FUJITSU

**Fujitsu
Enterprise Postgres 17 for Kubernetes**

Overview

Linux

Preface

Purpose of this document

This document explains the Fujitsu Enterprise Postgres for Kubernetes concepts to those who are to operate databases using it.

This document explains the features of 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 Know about the Product

Explains the features of Fujitsu Enterprise Postgres for Kubernetes.

Chapter 2 Know What it does

Explains what you need to do.

Appendix A OSS Supported by Fujitsu Enterprise Postgres for Kubernetes

Explains the OSS supported by Fujitsu Enterprise Postgres for Kubernetes.

Abbreviations

The following abbreviations are used in this manual:

Full Name	Abbreviations
Fujitsu Enterprise Postgres for Kubernetes Fujitsu Enterprise Postgres	FEP
Custom Resource	CR
Custom Resource Definition	CRD
Grafana, Alert Manager, Prometheus	GAP
Persistent Volume	PV
Persistent Volume Claim	PVC

Abbreviations of manual titles

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

Full Manual Title	Abbreviations
Fujitsu Enterprise Postgres for Kubernetes User's Guide	User's Guide

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 ("IBM") in the U.S. and other countries.

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 2.0: March 2025
Edition 1.0: November 2024
```

Copyright

Contents

Chapter 1 Know about the Product

This chapter explains the features of Fujitsu Enterprise Postgres for Kubernetes.

1.1 What is Fujitsu Enterprise Postgres for Kubernetes?

Fujitsu Enterprise Postgres for Kubernetes provides automated operations for installing and managing your Fujitsu Enterprise Postgres 17 on OpenShift Container Platform or Kubernetes.

There are multiple components in the solution.

FEP operator: Manages the lifecycle of FEP server container, including deployment, configuration update, backup and recovery of FEP database.

FEP server container: Contains the FEP server software to run the Postgres engine.

FEP backup container: Contains the FEP server software to perform scheduled backup operations.

FEP restore container: Contains the FEP server software to perform the restore operation.

FEP pgpool2 container: Contains the FEP server software to use Pgpool-II to provide load balancing and connection pooling.

FEP exporter container: expose various health metrics to Prometheus for monitoring

Up and running in minutes, the operator provides the features required to maximise the benefits of this enterprise PostgreSQL solution.
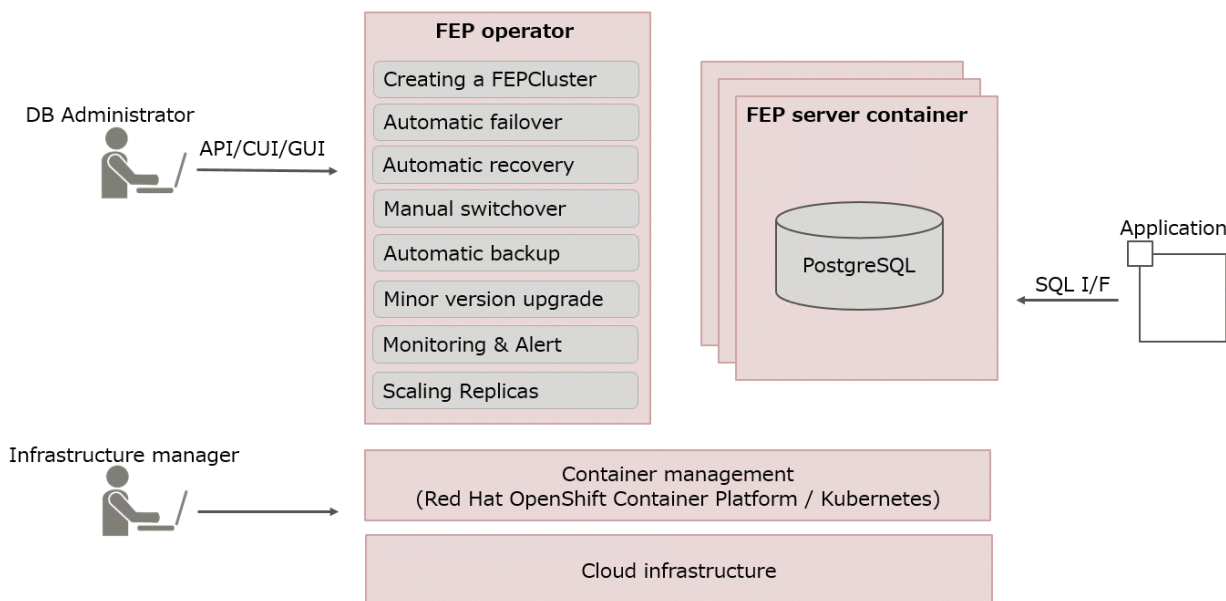
This operator will deploy a standalone as well as highly available Fujitsu Enterprise Postgres for Kubernetes provides automated operations for installing and managing your Fujitsu Enterprise Postgres 17 on OpenShift Container Platform or Kubernetes.

Enterprise Postgres cluster with pre-defined configuration to get started with small workload. User can adjust the configuration parameters at the time of deployment and after to make the instance suitable for the workload.

As the name implies, the FEP server container is intended to incorporate the Fujitsu Enterprise Postgres for Kubernetes provides automated operations for installing and managing your Fujitsu Enterprise Postgres 17 on OpenShift Container Platform or Kubernetes.

Enterprise Postgres server component.

In principle, a running FEP server container is considered as equivalent to a Fujitsu Enterprise Postgres for Kubernetes provides automated operations for installing and managing your Fujitsu Enterprise Postgres 17 on OpenShift Container Platform or Kubernetes.

Enterprise Postgres Server instance.

1.2 Operator Features

This product provides operator services to automate the construction and operation of databases on the customer's container management infrastructure. The features of the operator are as follows:

- Cluster Deployment

    - Creating a FEPCluster

    - Creating a FEP Pgpool2 Container

- Highly Available Feature

    - Automatic Failover

    - Automatic Recovery

    - Manual Switchover

- Backup Recovery

    - Automatic Backup

