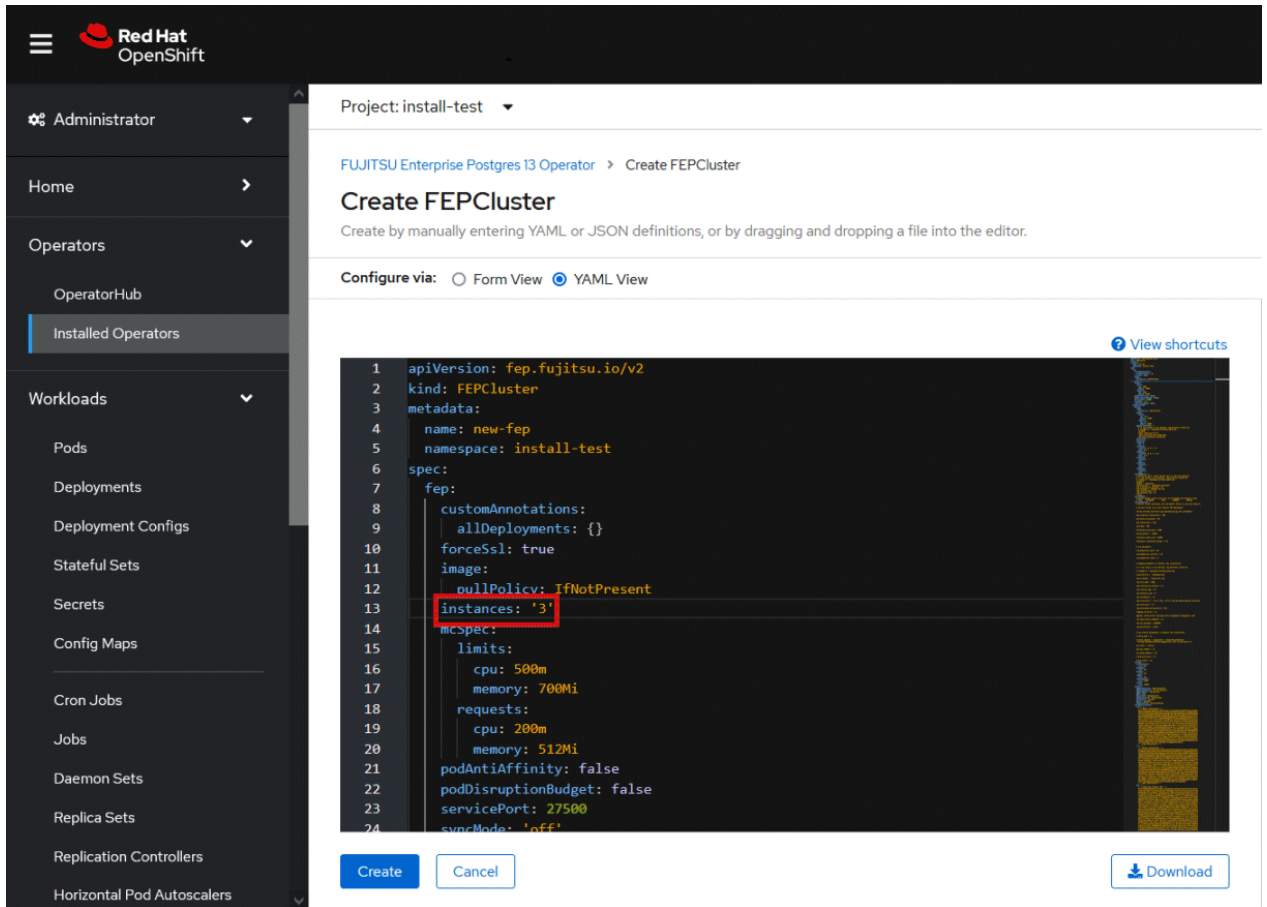
[Prerequisites]

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.

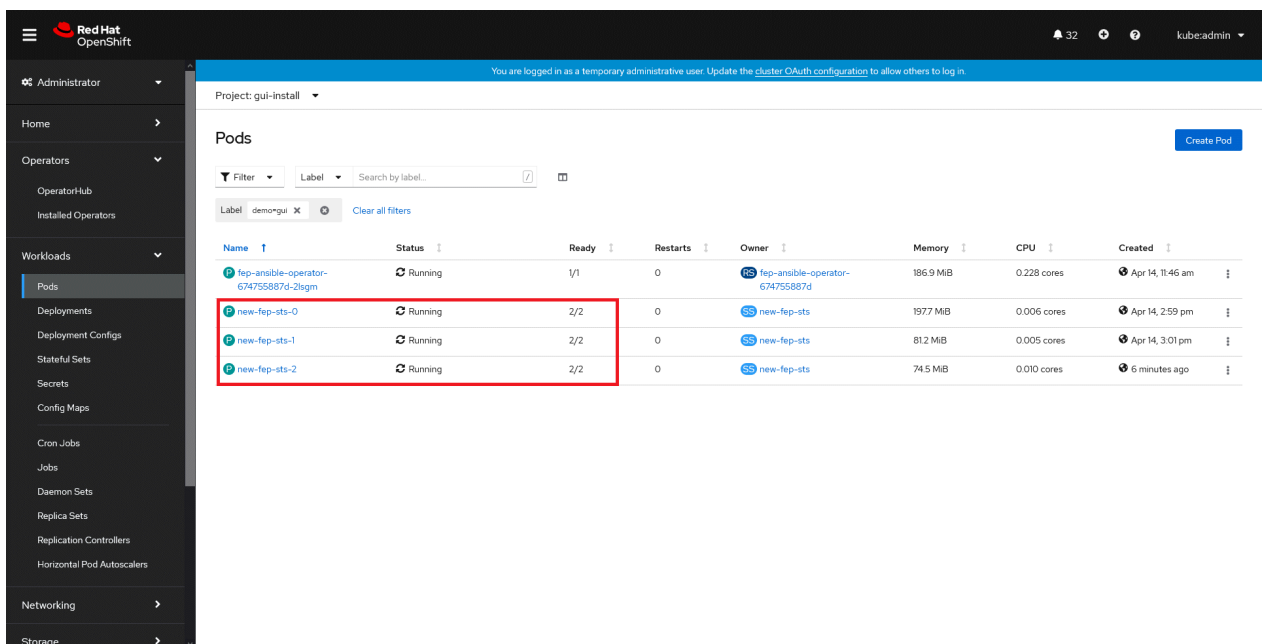
If you do not have shared storage, you can remove the backup section and the backup and archive volume sections to disable the backup feature and deploy the FEP cluster.

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

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



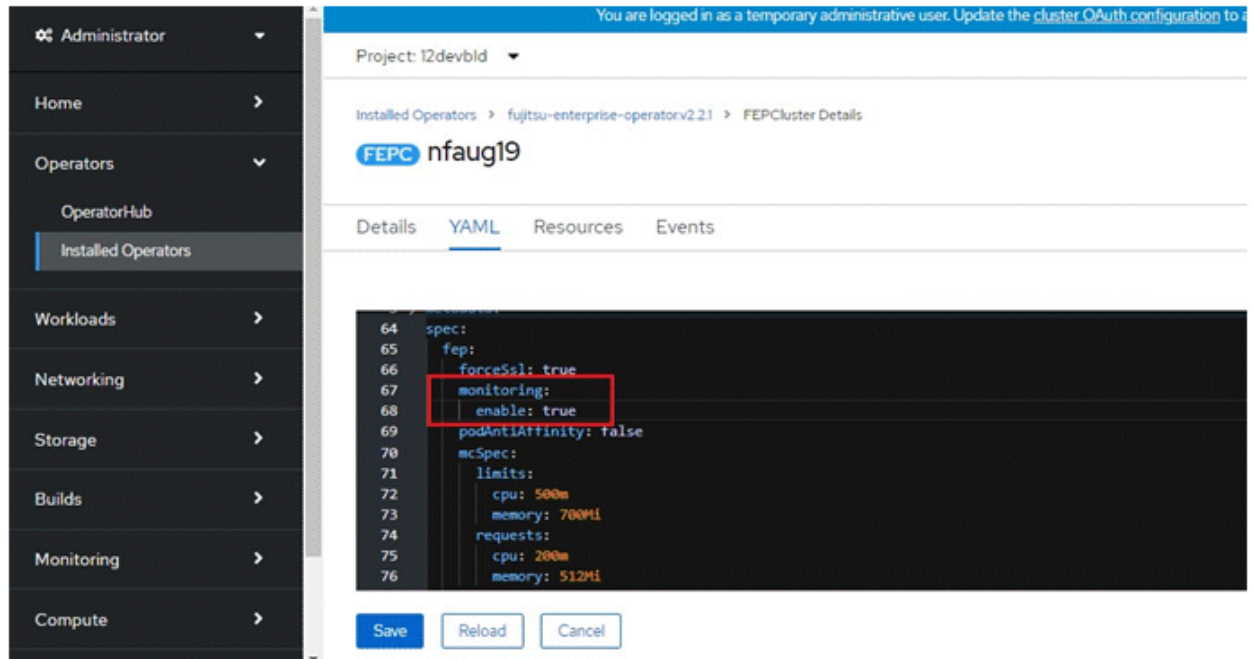
3. It is the same as the procedure from step 5 to step 10 in "4.1 Deploying FEPCluster using Operator".
4. Three pods deployed and ready for a highly available FEPCluster.



