



215 lines (148 loc) · 10 KB



Crunchy-Postgres-Exporter

Task requirement

To set up Postgres exporter which captures the slow query also.

The **crunchy-postgres-exporter** container provides real time metrics about the PostgreSQL database via an API. These metrics are scraped and stored by a Prometheus time-series database and are then graphed and visualized through the open source data visualizer Grafana.

System Configuration

• RAM: 8 GB

· CPU: 4 cores

Environment details

• Podman - Version 3

List of tools and technologies

- Podman
- Postgres Version 12

Definition of tools

- Podman (the POD manager) is an open source tool for developing, managing, and running containers on your Linux systems. Originally developed by Red Hat® engineers along with the open source community, Podman manages the entire container ecosystem using the libpod library.
- PostgreSQL, also known as Postgres, is a free and open-source relational
 database management system emphasizing extensibility and SQL compliance. It
 was originally named POSTGRES, referring to its origins as a successor to the
 Ingres database developed at the University of California, Berkeley.
- /code

Command for the setup or configuration

1. Create Pod:

podman pod create --name crunchy-postgres --publish 9090:9090 --pul 🖵

- **podman pod create**: This command is used to create a new pod, which is a group of containers that share the same network namespace. Containers within a pod can communicate with each other using the loopback interface.
- --name crunchy-postgres: the pod will be named "crunchy-postgres".
- --publish 9090:9090: This flag maps port 9090 from the host to port 9090 within the pod. Port 9090 is commonly used for services like Prometheus.
- --publish 9187:9187: This flag maps port 9187 from the host to port 9187 within the pod. Port 9187 is the default port for the PostgreSQL Exporter, which exposes PostgreSQL performance metrics.
- --publish 5432:5432: This flag maps port 5432 from the host to port 5432 within the pod. Port 5432 is the default port for PostgreSQL database connections.
- --publish 3000:3000: This flag maps port 3000 from the host to port 3000 within the pod. Port 3000 is commonly used for Grafana.

2. Create Postgres container:

podman run -d --pod crunchy-postgres --name postgres_crunchy -e "P(🖵

- -d: This flag indicates that the container should run in detached mode (in the background).
- **-pod crunchy-postgres**: This flag specifies that the container should be part of the existing "crunchy-postgres" pod.
- --name postgres_crunchy: This flag assigns the name "postgres_crunchy" to the container.
- -e "POSTGRES_DB=postgres": This flag sets the environment variable POSTGRES_DB within the container to "postgres",
- -e "POSTGRES_USER=postgres": This flag sets the environment variable POSTGRES_USER within the container to "postgres", It indicate the username.
- -e "POSTGRES_PASSWORD=redhat": This flag sets the environment variable POSTGRES_PASSWORD within the container to "redhat", which is the password for the PostgreSQL user.
- -v /home/manoj/shiksha_portal/crunchy/postgres/data:/var/lib/postgresql /data: This allows you to persist the PostgreSQL data outside the container, ensuring that the data is retained even if the container is removed.
- docker.io/postgres:12: This specifies the Docker image to use for the container. the container will be based on the "postgres:12" image from Docker Hub.

3. Do Changes in configuration file:

WE will need to modify our postgresql.conf configuration file to tell PostgreSQL to load

shared libraries.

```
echo "shared_preload_libraries = 'pg_stat_statements,auto_explain'' 🗗
```

4. Pull crunchy-postgres-exporter image: Error faced when pulling this image without redhat credentials

manoj@keen:~\$ podman pull registry.connect.redhat.com/crunchydata/c L Trying to pull registry.connect.redhat.com/crunchydata/crunchy-post Error: initializing source docker://registry.connect.redhat.com/cru

NOTE:- Redhat Credentials are required for pulling this image. Steps to resolved 1st login in redhat registry then pull the image

3 of 8

podman login registry.connect.redhat.com -u manoj@fosteringlinux.cc 🖵 Login Succeeded!

podman pull registry.connect.redhat.com/crunchydata/crunchy-postgre Trying to pull registry.connect.redhat.com/crunchydata/crunchy-postGetting image source signatures
Copying blob 07e9ce81867b done
Copying blob 3a0c43549655 done
Writing manifest to image destination
Storing signatures

5. Create a demo container of crunchy for setup.sql.

podman run -itd --pod crunchy-postgres --name crunchy -e EXPORTER_[\Box

6. Get the setup.sql from /opt/cpm/conf/pgxx/setup.sql from the crunchy container according to your postgres version.

podman cp crunchy:/opt/cpm/conf/pg12/setup.sql .

NOTE:- **setup.sql** Creates **ccp_monitoring** role with all necessary grants. Creates all necessary database objects (functions, tables, etc) required for monitoring.

This will copy the **setup.sql** file from the container to the directory where you executed the command.

NOTE:- Dont miss the (.) which is taking place after setup.sql because it represent copy in your current directory.

7. Remove the test container of crunchy:

podman rm -f crunchy

8. Push setup.sql in postgres database:

Get the **setup.sql** from **/opt/cpm/conf/pg12/setup.sql** from the exporter container.

psql -h 127.0.0.1 -U postgres -d template1 < setup.sql

This command is used to execute SQL commands from the setup.sql file within a PostgreSQL database, making changes to the database schema, data, or settings as specified in the SQL file.