    - Point-in-time Recovery

- Configuration Change

    - Parameter Change

    - Resource Change

    - Disk Expansion

- Minor Version Upgrade

- FEP Features

- Automating Audit Log Operations

- Monitoring & Alert

- Collaboration with Amazon CloudWatch

- Scaling Replicas

- Disaster Recovery

- Transparent Data Encryption Using Key Management System

- Fixed Statistics

- Protection, efficient management and utilization of knowledge data for AI applications

1.2.1 Cluster Deployment

1.2.1.1 Creating a FEPCluster

Users can instruct the operator to build a system that includes the provisioning of containers and volumes with FEP installed, and network resources. The resulting system is called a FEPCluster. The FEPCluster can be created a single master server or multi-servers with one master and two replicas. You can choose between synchronous and asynchronous replication replica servers. The default is asynchronous replication.

FEPCluster is composed of the following components:

- FEP server container

    - FEP server

    - Patroni

- FEP backup container

- CR FEPVolume for volumes

- CR FEPUser for database users

- CR FEPConfig for Postgres configuration

- CR FEPCert for secrets such as TLS certificate, keystore passphrase

The Below diagram depicts a FEPCluster with one Master and one Replica POD.
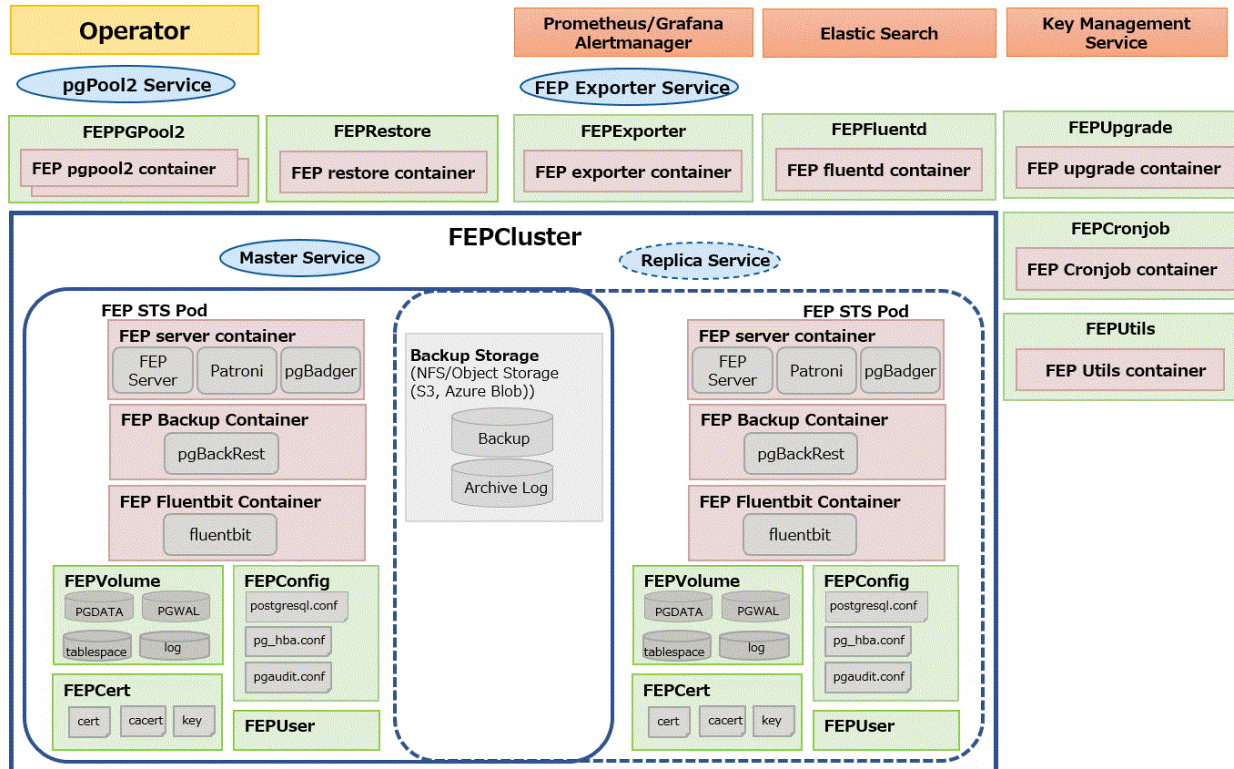
1.2.1.2 Creating a FEP Pgpool2 Container

Users can deploy Pgpool-II for load balancing and connection pooling with FEP pgpool2 container.

Users can deploy multiple FEP pgpool2 Pods in a single deployment to increase availability.

1.2.2 Highly Available Feature

1.2.2.1 Automatic Failover

When an error is detected in the container or POD of the master server, the cluster will perform an automatic failover by promoting one of the replicas to become the new master, and the connection destination of the database is switched. The database connection is broken, but you can reconnect by establishing a connection from the application again.

1.2.2.2 Automatic Recovery

If an error occurs on the master server and an automatic failover occurs, the POD or container of the failed old master server is automatically restarted and reincorporated into the cluster as a replica server.

If a replica server fails, it automatically restarts and rejoins the cluster as a replica server.

1.2.2.3 Manual Switchover

You can manually switch any replica server to the master server. In this case, the original master server becomes the replica server.

1.2.3 Backup Recovery

1.2.3.1 Automatic Backup

By taking regular backups, you can be prepared for full database downtime or data corruption due to application errors. Users can set an arbitrary schedule for automatic backup. The backup type can be a full backup or an incremental backup. You can back up the database to shared storage such as NFS persistent volume or AWS S3 compatible storage. Backups can be automatically deleted by setting a retention period of your choice.

1.2.3.2 Point-in-time Recovery

Point-in-time recovery can be used to recover data at specific times due to business failures or to replicate a cluster for migration to production environment. Allows point-in-time recovery from automated backup data to restore the cluster. You can choose between restoring data to an existing cluster and a new cluster. You can also choose to restore to the most recent data or to any time you specify.

1.2.4 Configuration Change

1.2.4.1 Parameter Change

You can change the parameters that make up the FEP. PostgreSQL provides two types of parameters: those that take effect immediately, and those that take effect after restarting FEP server process.

- postgresql.conf

- pg_hba.conf

- pgaudit.conf

📝 Note

For parameters that take effect immediately, operator will apply the change to all FEP Pods and reload the FEP server process automatically. There is no outage on the cluster.