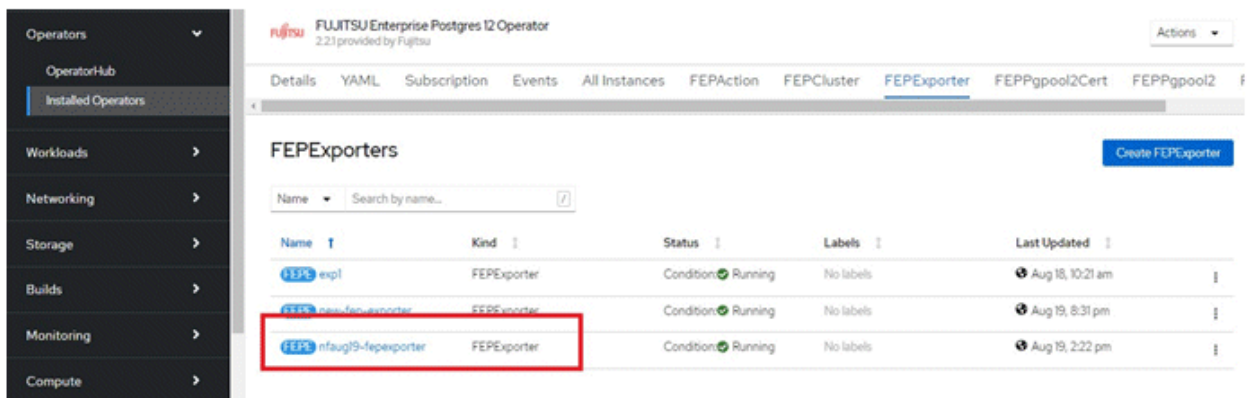
4.3 Deploying FEPEXporter

To deploy a FEPEXporter, follow these steps:

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.



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.



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 FEPEXporter 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)

- Now user can see scrape FEPEXporter specific metrics on Openshift Platform in monitoring section area using PROMQL to specify a metrics of interest

You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.								
pg_exporter_scrapes_total	prometheus-fep-exporter	new-fep-exporter-http	10.131.0.45:9187	new-fep-exporter-service	fepexporter-dev	new-fep-exporter-deployment-67f6764db8-pfngl	openshift-user-workload-monitoring/user-workload	new-fep-exporter-service
pg_exporter_scrapes_total	prometheus-fep-exporter	new-fep-exporter-http	10.131.2.133:9187	new-fep-exporter-service	l2devblid	new-fep-exporter-deployment-7569c8f8b7-96vht	openshift-user-workload-monitoring/user-workload	new-fep-exporter-service
pg_exporter_scrapes_total	prometheus-fep-exporter	nf18a12-fepexporter-http	10.129.3.28:9187	nf18a12-fepexporter-service	a-reg-install	nf18a12-fepexporter-deployment-65db656d7-p756c	openshift-user-workload-monitoring/user-workload	nf18a12-fepexporter-service
pg_exporter_scrapes_total	prometheus-fep-exporter	nf19aug-fepexporter-http	10.129.2.309:9187	nf19aug-fepexporter-service	aashish-test	nf19aug-fepexporter-deployment-68d65685f7-2zst4	openshift-user-workload-monitoring/user-workload	nf19aug-fepexporter-service
pg_exporter_scrapes_total	prometheus-fep-exporter	nf19g9-fepexporter-http	10.129.2.352:9187	nf19g9-fepexporter-service	l2devblid	nf19g9-fepexporter-deployment-f8db9f7cf-89wm7	openshift-user-workload-monitoring/user-workload	nf19g9-fepexporter-service

Note

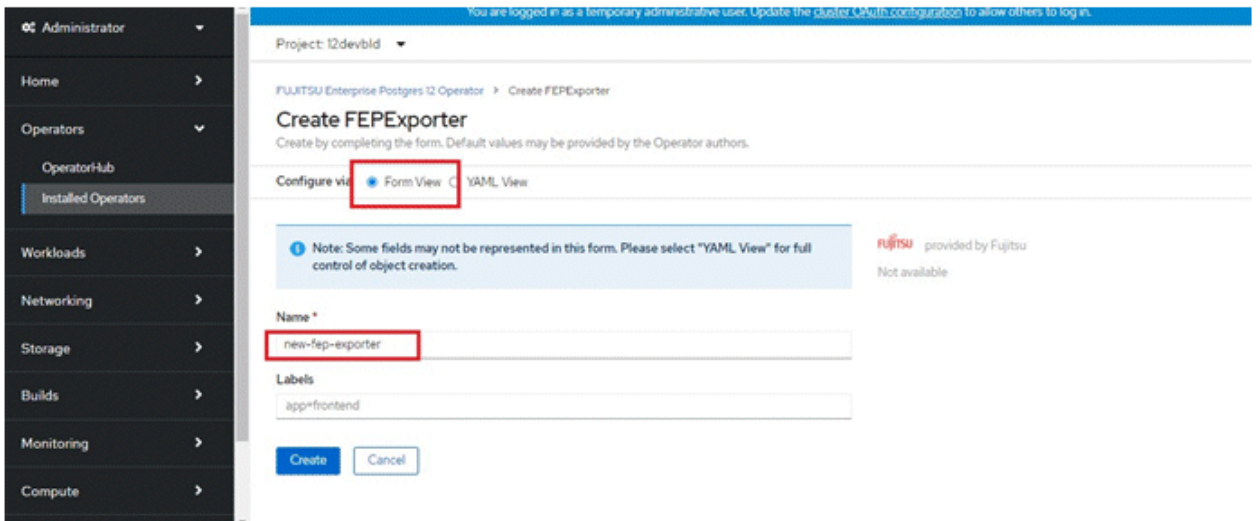
- User can set `fep.monitoring.enable` to true or false on an already instantiated cluster as well to achieve desired results
- `pgMetricsUser` can be defined later on a running FEPCluster with monitoring enabled and can force FEPEXporter to use `pgMetricsUser` by mere restarting it (refer `restartRequired`). 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 `postgres.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