For parameters that take effect after restarting FEP server process, operator will update the configuration files on all FEP Pods. However, users have to initiate a manual restart of FEP process on all the FEP Pods using the FEPAction CR. There is a momentary outage on the cluster and the users should perform this action at a time that has least disruption to the service.

1.2.4.2 Resource Change

You can change the amount of CPU and memory resources allocated to FEP server containers, FEP backup containers, or FEP pgpool2 containers by changing the FEPCluster CR. The operator will apply the change to the Statefulset. However, the users have to perform a restart of all the Pods for the new resource allocation to take effect.

📝 Note

Changing resource allocation will not take effect immediately. The users have to restart all the Pods for new resource allocation to take effect. There is a momentary outage on the cluster and the users should perform this action at a time that has least disruption to the service.

1.2.4.3 Disk Expansion

You can extend the disk size mounted on the FEP server container by modifying the FEPCluster custom resource.

Disk expansion can also work with Prometheus to automatically expand based on disk usage.

Disk extensions must take advantage of disks that support the Kubernetes PVC extension.

1.2.5 Minor Version Upgrade

1.2.5.1 Minor Version Upgrade

New and patched FEP releases are made available as new container image. When the latest container image is provided, the user can perform a minor version upgrade by changing the FEPCluster CR. The operator will perform a rolling update to enable the minor version upgrade with minimal system disruption.

Note

....
The minor version upgrade will take effect immediately. There is a momentary outage on the cluster and the users should perform this action at a time that has least disruption to the service.
....

1.2.6 FEP Features

1.2.6.1 Scope of FEP Feature Support

Note

....
These features also require the FEP Client.
....

The FEPCluster that is created supports the following features in addition to the PostgreSQL features of OSS. Enhances security and performance with transparent data encryption to prevent data loss in the event of database storage theft and in-memory capabilities with column-type index and data memory resident features to speed aggregation. Details of each feature can be found in the FEP documentation.

For transparent data encryption using a key management system, see "1.2.12 Transparent Data Encryption Using Key Management System".

Category	Feature
Operation	pgAdmin
	Global Meta Cache
Security	Transparent Data Encryption
	Audit Log
	Data Masking
High Performance	In-memory feature
	High-speed data load
Application Interface	Java Integration
	ODBC Integration
	.NET Framework Integration
	Embedded SQL Integration (C language)
	Embedded SQL Integration (COBOL)
	Python language (psycopg) Integration

1.2.7 Automating Audit Log Operations

As a Database Administrator you want to be able to perform audit schema changes or as a Security Officer, you want to be able to audit login/logout events, successful or failure, it is as part of company policy, you need to store auditlogs on external system for period of time.

As a Database Administrator, specify the following

- When the pgaudit function is enabled with the enable parameter in FEPCluster, the following is automatically set by the operator.

    - pgaudit is added in 'shared_preload_libraries'

    - pgaudit log directory is configured

    - pgaudit file name is configured

    - pgaudit extension is created

- Specify an external ConfigMap for pgaudit configuration.

    - This ConfigMap contains full content of pgaudit.conf

    - Both session audit and object audit can be configured

    - The configurations in this ConfigMap overwrites the pgaudit configuration specified in FEPCluster CR

- Specify a destination so that the pgaudit log files can be uploaded periodically.

    - web server

    - Azure Blob

    - AWS S3 storage

Note

Azure blob is not supported on s390x platform.

1.2.8 Monitoring & Alert

1.2.8.1 Monitoring

Infrastructure administrator can start monitoring database almost simultaneously with database construction with standard monitoring tools.

Evaluation indicator data from a database point of view is provided in a format that can be displayed in Prometheus and Grafana.

The monitoring items are as follows:

- Database health

- OS performance information

- Disk usage

- Backup status

- Client connection information

- Server Log

- Audit Log

1.2.8.2 Alert and Event

Alerts enable infrastructure administrator to immediately understand and address anomalies. Define anomalous conditions from Monitoring's Matrix and set notifications in Prometheus. It is possible to integrate alerts with other services like emails, slack, sms or back-office systems for communication and action.

Perform recovery processing at the application layer after failover, synchronize with database backup, perform application backup, etc.

1.2.9 Collaboration with Amazon CloudWatch

You can work with CloudWatch, the Amazon Web Services(Hereinafter, AWS is used as an abbreviation.) monitoring service, to monitor FEP metrics and collect logs. The operator feature can automatically forward FEP metrics and logs to CloudWatch. The FEP and services

provided by AWS, such as EC2 and S3 object storage built into business systems, can be centrally monitored, reducing the workload of monitoring tasks.

1.2.10 Scaling Replicas

You can dynamically expand a read replica depending on the load on the read replica.

1.2.10.1 Automatic Scale out

With automatic scale out, the operator automatically extends the read replica according to the policy you specify.

The available policies are controlled by the CPU load or number of connections of read replica instance to automatically extend beyond a specified threshold.

Automatic scale out eliminates the need to use unnecessary resources for maximum potential load. It also reduces manual operations as the load increases.

📝 Note

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

- Even if the automatic scale out feature increases the number of replicas, incoming requests are not given priority to those replicas. As a result, existing FEP instances may continue to be temporarily overloaded after the number of replicas increases.

- The automatic scale out feature increases the number of replica requests that can be handled only by reference requests to the database. Requests with updates continue to be processed on the primary FEP instance. Therefore, the autoscale out feature may not reduce the load on the primary FEP instance.

- Currently, the automatic scale out feature does not delete replicas (reduce the number of replicas). If the load decreases after the number of replicas increases due to a temporary increase in load, the number of replicas remains increased. If necessary, manually change the number of replicas.

1.2.10.2 Manual Scale in/out

You can scale out or scale in the read replica at any time. This can be done by manipulating the CR of the FEPCluster.

1.2.11 Disaster Recovery

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

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

There are three disaster recovery methods available:

- Backup/Restore method

  Store backups of production environment data in object storage and restore data from object storage after a disaster.

- Continuous recovery method

  Create a container environment for production and another one for disaster recovery. Store the production environment data in object storage, and continuously restore to the disaster recovery environment.

- Streaming replication method

  Similarly to the continuous recovery method, create a container environment for production and another one for disaster recovery. Use streaming replication methods to synchronize data to the disaster recovery environment.

1.2.12 Transparent Data Encryption Using Key Management System

Key management system can be used to manage the master encryption key used in the transparent data encryption function. By using a key management system, you can centrally manage encryption keys according to your business security requirements.

The available key management systems for operators are:

- key management server using the KMIP protocol

- AWS key management service

- Azure key management service

Note
....
AWS and Azure key management services are only available in environments with a x86 CPU architecture.
....

1.2.13 Fixed Statistics

To prevent performance instability due to variations in execution plans, statistics can be fixed. By scheduling the fixation of statistics, regular statistics can be fixed without any burden on infrastructure administrators.

1.2.14 Protection, efficient management and utilization of knowledge data for AI applications

Generative AI applications, where accuracy and reliability are critical, often use an approach called RAG that leverages external data sources to improve the accuracy of the generated AI responses and make use of expertise and up-to-date information. Fujitsu Enterprise Postgres provides knowledge data management capabilities for using Fujitsu Enterprise Postgres as an external data source in the RAG approach.

Knowledge in the RAG approach is expressed in text and graph form, and vector data is also leveraged for fast retrieval. Knowledge data management allows you to centralize the management of knowledge data such as vectors and graphs in Fujitsu Enterprise Postgres. In addition to eliminating the need for a dedicated database for each data type, you can protect your knowledge data with the reliable and secure features of Fujitsu Enterprise Postgres.

For more information, refer to "Setup of Knowledge Data Management" in the User's Guide.

1.3 Operator System Configuration

The basic relationships among POD, containers and services are as follows.

Example) Deployment with Pgpool-II

In this deployment scenario, Pgpool-II is used to provide connection pooling and load balancing. End user application will point its connection to Pgpool service. Depending on the transaction type, Pgpool will forward the connection to either the Master Pod or the Replica Pod. If a failover/switchover occurs, the FEP pgpool2 will direct traffic to the new FEP master Pod. This is transparent to the end user application.
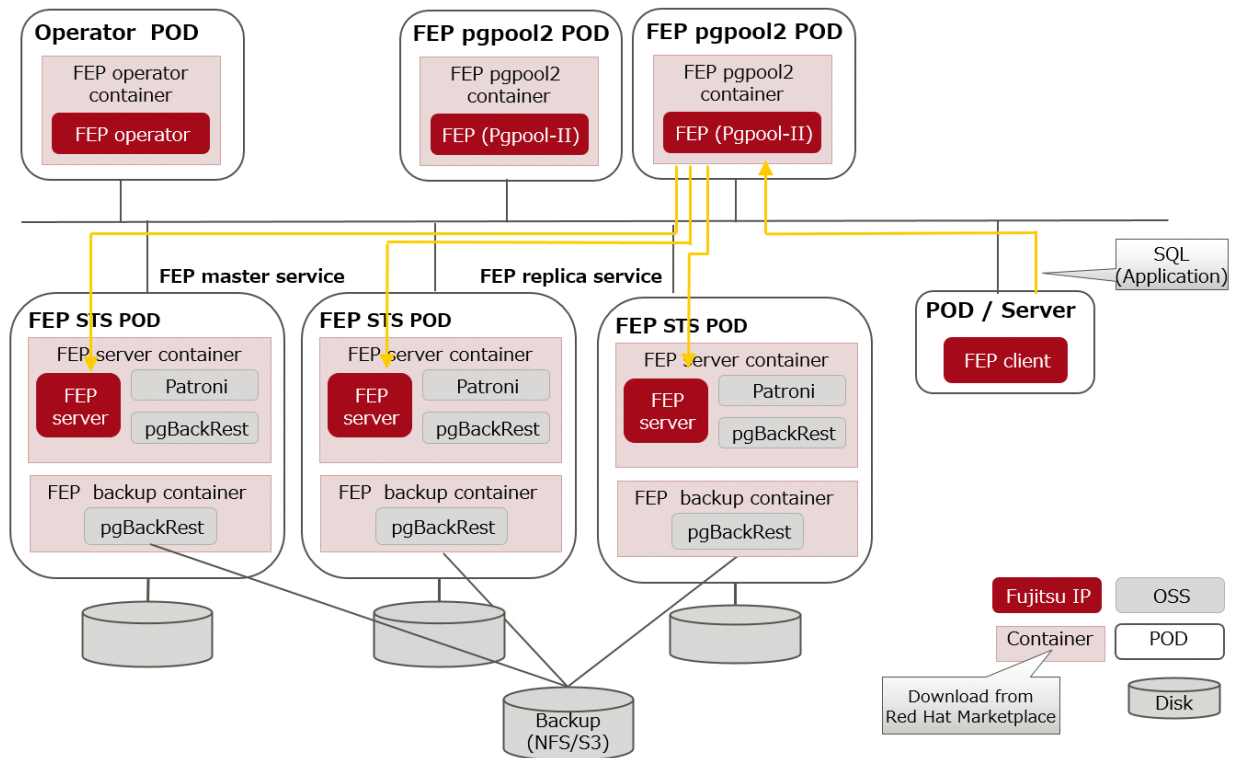
Example) Deployment without Pgpool-II

Users can also run applications such as SQL directly against the FEPCluster without configuring Pgpool-II. In this deployment scenario, end user application will point its connection to the FEP master service. If a failover/switchover occurs, the FEP master service will point to the new FEP master Pod automatically. The end user application will experience a disconnection. When it re-establishes the connection, it will be connected to the new FEP master Pod. There is no need to reconfigure the application connection string.
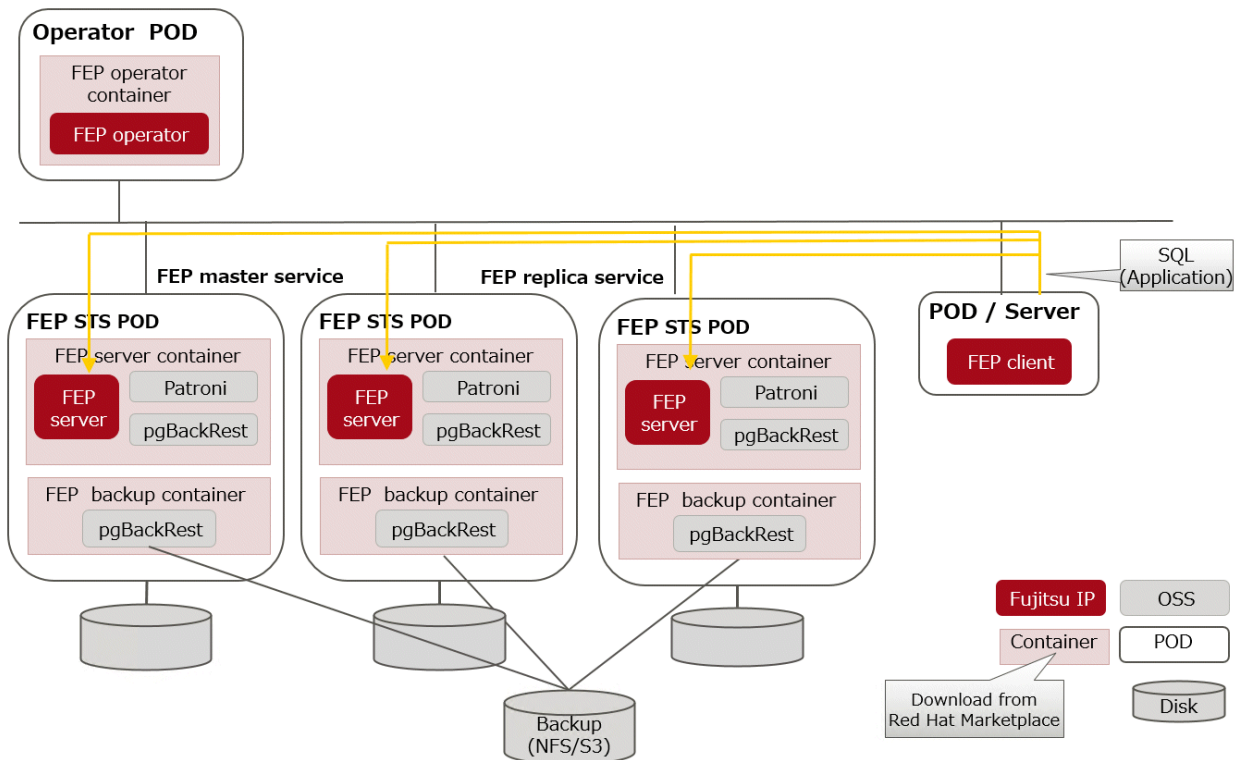
Chapter 2 Know What it does

This chapter explains what you need to do.

2.1 Deployment

FEP operator is responsible for the lifecycle of FEPCluster. The operator will deploy a HA FEPCluster, together with all the associated containers such as backup container.

2.2 High Availability (Automatic failover and recovery)

The high availability and failover management of FEP is provided by Patroni. Both Patroni and FEP will be installed on the same container image. Patroni will then initialize and start an FEP instance. Patroni will then acquire a lock on a shared resource. In our case, it is a Kubernetes configmap. Whichever POD that can acquire the lock will become the Master. When subsequent FEP server container starts, Patroni will initialize that POD as a Replica with streaming replication.
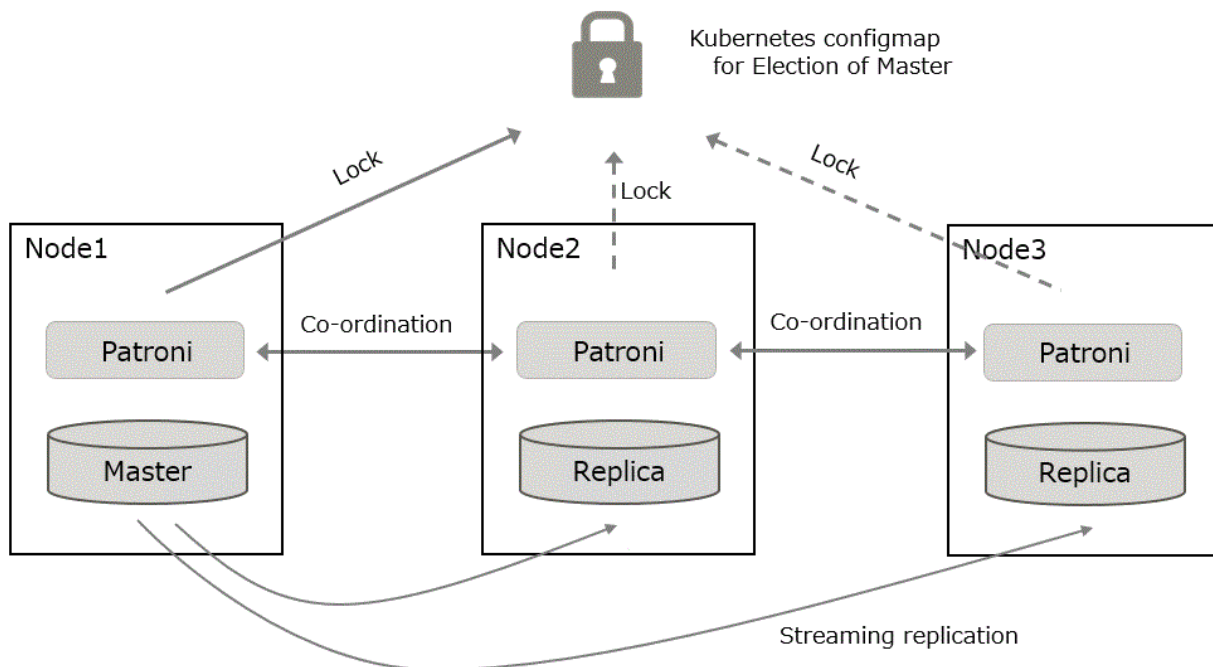
If Patroni detects a failure in the cluster, either because the Postgres process crashed or the container where Postgres is running dies, Patroni will initiate a failover automatically.

2.3 Configuration Change

Traditionally, changing FEP configurations such as postgresql.conf, pg_hba.conf, TLS certificates and keystore passphrase will require a redeployment of FEP server container. That causes an outage in a Highly Available environment.