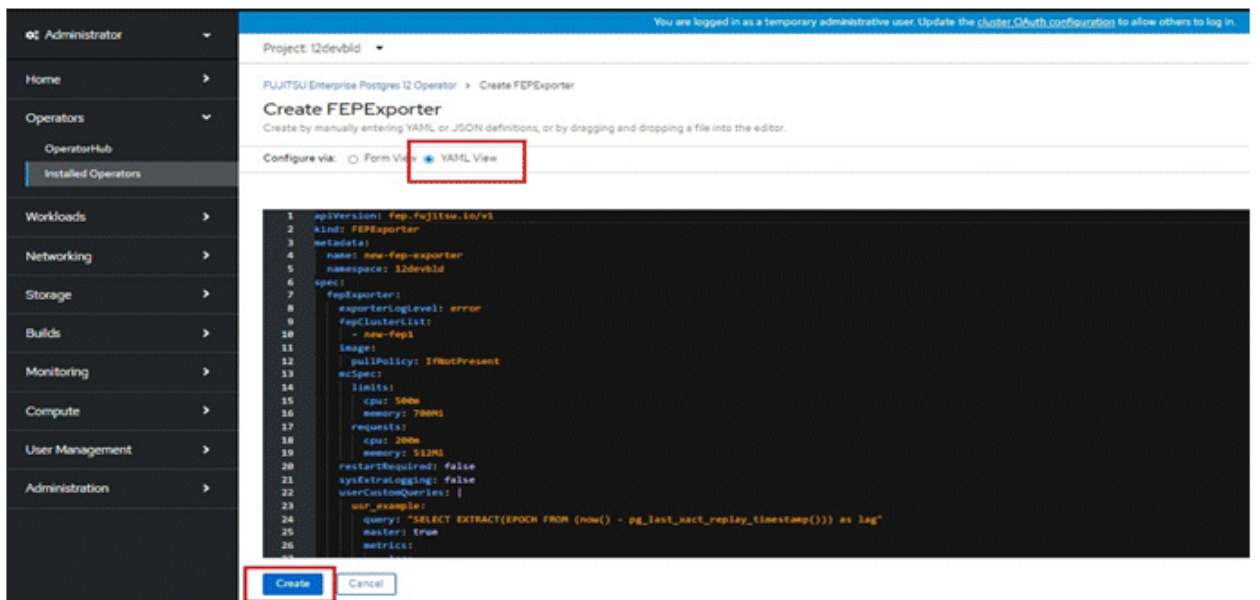
FEPEXporter is an independent CR; hence it does not necessarily depend on main FEPCluster CR. To deploy a FEPEXporter in given namespace follow the below step.

- To create FEPEXporter CR, either
 - Click on "**Create Instance**" under FEPEXporter.
 - OR
 - Click on "**FEPEXporter**" on top and then click on "**Create FEPEXporter**" on the next page.
- 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.

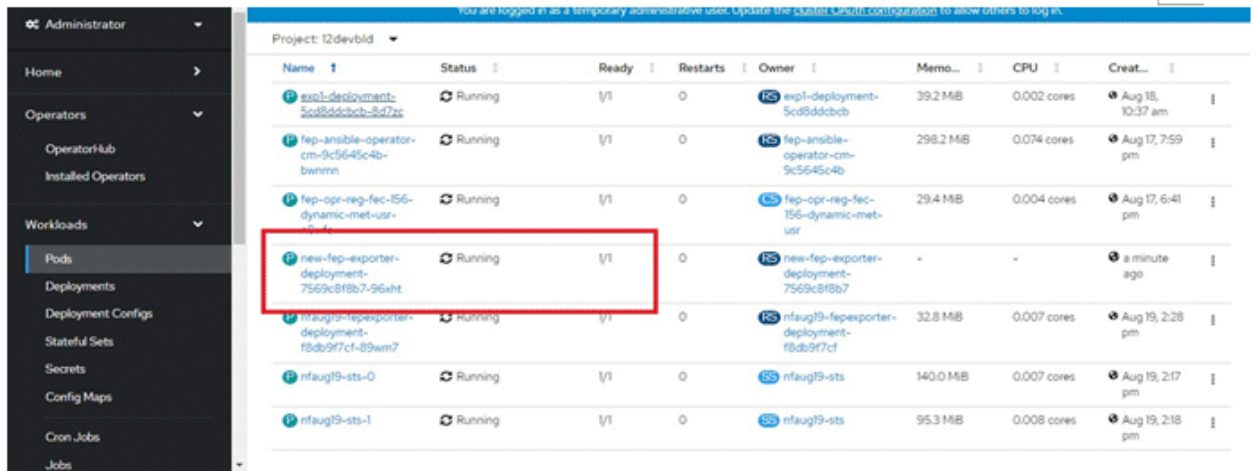
- FEPEXporter scrapes metrics for FEPCluster within same namespace.



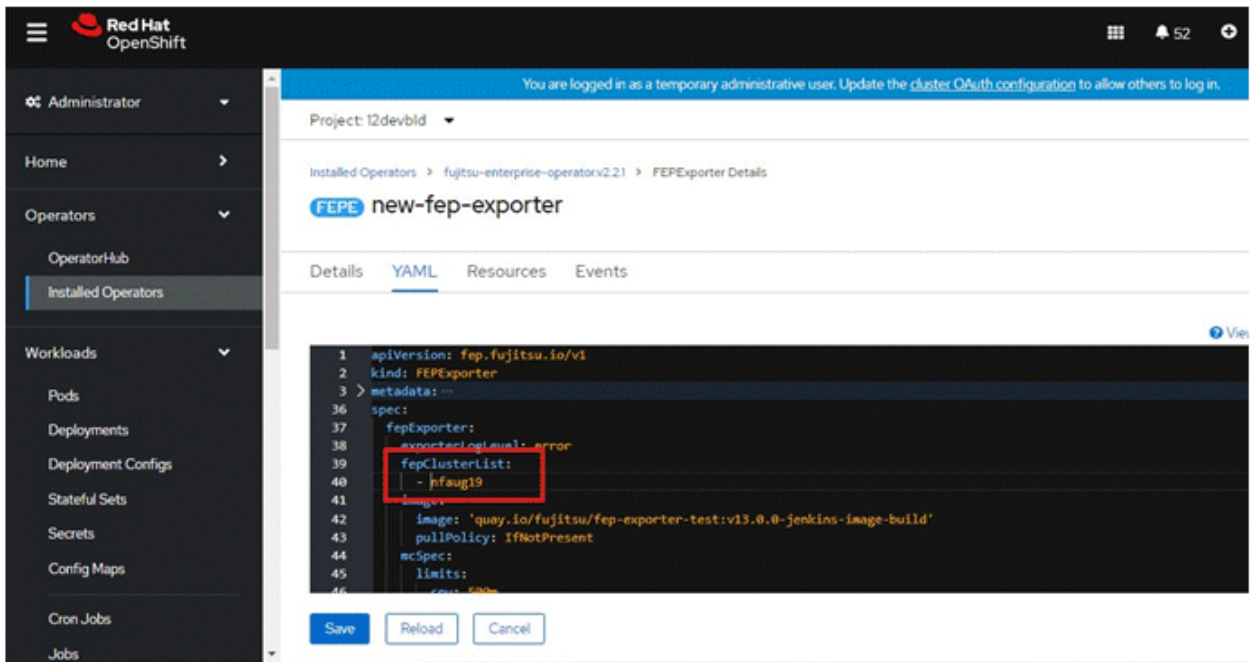
- In YAML View, starting value of FEPEXporter CR is visible and one can choose to modify parameters before creating CR. Refer to the Reference for details of parameters.



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



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



- It will recreate FEPEXporter pod with a new dataresource secret. It will list down all the database with statistics of specified FEPEXporter 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 FEPEXporter
- Patroni RESTAPI within FEPEXporter
- Postgres traffic within FEPEXporter (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.



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"



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.