A new CRD FEPConfig is defined to encapsulate those configurations. The operator will monitor the CR with this CRD definition and perform action accordingly to minimize outages. For example, operator will reload FEP daemon, instead of redeploying the FEP server container when a reloadable postgresql.conf parameter is changed. If a parameter change requires restart of FEP (e.g. max_connections), the operator will update the configuration file but defer the restart. End user can follow a defined procedure to restart the cluster manually at a scheduled maintenance time.

2.4 Upgrade

2.4.1 Minor Version Upgrade

FEP version Minor upgrade is done by updating the Custom Resource with a new FEP image name. The POD will be redeployed with new image in a controlled manner. First, replica servers are upgraded, restarted and waited to be ready, one server at a time. When all replicas are upgraded, a controlled switchover is performed to pick a new master. Once that is done, the old master is upgraded as well.

2.4.2 Major Version Upgrade

You can upgrade a major version of FEP by specifying the upgrade parameters when you create the latest FEP cluster. Automates the process of dumping data from an older FEP cluster and restoring data to the new, latest cluster.

Application outages are required during major version upgrades.

2.5 Configurable Volume Per Cluster

To improve performance, may want to separate the volume storing database files and WAL files. Similarly, one may want to use a dedicated volume for a new tablespace. The operator gives the end user the flexibility to create a FEPCluster with multiple PVs and select a suitable storage class for the PV. For example, one can create a FEPCluster with data volume, wal volume on a storage class backed up by SSD and a log volume on a storage class backed up by HDD.

2.6 Deploying Pgpool-II and Connect to FEPCluster from Operator

Users can deploy the FEP pgpool2 container and access the database via Pgpool-II to use load-balancing and connection pooling features.

Multiple FEP pgpool2 containers can be deployed for load-share and high availability. Users can request a Kubernetes service to distribute their work across multiple FEP pgpool2 containers.
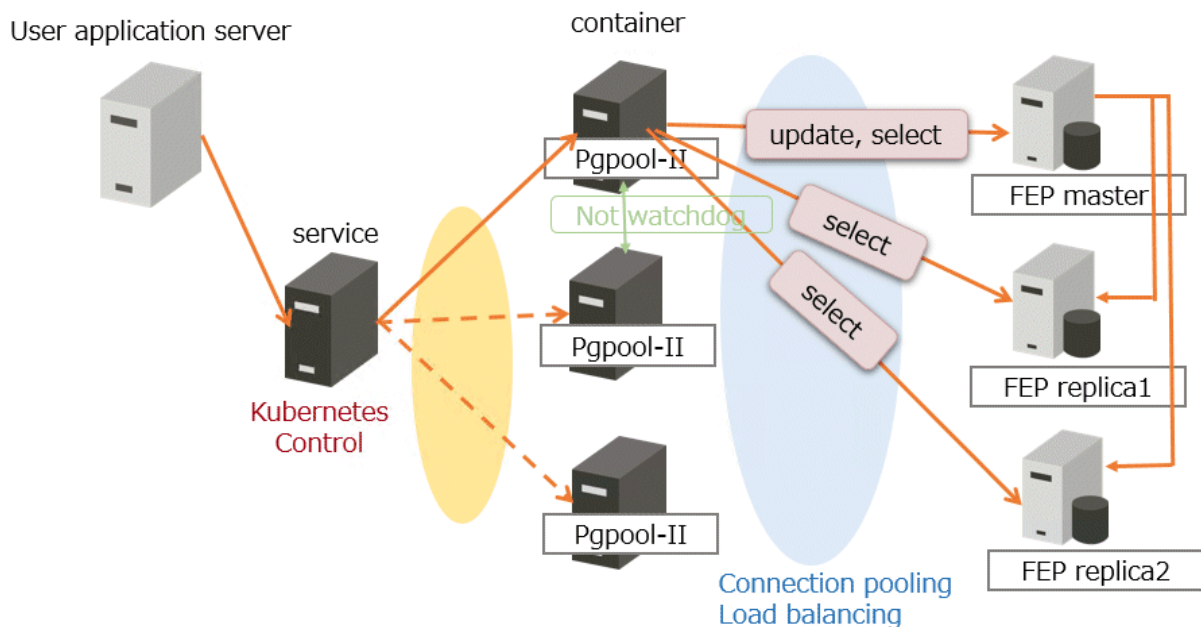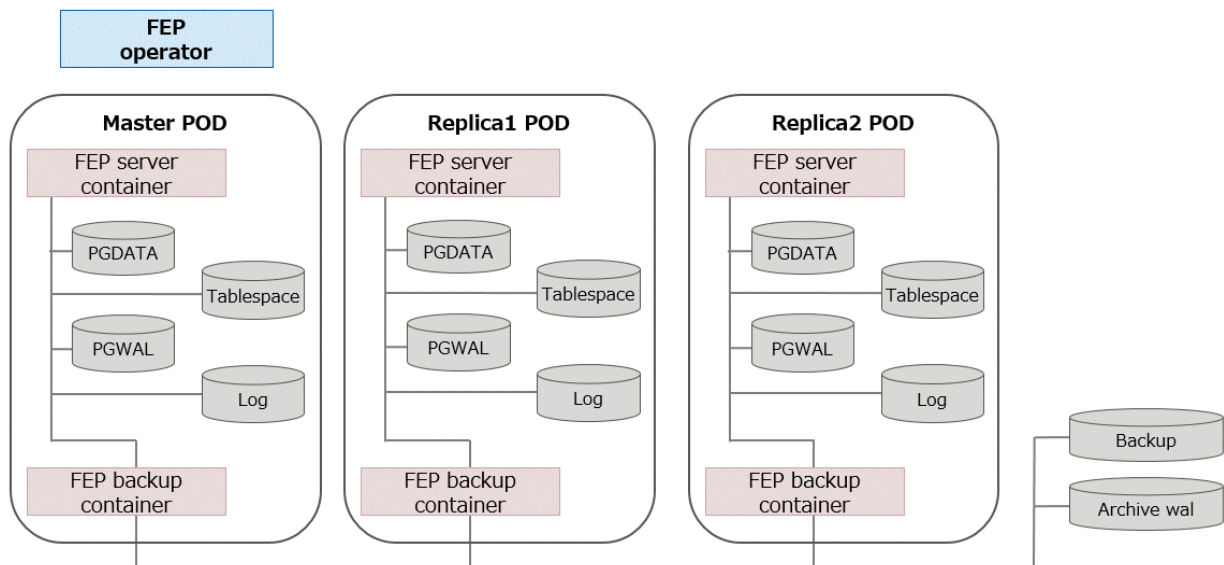
2.7 Backup

2.7.1 Scheduling Backup from Operator

The FEP backup container is deployed as a sidecar to each FEP server POD. The backup is performed at scheduled time set by the user (like crontab). The FEP backup container determines if the FEP server in the POD is a master or replica, and will perform the backup process only on the master POD. The volume storing backup and archived WAL files must be on a shared storage such as NFS or AWS S3.

Backup and WAL archiving is accomplished with pgBackRest.

2.7.2 On-Demand Backup

On-demand backups can also be taken at any time other than a predetermined schedule, such as before planned maintenance or after configuration changes.

The backup storage location and retention period settings are the same as those for scheduled backups.

2.8 Perform PITR and Latest Backup Restore from Operator

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

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

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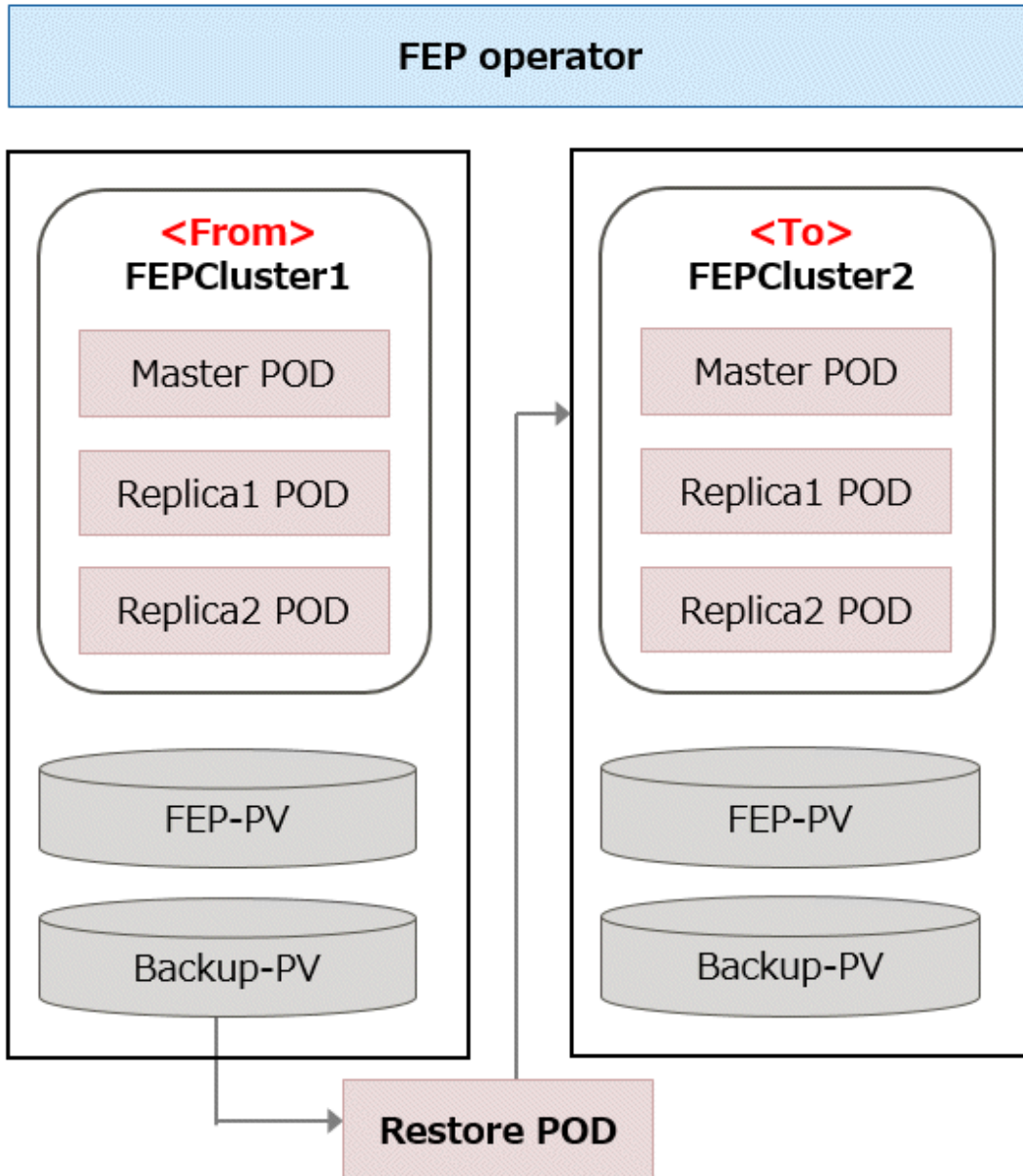
2.9 Monitoring & Alert

Monitoring and alerts system leverages standard GAP stack (Grafana, Alert manager, Prometheus) deployed on OCP(OpenShift Container Platform) 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 and guages 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 metrices 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.
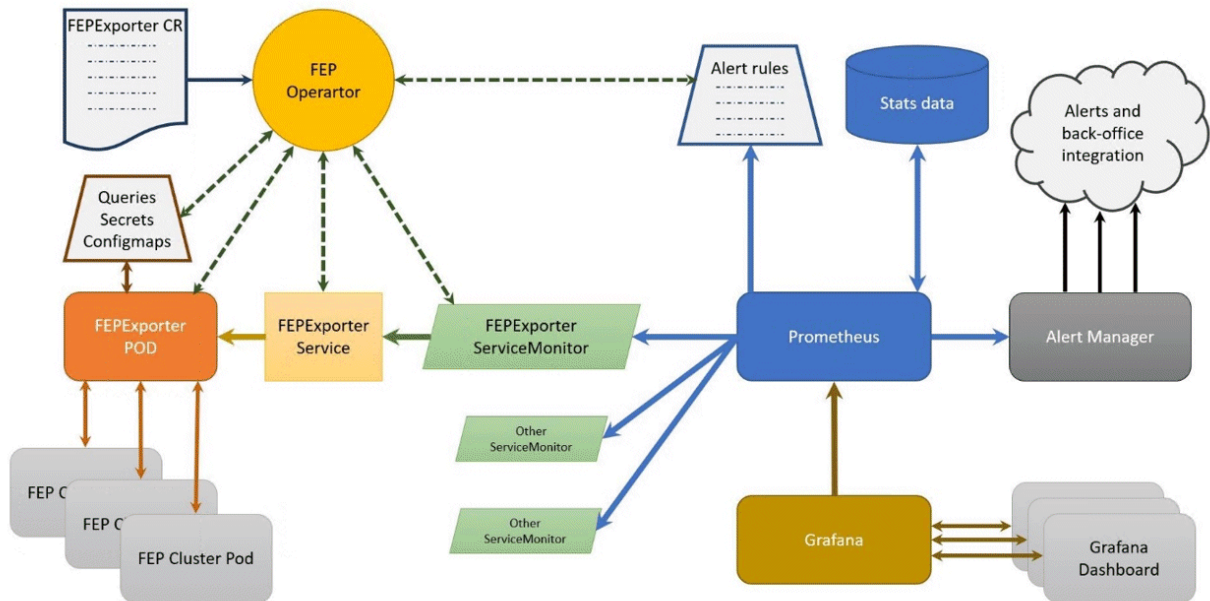
2.10 Server Log Monitoring

2.10.1 pgBadger Log Monitoring

pgBadger can parse PostgreSQL log files to produce daily and weekly statistical reports from many points of view.
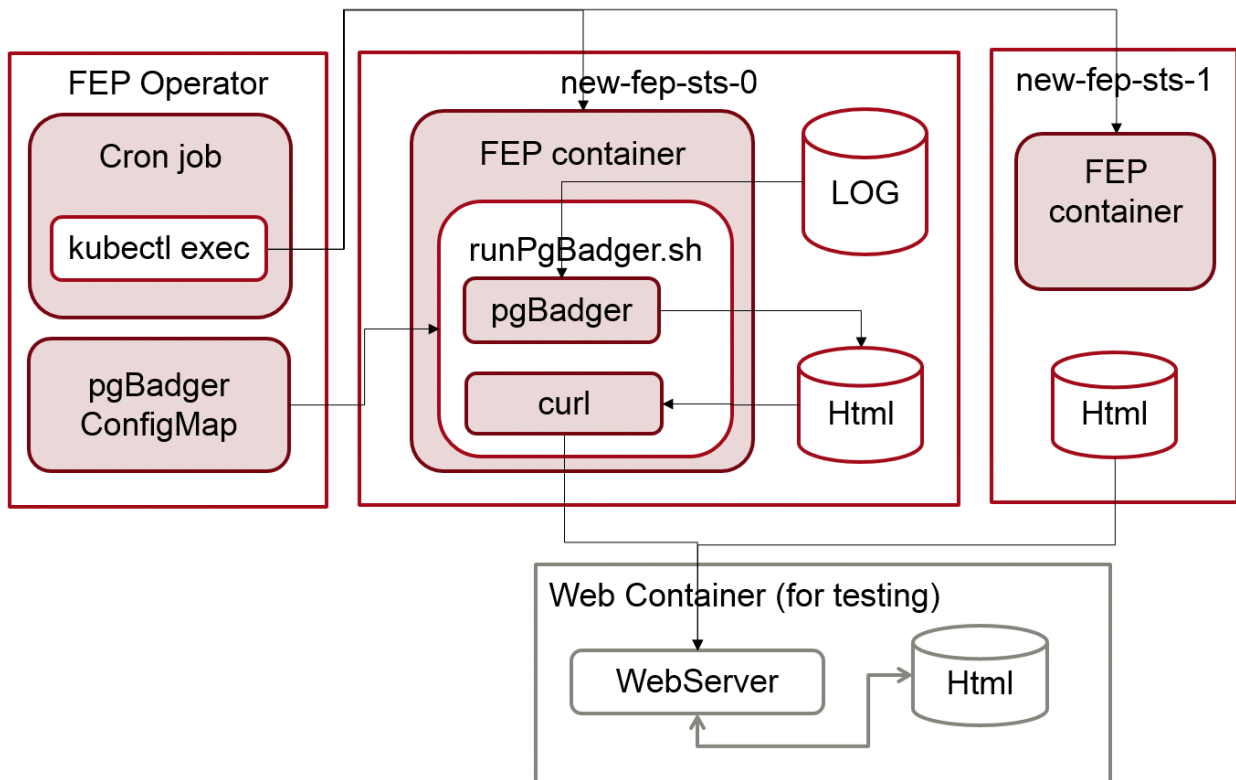
2.10.2 Prometheus and Elasticsearch Log Monitoring