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=abcdefghijklk -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)) # all in one line
```

Note

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
```

Note

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
```

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.

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 rewinduser user client certificate

```
cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: mydb-rewinduser-cert
  namespace: my-namespace
spec:
  subject:
    commonName: "rewinduser"
    duration: 8760h
  usages:
    - client auth
  secretName: mydb-rewinduser-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-13-server:ubi8-13-0.0'
    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-----
    MIIIEowIBAACAQEA0DFkImha8CIJiVcwXbBP1L+/DmS9/ipRhQQHxf05x7jSOnse
    IHdFd6+Qx2GX8KAIhVykf6kfacwBYTATU1xDgwWTm82KVRPh+kZDIj2wPcJr14m
    mTP6I6a2mavUgDhezHc9F8/dchYj3cw81X0kU6xamqrKQY1xQH48NkI0qcwh06sK
    AHF4eWfCr8Ot44xADIALJcU2CS1RKSZEtURZ+30Py+j907Enjp1YR33ZKUHW30pU
    9dpIneyfXBN/pT6cX3MetYwtgmpV/pHqY8pbxqGfoYRhGQDsSRC14dtlecaZeZ4j
    uTOotcPkZELHP6eu8gaLtycG9lpbAMQ15w0r8QIDAQABaoIBACq213qPuoimExrQ
    fQXaNjmqNYK4fJqxCB6oUwf0Flu4ubkx5V532hLSPHWLs+a01AWlbNozSoBVou8G
    64VwrA9bv3/cJVqZZ6/UzUTbHPU+Ogh24qhwF5QU8kXZEUI1To3YsPoftalGjX9G
    Ff0fLcLVC8nL3K9RiaDXxXbEYpWrYu39M3FCpAXAzV2PrNxsP9PKyNWHnBpc08z5
    tFj45/bHn+j31AVVvgWtqz0pLks57hc4Q7yW/2RoRYq2md1KI7090LNwtkWEOVqb
    qnraorh2TwnNaOB5oX5/1JvKtlq778fw96jGqykBr0+DKozj9rlr1OGgYOKDw1D
    nsZJPAECgYEA+Oqf/fxtPdsNGiaL2Z/heewvtaxjw/WoEVBFECb6/y4Ro7aux9nB
    16FcVi79CwfpOUTJ7cnZvYsmBk5GWEObEIAeo6llvm/QeltM5+usAPd5/TcHXLye
    92OnXmq7h3F4UXEkMayak8Lpu/TdmR5uOaL+m4aEu+XMY5tlxqDCnyECgYEA1h4X
    jCPi7Ja5CHK7a2Ud4TL2DNpIBE6GSK9iQ+0xFL6TsiK2Sfu6n8mx2sh+Jm0KHTiE
    /gWHdHQZSSWiuUlfHoYEq3Rq8S6Av3GsGtRSpo03j7BE8C20Vpt0FnNTjZmdzf2/
    YZxc5KuYlh9qeY7Y7ceOsWA8JckDgMHPYzyLatECgYBALD0TPgDr8Y1vMIDDmlqH
    FF04eTk/TBYIYKltgJ81KqthibeFzp4q+W7UyUhzj5a4XQOyS1fYhFpJReTc3JEd
    r+o2SH3ymuEkqmUpZZjyptrMbnWN4g3t4TDjaHqo6QQbD+GdcZyNy9M1Np9N5p17E
    fUEml4dg6d3H0Ehs7QVAAQKBgQDRUx3mLXc9oKRINBIyDerGLJILQqLBQxtYl81T
    ZuFizGWL8w+PCIAMkpxDrVpWqqcGpiiuRi2ElbPapOaOg2epaY/LJscd/j5z6uc8
    W3JoN1jpKoRa4f0578Pv5tM6TYHOzlf5Veoiy/a8sI3hrNuiqkM/+TsUHY5FJDRh
    aeDk4QKBgCOHIEvvr+MwuwakzD6lNCbb8H6fvZWRAT8BYyz3wW9YfnV4J4uh/Bl
    moWYgIK2UpkrrhA8scMUC790FoybQeParQ35x7J191bmTKkCqsX63fyqqYhx3SXRl
    JSktmH4E2cGmosZisjB7COKHR32w0J5JCgaGInQxjldbGrwhZQpn
    -----END RSA PRIVATE KEY-----
  crt: |-
    -----BEGIN CERTIFICATE-----
    MIID2CCAsCgAwIBAgIQDfFYteD4kzj4Sko2iy1IJTANBgkqhkiG9w0BAQsFADBX
    MRgwFgYDVQKEw9NeSBPcmdhbm16YXRpb24xCzAJBgNVBAsTAkNBMS4wLAYDVQQD
    EyVNeSBPcmdhbm16YXRpb24gQ2Vydg1maWNhdGUgQXV0aG9yaXR5MB4XDTEyMDQy
    MDAwMDQ1OV0XDTIxMDQyMDAxMDQ1OVowGDEWMBQGAlUEAwNKi5jaGctCHRjLnBv
    ZDCCASIdQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBANaxZCJoWvAiCYlXMF2w
    T5S/vw5kvf4qUYUEB8Xzuce40jp7HiB3RXevkMdh1/CgIgIVcpH+pH2nMAWEwE1N
    cQ4MFk5vNilUT4fpGQyI9sD3Ca9eJpkz+iOmtpmr1IA4Xsx3PRfP3XIWI93MPNV9
    JFOSWpqqyKGCUB+PDZCNKnMITurCgBxeHlnwq/DreOMQAYANSXFNgktUSkmRLVE
    Wft9D8vo/dOxJ46dWEd92S1B8N9KVPXaSJ3snlwTf6U+nF9zHrWMLYJqVf6R6mPK
    W8ahn6MkYEEA7EkQpeHbZxNgmXmeI7kzqLXD5GRCxz+nrvIGi7cnBvZaWwDEJecN
    K/ECaWEAAAoB3jCB2zATBgnVHSUEDDAKBggrBgEFBQcDATAMBgnVHRMBAf8EAjAA
    MIG1BgNVHREEga0wgaqCCWxvY2FsaG9zdIIBKi5jaGctCHRjLnBvZC5jbHVzdGVy
    LmxvY2FsgmMl5ZGItaGVhZGxl3Mtc3ZjghsqLm15ZGItaGVhZGxl3Mtc3Zj
    LmNoZy1wdG9yYXV0aG9yaXR5MjhhLXB0Yy5zdmMuY2xlc3Rlcj5sb2NhbDANBgkqhkiG
    9w0BAQsFAAOCAQEALnhliDflu+BHp5conq4dXBwD/Ti2YR5TWQixM/0a6OD4KecZ
    MmaLl0T+OJJvA/j2IufZpc7dzEx5mZDKR2CRmoq10qZXqCRTrBZSxm6ARQWoYpeg
    9c014f8roxrkMGUKVPTKUwAvbnNYhD2l6PlBPwMpkMUFqFaSEXMaPyQKhrTQxdpH
    WjuS54OP0lmpEyu/yiaD98LtrTXnb6jch84SKf6Vii4HAVQyMeJaW+dpkqcI2+V
    Q4fkWYSjy8BNcmXCwvHDLdy+s4EXWvHafhusuUhc4HyMblA6hd5hJhgFSnEvLy
    kLA0L9LaScxee6V756Vt9TNlNGjmwYQDohnQQ==
    -----END CERTIFICATE-----
  cacert: |-
    -----BEGIN CERTIFICATE-----
```

```

MIIDXCcAkSgAwIBAgIRAMPzF3BNFxt9HWE+NXlFQjQwDQYJKoZIhvcNAQELBQAw
VzEYMBYGA1UEChMPTXkgT3JnYW5pemF0aW9uMQswCQYDVQQLLEwJJDQTEuMCwGA1UE
AxMlTXkgT3JnYW5pemF0aW9uIENlcnRpZmljYXR1IEF1dGhvcml0eTAeFw0yMTA0
MTkwNDQ0MjNaFw0zMTA0MTcwNDQ0MjNaMFcxGDAwBGNVBAoTD015IE9yZ2FuaXph
dGlvbjEELMAkGA1UECmQ0ExLjAsBgNVBAMTJU15IE9yZ2FuaXphdGlvbiBDZXJ0
aWZpY2F0ZSBBDXRob3JpdHkwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIB
AQc5t6CS23G1k65YMw5e4i4xHldyxkCZS67w/6LWqeIlyKmfAaEl83Wwy8MHUpOb
4mahtUafEzDEOX6+URf72J8m0voldQ5FYr1AyUOyX8U90wGFqhbEgKRqt7vZEwIe
2961fwgHh6917zI4xmt5W6ZJ5dBQVtkhzB+Pf706KBYjHoCnBBkfnVzsfzQ/1hnR
0UzimfAc7Ze+UNwhXJhinFRJ3YuR+xiOTpPk1lGXPhLgFSQheKz4KepcbQEQKejb
jg0dumloBYIXZTSSbi09rNmFUVLB5DcV0vZbSrGxLjWLBt5U8N2xf2d1bvkQW+bw
Kklf90G26bAi27tuJurzn3r3AgMBAAGjIzAhMA4GA1UdDwEB/wQEAwICpDAPBgNV
HRMBAf8EBTADAQH/MA0GCSqGSIb3DQEBCwUAA4IBAQAAM0CN3n5C/KOT4uZ4ewwKK
rHmANBPVM9u6MJB08U62HcqLeoCuDFeU8zmUjLHjsQaPX64mJZ1R7T5y52gEKO5A
0qsBz3pg/vJ5DJTtV0698+1Q1hb9k3smQdksAim19FZqysB7J4zK/+8aJ/q2kIFvs
Jk3ekwQdQ3xfggklBQVuf76gr1v0uY1PtPffP1fcGZ06Im6mqbajenXoR1PxPB0
+zyCS8DkgPtDulplruwvXCFMYw9TPbzXKlt7t1sqrXogYLnXWJDzM1nOYcNd+rDm
qxenV9Ir8RqZ0XSYuYyzRka5N4dhIhrzTAiNdeU5gzynXOz67u/Iefz1iK9ZcdE3
-----END CERTIFICATE-----

```

Deploy FEPCluster with automatic certificate management

Use the following yam1 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-13-server:ubi8-13-0.0'
    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

# 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-----
    MIIIEowIBAAKCAQEAODFkImha8CIJiVcwXbBP1L+/DmS9/ipRhQQHxfO5x7jSOnse
    IHdFd6+Qx2GX8KAiAhVykf6kfacwBYTATU1xDgwWTm82KVRPh+kZDIj2wPcJr14m
    mTP6I6a2mavUgDhezHc9F8/dchYj3cw81X0kU6xamqrKQYlXQH48NkI0qcwh06sK
    AHF4eWfCr8Ot44xADIA1JcU2CS1RKSZEtURZ+30Py+j907EnjplYR33ZKUHW30pU
    9dpIneyfXBN/pT6cX3MetYwtgmpV/pHqY8pbxqGfoYRhgQDsSRC14dtlecaZeZ4j
    uTOotcPkZELHP6eu8gaLtycG9lpbAMQ15w0r8QIDAQABAoIBACq213qPuoimExrQ
    fqXaNJmqNYK4fJqXC6oUwf0Flu4ubkx5V532hLSPHWLs+a0lAW1bNozSoBV0u8G
    64VwrA9bv3/cJVqZZ6/UzUTbHPU+Ogh24qhwF5QU8kXZEU11To3YsPoftalgjX9G
    Ff0fLcLVC8nL3K9RiaDXXbEYpWrYu39M3FCpAXAZV2PrNxsP9PKyNWHnBpc08z5
    tFj45/bHn+j31AVVvgWtqz0pLks57hc4Q7yW/2RoRYq2md1KI7090LNwtkWEOvqb
    qnraorh2TgGnNaOB5oX5/lJvKtlq778fw96jGqykBr0+DKozj9rlr1OGgYOKDwLD
    nsZJPAECgYEA+Oqf/fxtPdsNgialL2Z/heewvtaxjw/WoEVBFEcb6/y4Ro7aux9nB
    16FcVi79CwfP0UTJ7cnZvYsMbk5GWEObEIAeo61lvm/QeltM5+usAPd5/TcHXLye
    92OnXmq7h3F4UXEkMayak8Lpu/TdmR5uOaL+m4aEu+XMY5tlxqDCnyECgYEA1h4X
    jCPi7Ja5CHK7a2Ud4TL2DNpIBE6GSK9iQ+0xFL6TsiK2Sfu6n8mx2sh+Jm0KHTiE
    /gWHdHQZSSWiuULfHoYEeq3Rq8S6Av3GsGtRSpo03j7BE8C20Vpt0FnNTjZmdzf2/
    YZxc5KuYLh9qeY7Y7ceOsWA8JckDgMHPYzyLAtECgYBALD0TPgDr8Y1vMIDdmlqH
    FF04eTk/TBYIYKltgJ81KqthibeFzp4q+W7UyUhzj5a4XQOyS1fYhFpJReTc3JEd
    r+o2SH3ymEkqmUpZZjyptrMbwN4g3t4TDjaHqo6QQbd+GdcZyNy9M1Np9N5pl7E
    fUEm14dg6d3H0Ehs7QVAAQKBgQDRUx3mLXc9oKRINBiYDerGLJILQqLBQxtY181T
    ZuFizGWL8w+PCIAMkpxDrVpWqccGpiiurI2ElbPapOaOg2epaY/LJscd/j5z6uc8
    W3JoNljpKoRa4fO578Pv5tM6TYHOz1F5Veoiy/a8sI3hrNuiqkM/+TsUHY5FJDRh
    aeDk4QKBgCOHievR+MwuwakzD6lNCbb8H6fvZ3WRAT8BYyz3wW9YfnV4J4uh/Bl
    moWYgIK2UpkrhA8scMUC790FoybQeParQ35x7J191bmTKkCqsX63fyqqYhx3SXR1
    JSktmH4E2cGmosZisjB7COKHR32w0J5JCgaGInQxjldbGrwhZQpn
    -----END RSA PRIVATE KEY-----
  crt: |-
    -----BEGIN CERTIFICATE-----
    MIID2CCAsCgAwIBAgIQDfFYteD4kZj4Sko2iy1IJTANBgkqhkiG9w0BAQsFADBX
    MRgwFgYDVQQKEw9NeSBPcmdbmbl6YXRpb24gQ2VydG1maWNhdGUgQXV0aG9yaXR5MB4XDTIxMDQy
    EyVNeSBPcmdbmbl6YXRpb24gQ2VydG1maWNhdGUgQXV0aG9yaXR5MB4XDTIxMDQy
    MDAwMDQ1OVVoXDTIxMDQyMDAxMDQ1OVVowGDEWMBQGA1UEAwNKi5jaGctCHRjLnBv
    ZDCCASIdQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBANAxZCJoWwAiCYlXMF2w
    T5S/vw5kvf4qUYUEB8Xzuce40jp7HiB3RXevkMdh1/CgIgIVcpH+pH2nMAWEwE1N
    cQ4MFk5vNilUT4fpGQyI9sD3Ca9eJpkz+iOmtpmr1IA4Xsx3PrfP3XIWI93MPNV9
    JFOsWpqqyKJcUB+PDZCNkNMITurCgBxeHlnwq/DreOMQAYANSXFNgtUSkmRLVE
    Wft9D8vo/dOxJ46dWED92S1B8N9KVPXaSJ3snlwTf6U+nF9zHrWMLYJqVf6R6mPK

```

```

W8ahn6MkYYEA7EkQpeHbZXnGmXmeI7kzqLXD5GRCxz+nrviGi7cnBvZaWwDEJecN
K/ECaWEAAaOB3jCB2zATBgNVHSUEDDAKBggrBgEFBQcDATAMBGNVHRMBAf8EAjAA
MIG1BgNVHREEga0wgaqCCWxvY2FsaG9zdIIbKi5jaGctcHRjLnBvZC5jbHVzdGVy
LmxvY2FsgHMqLm15ZGItaGVhZGxlcm3Mtc3ZjghsqLm15ZGItaGVhZGxlcm3Mtc3Zj
LmNoZyldGOCyoubXlkYiIoZWFKkbGVzcy1zdmMuY2hnLXB0Yy5zdmOCLSoubXlk
YiIoZWFKkbGVzcy1zdmMuY2hnLXB0Yy5zdmMuY2xlc3Rlcj5sb2NhbDANBgkqhkiG
9w0BAQsFAAOCAQEALnhliDflu+BHp5cong4dXBwD/Ti2YR5TWQixM/0a6OD4KecZ
MmaLl0T+OJjvA/j2IufZpc7dzEx5mZDKR2CRmoq10qZxqCRTrBZSxm6ARQWoYpeg
9c014f8roxrkMGUKVPTKUwAvbnNYhd216P1BPwMpkMUFqFaSEXMaPyQKhrTQxdpH
WjuS54OP0lmp0peYu/yiaD98LtrTXnb6jch84SKf6Vii4HAVQyMeJaW+dpkqcI2+V
Q4fkWYSJy8BNcmXCwvHDLdy+s4EXWwHafhusuUhcp4HyMblA6hd5hJhgFSnEvLy
kLA0L9LaScxee6V756Vt9TN1NGjwmwyQDohnQQ==