Fluentbit is deployed as a sidecar in the FEPCluster along with patroni and collects postgres database logs. Fluentd is installed with the fluent-plugin-prometheus plugin required for integration with Prometheus and fluent-plugin-elasticsearch plugin required for integration with Elasticsearch.

Fluentd counts the occurrence of different severity levels (PANIC, FATAL, ERROR, WARNING, DEBUG etc) by filtering the incoming log records and the counts are passed onto Prometheus along with the logfile name. The log file name makes it easier to investigate the reason of severity/issue.

If elastic search is enabled and configured, then fluentd sends logs to elastic search which can be viewed in kibana.
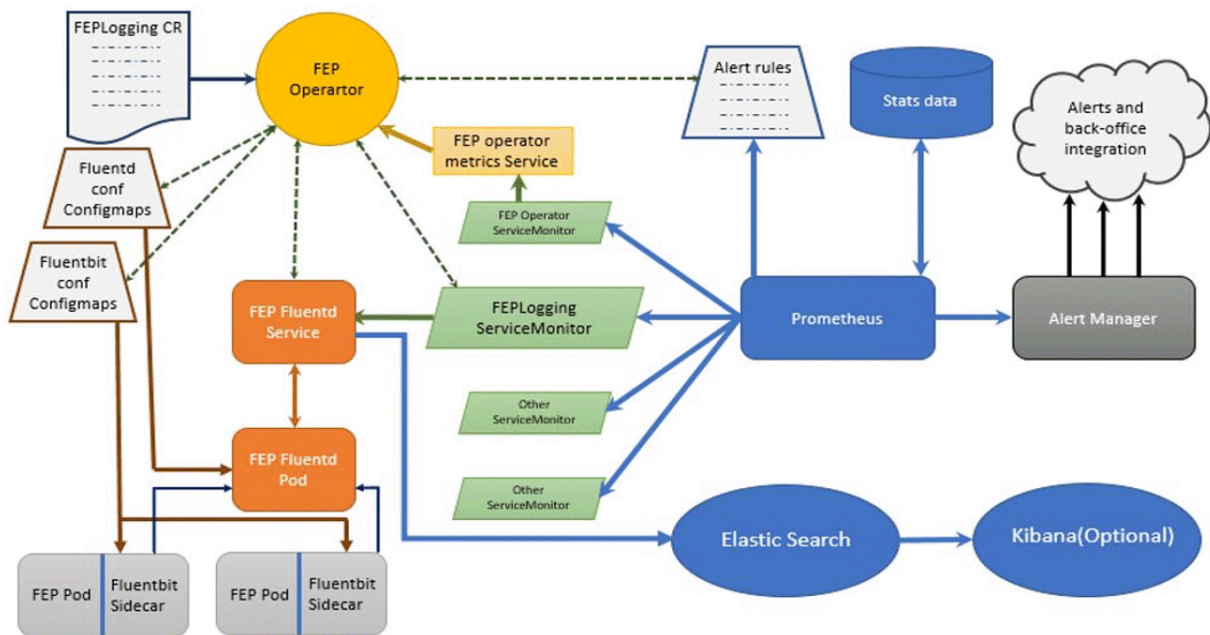
2.11 Scaling Replicas

The scaling feature creates a replica of the reference replica either automatically or manually by the customer. By querying the reference replica service, the customer will be able to direct the query to the automatically added replica instance.

Application

Master Service

Replica Service

Master POD

Replica1 POD

Replica2 POD

Replica3 POD

Automatically
Policy setting

Manual Request

FEP Operator

Scale out automatically based on
resource status or on request
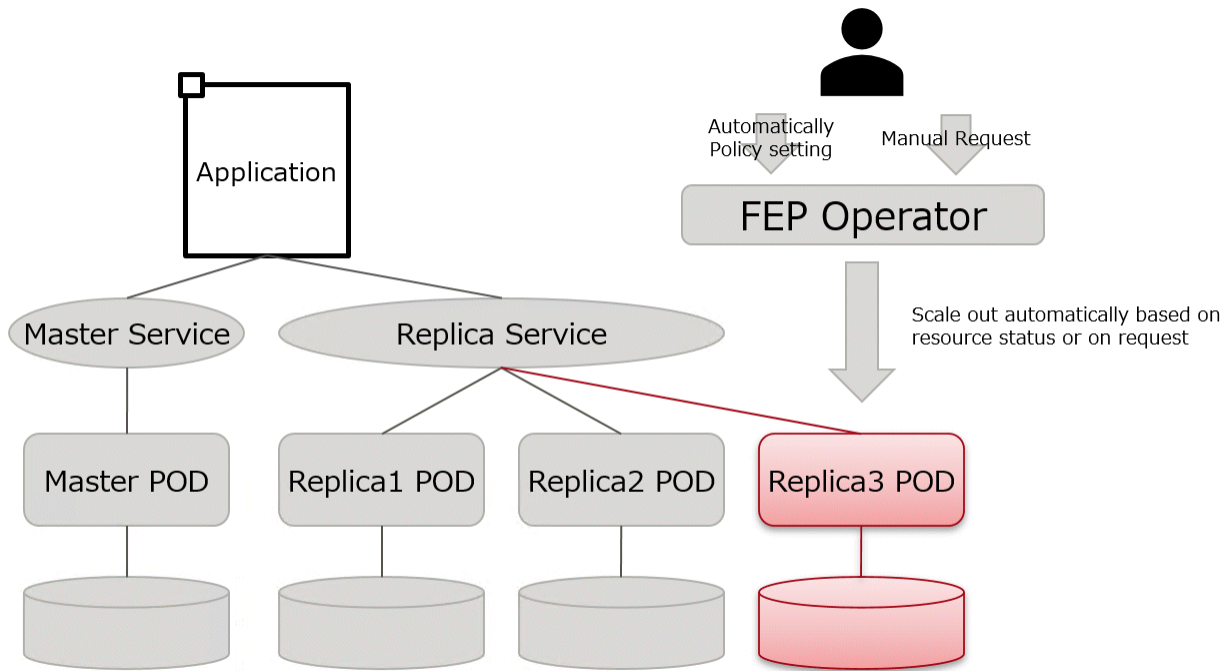
2.11.1 Using the automatic scale out function

The automatic scale out feature works based on the performance metrics (metrics) of objects in a Kubernetes cluster. Metrics are provided by a service that implements the metrics API, called a metrics server.

There are three types of metrics servers (metrics APIs) in OpenShift/Kubernetes.

- Standard metrics server

- Custom metrics server

- External metrics server

Automatic scale out based on CPU utilization works with metrics obtained from standard metrics server.
Automatic scale out based on the number of connections works with metrics obtained from custom metrics server.

The custom metrics server looks at the metrics information for the FEP cluster collected by the monitoring function in Prometheus and publishes it in the form of a metrics API.

Custom metrics server are resources shared by OpenShift/Kubernetes clusters, so they are not built and configured for an extended FEP for Kubernetes installation. To take advantage of automatic scale out based on the number of connections, you must have a custom metrics server.

The custom metrics server must also reference Prometheus, which collects custom metrics for FEP clusters, and must be configured to publish custom metrics for FEP clusters.
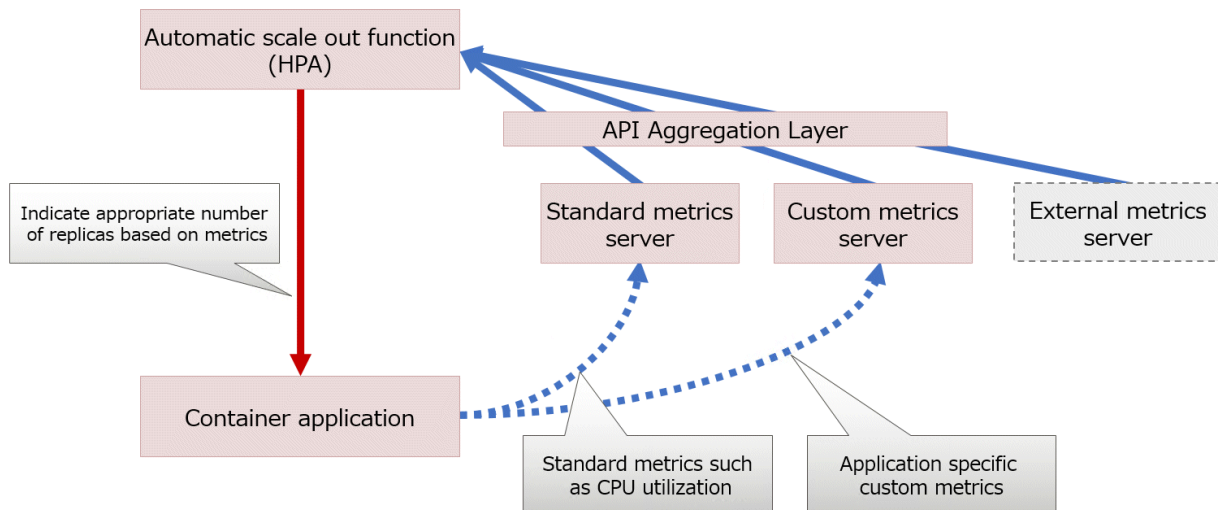
2.12 Disaster Recovery

Retrieve an automated backup of the production environment to object storage.

Restore the FEP cluster from a backup on object storage on the disaster recovery environment OpenShift cluster.
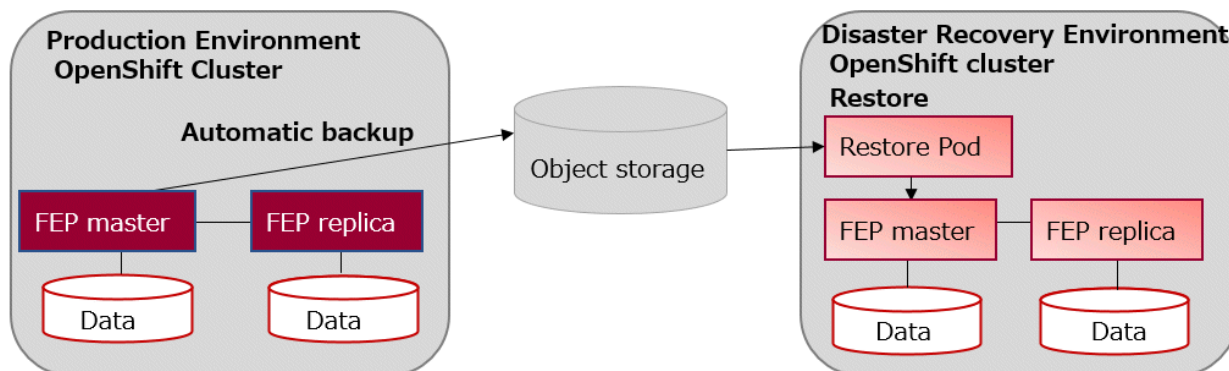
2.13 Ability to Customize the FEP Server Container Image

Fujitsu Enterprise Postgres for Kubernetes bundles a number of OSS modules to extend the PostgreSQL capabilities and is able to handle most demanding workload. There are occasions where a customer would like to include additional modules to further extend PostgreSQL capability. The general idea is to use the FEP Server container image as a base, compile and add the extra modules to the base image to create a customized image hosted on a private container registry.

2.14 Cloud Secret Store

Fujitsu Enterprise Postgres for Kubernetes enhances security of PostgreSQL by providing some unique features. One such feature is the integration with Cloud Secret Store. Customers can opt to store the database secrets, such as database user password or certificates, in an external secret store such as:

- Azure Key Vault

- AWS Secrets Manager

- GCP Secret Manager and

- HashiCorp Vault

Cloud Secret Store leverages the Secret Store CSI Driver, https://secrets-store-csi-driver.sigs.k8s.io/ and respective provider drivers by Azure, AWS, GCP and HashiCorp to integrate with Fujitsu Enterprise Postgres for Kubernetes. With this integration, one can:

- Manage Postgres username/password on Cloud Secret Store

- Manage SSL Certificate on Cloud Secret Store

Benefits of this integration:

- Passwords and certificates are stored in a centralised Cloud Secret Store instead of locally on each Kubernetes cluster

- Allow automatic password and certificate rotation

- Separation of duties; person who maintain the FEP cluster can be a different person who maintain the passwords and certificates in Cloud Secret Store

- Access to secrets in Cloud Secret Store is controlled by authentication and authorization on the Cloud provider

Appendix A OSS Supported by Fujitsu Enterprise Postgres for Kubernetes

The OSS supported by Fujitsu Enterprise Postgres for Kubernetes is listed below.

OSS name	Version and level	Description	Reference
PostgreSQL	17.4	Database management system	PostgreSQL Documentation
orafce	4.13.5	Oracle-compatible SQL features	"Compatibility with Oracle Databases" in the Fujitsu Enterprise Postgres Application Development Guide
Pgpool-II	4.5.5	Failover, connection pooling, load balancing, etc.	"Pgpool-II" in the Fujitsu Enterprise Postgres Installation and Setup Guide for Server
pg_statsinfo	16.0	Collection and accumulation of statistics	"pg_statsinfo" in the Fujitsu Enterprise Postgres Installation and Setup Guide for Server
pg_hint_plan	17.1.7.0	Tuning (statistics management, query tuning)	- "pg_hint_plan" in the Fujitsu Enterprise Postgres Installation and Setup Guide for Server - "Enhanced Query Plan Stability" in the Fujitsu Enterprise Postgres Operation Guide
pg_dbms_stats	14.0		- "pg_dbms_stats" in the Fujitsu Enterprise Postgres Installation and Setup Guide for Server - "Enhanced Query Plan Stability" in the Fujitsu Enterprise Postgres Operation Guide
pg_repack	1.5.1	Table reorganization	"pg_repack" in the Fujitsu Enterprise Postgres Installation and Setup Guide for Server
pg_rman	1.3.16	Backup and restore management	"pg_rman" in the Fujitsu Enterprise Postgres Installation and Setup Guide for Server
pgBadger	12.4	Log analysis	"pgBadger" in the Fujitsu Enterprise Postgres Installation and Setup Guide for Server
pg_bigm	1.2-20240606	Full-text search (multibyte)	"pg_bigm" in the Fujitsu Enterprise Postgres Installation and Setup Guide for Server
ldap2pg	6.2	Mnaging user	- "LDAP Authentication File Settings" in the Fujitsu Enterprise Postgres Installation and Setup Guide for Server - "ldap2pg" in the Fujitsu Enterprise Postgres Installation and Setup Guide for Client
PostgreSQL JDBC driver	42.7.4	JDBC driver	"JDBC Driver" in the Fujitsu Enterprise Postgres Application Development Guide
psqlODBC	17.00.0004	ODBC driver	"ODBC Driver" in the Fujitsu Enterprise Postgres Application Development Guide
psycopg	3.2.4	Python driver	"Python Language Package (psycopg)" in the Fujitsu Enterprise Postgres Application Development Guide
pgvector	0.7.4	Vector data storage, operation, and search (*1)	"pgvector" in the Fujitsu Enterprise Postgres Installation and Setup Guide for Server

OSS name	Version and level	Description	Reference
pgvectorscale	0.5.1	Accelerating vector data search processing	"pgvectorscale" in the Fujitsu Enterprise Postgres Installation and Setup Guide for Server
Apache AGE	1.5.0	Graph data storage and search	"Apache AGE " in the Fujitsu Enterprise Postgres Installation and Setup Guide for Server
pgai	extension-0.8.0	Vectorization of text data using external embedding model and declarative vectorization specification	Fujitsu Enterprise Postgres Knowledge Data Management Feature User's Guide
pg_cron	1.6.5	SQL Job scheduler	"pg_cron" in the Fujitsu Enterprise Postgres Installation and Setup Guide for Server
pgBackRest	2.54.1	Backup and restore management	"Scheduling Backup from Operator" in the User's Guide
patroni	4.0.4	Postgres cluster management	"High Availability" in the User's Guide
postgres_exporter	0.16.0	Postgresql metrics monitoring capabilities for Prometheus with Fujitsu updated queries	"Monitoring" in the User's Guide

*1: pgvector by itself cannot generate vector data. pgai allows you to generate vector data for text data using an external embedded model. To use vector representations in other data formats, use software or services that generate vector data and store the generated vector data in a database.