```

-----END CERTIFICATE-----

ca.crt: |-

-----BEGIN CERTIFICATE-----

```

MIIDXCcAKSgAwIBAgIRAMPzF3BNFXT9HWE+NX1FQjQwDQYJKoZIhvcNAQELBQAw
VzEYMBYGA1UEChMPTXkgT3JnYW5pemF0aW9uMQswCQYDVOQLEwJQTEuMCwGA1UE
AxMlTXkgT3JnYW5pemF0aW9uIENlcnRpbzZmljYXRlIEF1dGhvcml0eTAeFw0yMTA0
MTkwNDQ0MjNaFw0zMTA0MTcwNDQ0MjNaMFcxGDAWBgNVBAoTD015IE9yZ2FuaXph
dGlvbWJELMakGALUECxCQC0ExLjAsBgNVBAMTJU15IE9yZ2FuaXphdGlvbiBDZXJ0
aWZpY2F0ZSBDbXRob3JpdHkwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIB
AQc5t6CS23G1k65YMw5e4i4xH1dyxkCZS67w/6LWqeI1YKmfAaE183WwY8MHUpOb
4mahtUafEzDEOX6+URf72J8m0voldQ5FYr1AyUOyX8U90wGFqhbEgKRqt7vZEwIe
2961fwqHh6917zI4xmt5W6ZJ5dBQVtkhzB+Pf706KBYjHoCnBBkfnVzsfZQ/1hnR
0UzimfAc7Ze+UNwhXJhinFRJ3YuR+xiOTpPk11GXPhLgFSQheKz4KepcbQEKejb
jg0dumloBYIXZTSSbi09rNmFUVLB5DcV0vZbSrGxLjWLBt5U8N2xf2d1bvkQW+bw
Kklf90G26bAi27tujurzn3r3AgMBAAGjIzAhMA4GA1UdDwEB/wQEAWICpDAPBgNV
HRMBAf8EBTADAQH/MA0GCSqGSIb3DQEBCwUAA4IBAQA0CN3n5C/KOT4uZ4ewwKK
rHmANBPVM9u6MJB08U62HcqLeoCuDFeU8zmUjLHjsQaPX64mJZ1R7T5y52gEKO5A
0qsBz3pg/vJ5DJTv0698+1Q1hb9k3smQdksAim19FZqysB7J4zK/+8aJ/q2kIFvs
Jk3ekwQdQ3xfggk1BQVuf76gr1v0uY1PtPfP1fcGZ06Im6mqbajenXoR1PxPB0
+zyCS8DkgPtDulplruwvXCFMyw9TPbzXK1t7t1sqRXogYLnXWJDzM1nOYcnd+rDm
qxenV9Ir8RqZ0XSYuUyzRka5N4dhIhrzTAiNdeU5gzynX0z67u/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

Key	Value	Details
		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.
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

Key	Value	Details
		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 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 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 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

Chapter 5 Post-Deployment Operations

This chapter describes the operation after deploying the container.

5.1 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	<code>.metadata.name</code>	Name of Cluster
COUNT	<code>.spec.fep.instances</code>	Number of FEP nodes in the cluster

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
<code>.spec.fep.instances: <i>n</i></code>	Increase the number of nodes in the cluster to <i>n</i> .
<code>.spec.fep.image.image:</code> <code>'quay.io/fujitsu/fujitsu-enterprise-postgres-13-server:ubi8-13-1.1'</code>	Minor upgrade of FEP image to ubi8-13-1.1.
<code>spec.fepChildCrVal.backup.image.image:</code> <code>'quay.io/fujitsu/fujitsu-enterprise-postgres-13-backup:ubi8-13-1.1'</code>	Minor upgrade of Backup image to ubi8-13-1.1.

This will impact behaviour for values in `fep` 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, FEPClusterConfig, FEPCert & FEPUser) & resources associated with it.



Deleting a FEPCluster will delete all PV associated with the cluster, including backup and archived WAL volumes (unless using AWS S3). This is an unrecoverable action.

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 the "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 FEPCluster Resource Change

5.2.1 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.fep.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.

5.2.2 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.fepChildCrVal.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.

5.3 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 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 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.

Custom Resource spec	Change Effect
.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.

5.4 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): 000000010000000000000001/000000010000000000000005

  full backup: 20201125-025043F
    timestamp start/stop: 2020-11-25 02:50:43 / 2020-11-25 02:50:52
    wal start/stop: 000000010000000000000003 / 000000010000000000000003
    database size: 31.7MB, backup size: 31.7MB
    repository size: 3.9MB, repository backup size: 3.9MB

  incr backup: 20201125-025043F_20201125-025600I
    timestamp start/stop: 2020-11-25 02:56:00 / 2020-11-25 02:56:02
    wal start/stop: 000000010000000000000005 / 000000010000000000000005
    database size: 31.7MB, backup size: 24.3KB
    repository size: 3.9MB, repository backup size: 619B
    backup reference list: 20201125-025043F
```

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

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)

5.5 Configure MTLs Setting

5.5.1 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.

5.6 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.

5.6.1 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	You can monitor the following resource status <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.

5.6.2 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.

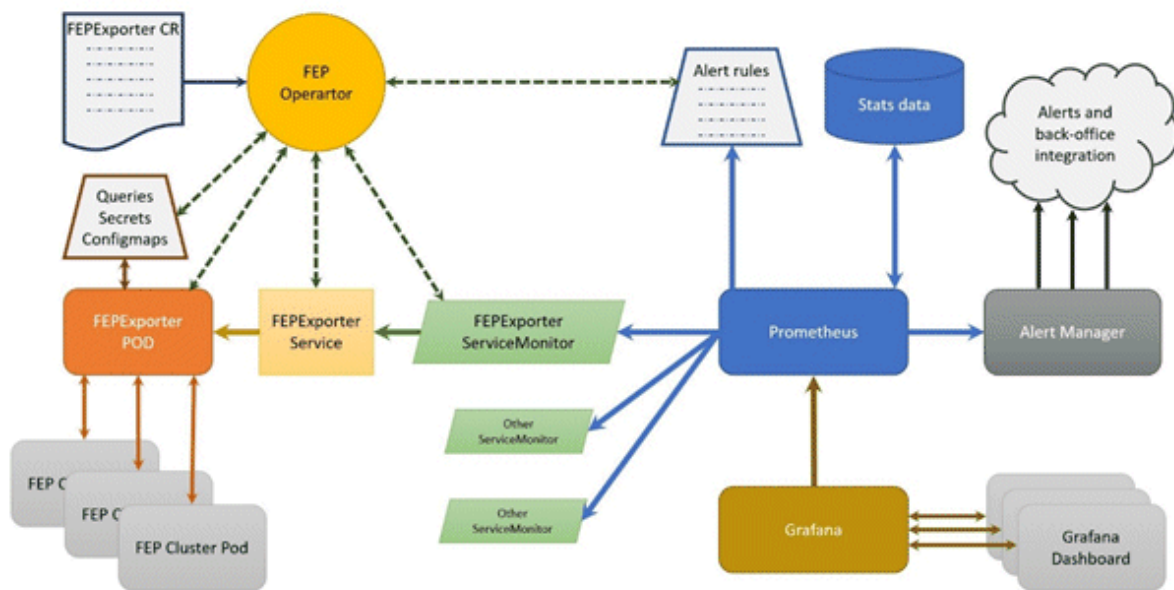
The Prometheus acts as storage and a polling consumer for the time-series data of FEP container. Grafana queries Prometheus to displaying informative and very pretty graphs.

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.

5.6.2.1 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.

5.6.2.2 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
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_tblbloat_*	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.

The screenshot shows the Red Hat OpenShift Monitoring interface. On the left, the 'Monitoring' menu is expanded, and 'Metrics' is selected. The main area displays a table of metrics. One row is highlighted with a red circle, showing the following details:

Icon	Metric Name	Source	Labels	Unit	Value	Status
Blue square	pg_stat_activity_count	prometheus-fp-exporter	mydb, new-Rep-exporter-http, 10/31/07/9187, new-Rep-exporter-service, 12devid, new-Rep-exporter-deployment-7569c99s7-jmrc		2	idle

5.6.2.3 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
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.

5.6.2.4 Graphical user interface

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

An example Grafana dashboard screenshot is shown below



5.6.3 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 `feh_exporter` schema in the postgres database on the FEP server.

Table/View name	Details
pgbackrest_info_backup	Backup Processing Status

5.6.3.1 pgbackrest_info_backup view

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

Column	Type	Description
label	text	Information identifying the backup
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

5.6.4 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 fep_exporter schema 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

5.6.4.1 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

5.6.4.2 pgpool2_stat_conn_pool view

Indicates the state of the connection pool. Contains connection pool information for each pcpool2 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
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

5.6.4.3 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

5.7 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.

5.7.1 Events raised

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

5.7.2 Viewing the custom events

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

1. On cli

Executing the command

kubectl get events

OR

oc get events

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

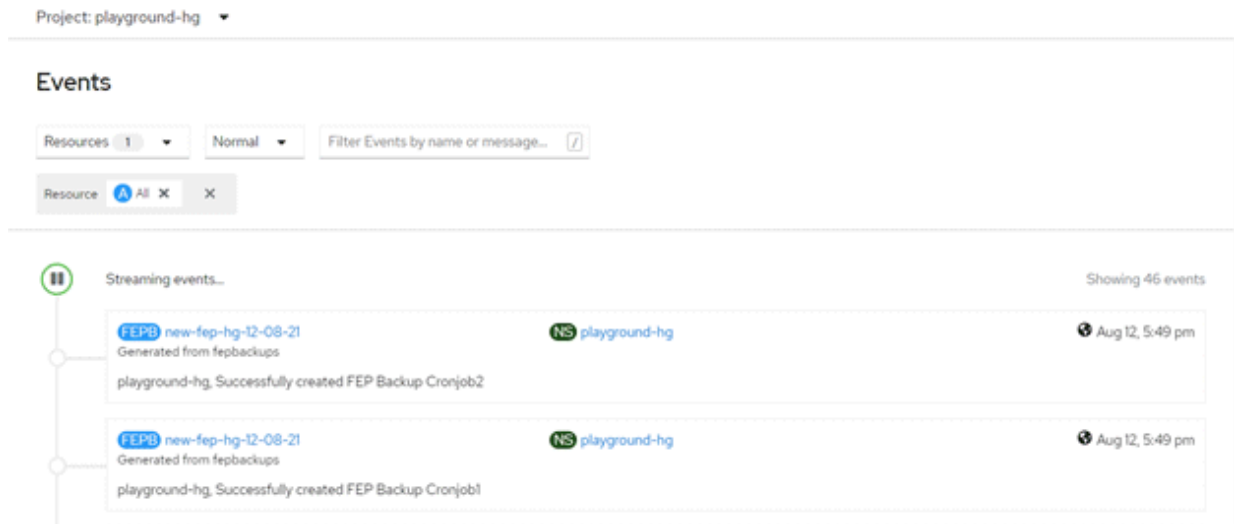
```

14m Normal InitiatedChildCRCreate feplistner/new-fep-hg-12-08-21 playground-hg, Started FEP Volume CR creation
13m Normal InitiatedChildCRCreate feplistner/new-fep-hg-12-08-21 playground-hg, Started FEP User CR creation
13m Normal InitiatedChildCRCreate feplistner/new-fep-hg-12-08-21 playground-hg, Started FEP Cert CR creation
13m Normal SuccessfulFepVolumeCreate feplistner/new-fep-hg-12-08-21 playground-hg, Successfully created FEP Volume
13m Normal SuccessfulFepUserCreate feplistner/new-fep-hg-12-08-21 playground-hg, Successfully created FEP User
13m Normal SuccessfulFepCertCreate feplistner/new-fep-hg-12-08-21 playground-hg, Successfully created FEP Cert
13m Normal SuccessfulFepConfigCreate feplistner/new-fep-hg-12-08-21 playground-hg, Successfully created FEP Config
13m Normal SuccessfulFepBackupCronjob1Create feplistner/new-fep-hg-12-08-21 playground-hg, Successfully created FEP Backup Cronjob1
13m Normal SuccessfulFepBackupCronjob2Create feplistner/new-fep-hg-12-08-21 playground-hg, Successfully created FEP Backup Cronjob2
13m Normal SuccessfulFepVolumeCreate feplistner/new-fep-hg-12-08-21 playground-hg, Successfully created FEP Volume

```

2. 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.



5.8 Scaling Replicas

5.8.1 Auto Scale Out

Auto-scale-out occurs when the average CPU utilization of the DB container exceeds the threshold.

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

Specify `spec.fepChildCrVal.autoscale.scaleout` in `FEPClusterCR` when you want to perform Auto scale out.

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

5.8.2 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 feplistner <FEPClusterCR name>
```


 Note

Do not scale in from two to one replica instance when the syncMode is 'on'. Update SQL cannot be executed.

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.



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

Provides major version upgrade procedures for operators and containers.

6.4.1 Install a New Version of the Operator

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

The Namespace you install can be different from the older version of the operator you are upgrading.

6.4.2 Deploy a New Version of a Container

See "[Chapter 4 Deployment Container](#)" to deploy the FEP container.

If necessary, deploy the Pgpool2 container.

6.4.3 Migrating Data Between Containers

Database data migration requires business application outages. Use the pre-verification to understand the migration time in this section and estimate the downtime.

1. Stop the operation to the old DB container in operation.
2. Extract database data from the old database container. Run on the database client where the client corresponding to the old database engine is installed. Get the database data, schema from the old database container by executing the following command.

Example)

```
$CLIENT_DIR=[Directory path where the database engine client was installed]
$export PATH=${CLIENT_DIR}/bin:${PATH}
$export LD_LIBRARY_PATH=${CLIENT_DIR}/lib:${LD_LIBRARY_PATH}
$IP=[Destination IP address of the old DB container]
$PORT=[ Port number of the old DB container]
$pg_dumpall -h ${IP} -p ${PORT} -U postgres > db.dump
```



Check the database size in advance by connecting to database and executing SQL below.

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

Since the pg_dumpall command used above outputs the data of the database as an SQL command, the file actually created is:

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

However, since SQL commands output them as strings, this is 10 bytes. Therefore, run it in a location where there is sufficient disk space.

3. Populate the new DB container with data.

Run the new database engine client on the installed database client.

Populate the new DB container deployed in "6.4.2 Deploy a New Version of a Container" with the schema and data extracted from the old DB container using the `pg_dump` command.

Example)

```
$CLIENT_DIR=[ Directory path where the database engine client was installed]
$export PATH=${CLIENT_DIR}/bin:${PATH}
$export LD_LIBRARY_PATH=${CLIENT_DIR}/lib:${LD_LIBRARY_PATH}
$IP=[Destination IP address of the new DB container]
$PORT=[ Port number of the new DB container]
$psql -h ${IP} -p ${PORT} -U postgres -d postgres -f db.dump
```

 Note

When database data is input, a message that the following extension registration failed is output, but ignore this message.

- pgx_cpu
- pgx_disk
- pgx_io
- pgx_log
- pgx_memory
- pgx_network
- pgx_network_err
- pgx_paging
- pgx_process
- pg_stat_statements

4. Change the destination configured for the application to the new DB container or the new Pgpool2 container.

 Note

- For more information about the `pg_dumpall` command, see the following in the source database engine documentation ("PostgreSQL 13 documentation").
*"VI. Reference" - "II. PostgreSQL Client Applications" - "pg_dumpall"
- For more information on the `psql` command, see the destination database in the documentation ("PostgreSQL 13 documentation").
*"VI. Reference" - "II. PostgreSQL Client Applications" - "psql"

6.4.4 Removing Old Containers

If necessary, remove the old DB container.

"Operators" > "Installed Operators" > "FUJITSU Enterprise Postgres <Old version> Operator" > "FEPCluster" > "FEPCluster name to delete" > Choose "Delete FEPCluster" from Actions

6.4.5 Uninstalling Old Operators

Uninstall the old FEP operator if necessary.

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

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.

Edit the ClusterServiceVersion (CSV) to change the resources assigned to the operator container.

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].

The screenshot shows the OCP GUI console interface. On the left, a sidebar menu is visible with 'Installed Operators' selected. The main content area displays the 'Operator details' for 'FUJITSU Enterprise Postgres 13 Operator'. The 'YAML' tab is active, showing the configuration for the operator container. The 'resources' section is expanded, showing the following configuration:

```
481     vendor: Fujitsu
482     spec:
483       containers:
484         - resources:
485             limits:
486               cpu: '2'
487               memory: 3072Mi
488             requests:
489               cpu: 500m
490               memory: 768Mi
491             name: fep-ansible-operator
492             livenessProbe:
493               failureThreshold: 10
```

At the bottom of the console, there are buttons for 'Save', 'Reload', 'Cancel', and 'Download'.

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-operator.v3.1.6  FUJITSU Enterprise Postgres 13 Operator  3.1.6
Succeeded
```

Edit the CSV with the "oc edit" command.

```
$ oc edit csv fujitsu-enterprise-operator.v3.1.6
```

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 backup 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 Collecting Fault 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  smzk   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 "[5.1 Configuration Change](#)" and reapply. Depending on the parameters to be changed, the container may be redeployed. Refer to "[5.1 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](#)".

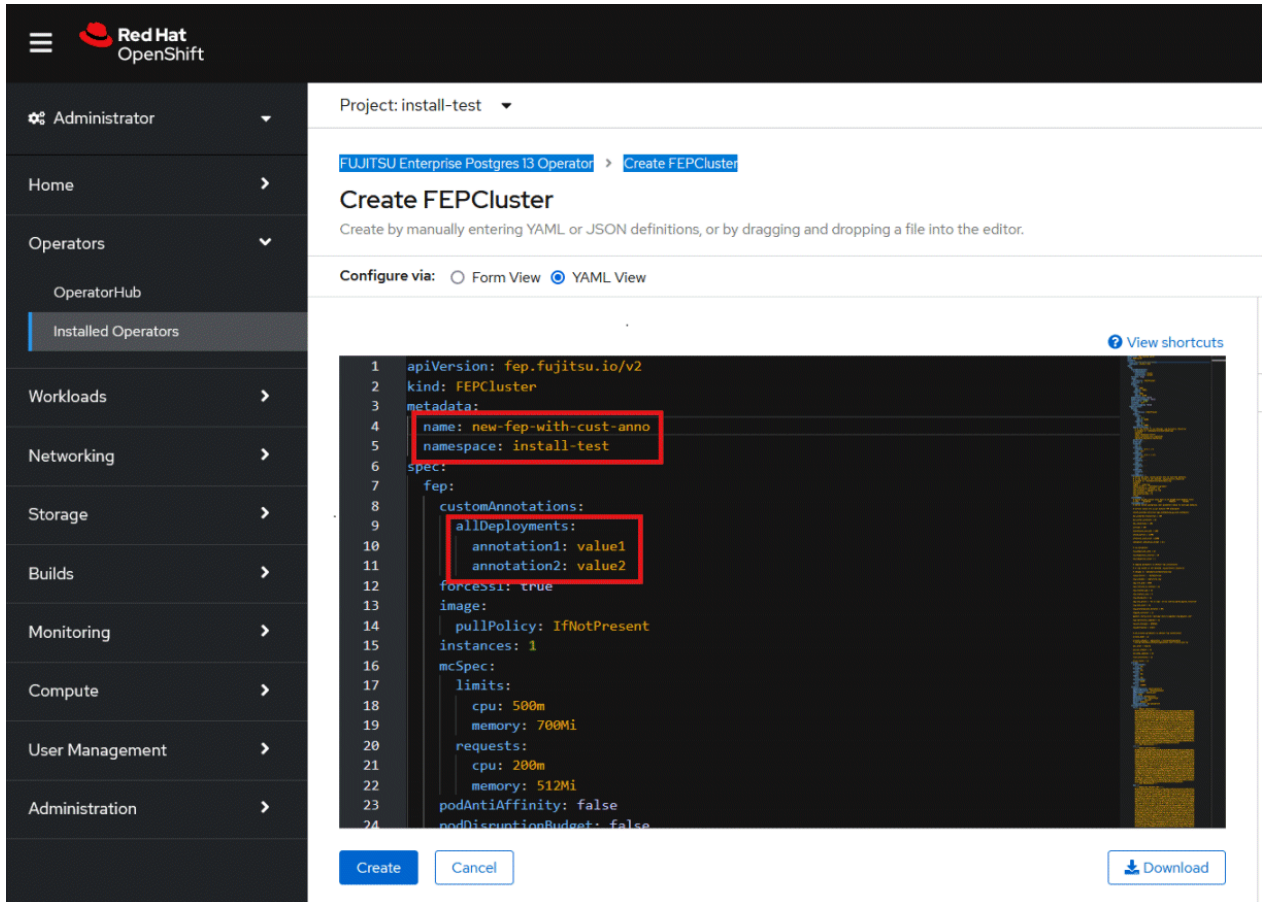
FEP features that needs to be set when using

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.



The screenshot shows the Red Hat OpenShift console interface. On the left is a navigation sidebar with options like Administrator, Home, Operators, Workloads, Networking, Storage, Builds, Monitoring, Compute, User Management, and Administration. The main content area is titled 'Create FEPCluster' and shows the 'FUJITSU Enterprise Postgres 13 Operator' breadcrumb. Below the title, it says 'Create by manually entering YAML or JSON definitions, or by dragging and dropping a file into the editor.' The 'Configure via' section has 'Form View' and 'YAML View' (selected) radio buttons. A 'View shortcuts' link is visible. The main area contains a code editor with the following YAML content:

```
1  apiVersion: fep.fujitsu.io/v2
2  kind: FEPCluster
3  metadata:
4    name: new-fep-with-cust-anno
5    namespace: install-test
6  spec:
7    fep:
8      customAnnotations:
9        allDeployments:
10         annotation1: value1
11         annotation2: value2
12     forceSSI: 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
```

At the bottom of the editor are 'Create', 'Cancel', and 'Download' buttons.

2. 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. The left sidebar contains navigation menus for Administrator, Home, Operators, Workloads, and Horizontal Pod Autoscalers. The main content area displays the details for a StatefulSet named 'new-fep-with-cust-anno-sts' in the 'install-test' project. The StatefulSet is managed by FEPC. 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: 269c888b-434d-48de-b1d4-832636ad521c
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      fieldsV1: FieldsV1
21      fieldsV1:
22        'f:metadata':
23          'f:annotations':
```

Buttons for 'Save', 'Reload', 'Cancel', and 'Download' are visible at the bottom of the YAML editor.

The screenshot shows the Red Hat OpenShift console interface. The left sidebar contains navigation menus for Administrator, Home, Operators, Workloads, and Horizontal Pod Autoscalers. The main content area displays the details for a StatefulSet named 'new-fep-with-cust-anno-sts' in the 'install-test' project. The StatefulSet is managed by FEPC. The 'YAML' tab is active, showing the pod template 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
```

Buttons for 'Save', 'Reload', 'Cancel', and 'Download' are visible at the bottom of the YAML editor.

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 ~
```