9. Create Extension:

```
psql -h 127.0.0.1 -U postgres -d template1 -c "CREATE EXTENSION pg_ 🖵
```

This command is used to enable the **pg_stat_statements** extension in the PostgreSQL database, allowing you to collect statistics about executed SQL statements for performance analysis.

10. Create password for user ccp_monitoring:

```
psql -h 127.0.0.1 -U postgres -d postgres
```

Here We will login in postgres database and then Create password for user ccp monitoring and also will create database name yogendra

```
postgres=# \password ccp_monitoring
Enter new password for user "ccp_monitoring":
Enter it again:
postgres=# create database yogendra;
CREATE DATABASE
```

11. Now create crunchy-postgres-exporter container:

```
podman run -itd --pod crunchy-postgres --name crunchy -e EXPORTER_I 🖵
```

- podman run: This command is used to run a new container.
- -itd: These flags are used together for interactive (console input/output enabled), detached (background) mode.
- --pod crunchy-postgres: This flag specifies that the container should be part of the existing "crunchy-postgres" pod.
- --name crunchy: This flag assigns the name "crunchy" to the container.
- **-e EXPORTER_PG_PASSWORD=redhat**: This flag sets the environment variable **EXPORTER_PG_PASSWORD** within the container to "redhat". This likely represents the password required for PostgreSQL Exporter to connect to the PostgreSQL instance.
- -e EXPORTER PG HOST=127.0.0.1: This flag sets the environment variable

EXPORTER_PG_HOST within the container to "127.0.0.1", indicating the host where the PostgreSQL database is located.

- -e EXPORTER_PG_USER=ccp_monitoring: This flag sets the environment variable EXPORTER_PG_USER within the container to "ccp_monitoring", which is likely the username used by the PostgreSQL Exporter to connect to the PostgreSQL instance.
- -e DATA_SOURCE_NAME=...: This flag sets the DATA_SOURCE_NAME
 environment variable. It specifies the connection details for the PostgreSQL
 Exporter to use when connecting to the PostgreSQL database. The provided
 URL includes the username, password, host, port, database name, and SSL
 mode settings.
- 83a59722eb87: This represents the ID of the container image that you want to run as a container.

12. Check metrics:

```
curl localhost:9187/metrics | grep query
```

```
Q
```

```
Time Current
                                                                      Dload Upload
                                                                                                                         Spent
                                                                                                                                             Left Speed
                                                                                                                      time
 Length of time in seconds of the longest running
  TYPE ccp_connection_stats_max_blocked_query_time gauge
cp_connection_stats_max_blocked_query_time gauge
HELP ccp_connection_stats_max_query_time Length of time in seconds of the longest running
  cp_connection_stats_max_blocked_qu
HELP ccp_connection_stats_max_que
  TYPE ccp_connection_stats_max
cp_connection_stats_max_query_t
                                                               ax_query_time gauge
y_time{server="127.0.0.1:5432"} 0.009827
ccp_connection_stats_max_auery_time(server="127.0.0.1:5432"} 0.00
ccp_connection_stats_max_auery_time(server="127.0.0.1:5432") 0.00
ccp_pg_stat_statements_top_max_exec_time_ms{dbname="postgres",que
="postgres",server="127.0.0.1:5432"} 142.076245
ccp_pg_stat_statements_top_max_exec_time_ms{dbname="template1",que
1377769",role="postgres",server="127.0.0.1:5432"} 9.560248
4682622908088",role="postgres",server="127.0.0.1:5432"} 9.913726
ccp_pg_stat_statements_top_max_exec_time_ms{dbname="template1",que
587586908043",role="postgres",server="127.0.0.1:5432"} 4.53452
ccp_pg_stat_statements_top_max_exec_time_ms{dbname="template1",que
587586908043",role="postgres",server="127.0.0.1:5432"} 4.33452
ccp_pg_stat_statements_top_max_exec_time_ms{dbname="template1",que
14148267286380",role="postgres",server="127.0.0.1:5432"} 3.337925
                                                                                                                                                 create database yogendra",queryid="-638912307742083007",rol"
                                                                                                                                                 ="CREATE EXTENSION pg_stat_statements",queryid="264375628650
                                                                                                                                                 y="CREATE FUNCTION monitor.sequence_status(",q
                                                                                                                                                                                                                                                            id="-182069
                                                                                                                                                 ="CREATE INDEX ON monitor.pg_hba_checksum",
                                                                                                                                                                                                                                                          rid="-6382206
                                                                                                                                                                                                                                                          yid="-120187
                                                                                                                                                 ="CREATE INDEX ON monitor.pg_settings_chec",q
 148267286380",role="postgres",server="127.0.0.1:5432"} 3.3<u>37925</u>
```

- curl is a command-line tool to transfer data to or from a server
- **grep query**: This part of the command uses the **grep** command to search for lines in the input that contain the word "query". This is used to filter the metrics output to only show lines related to queries.

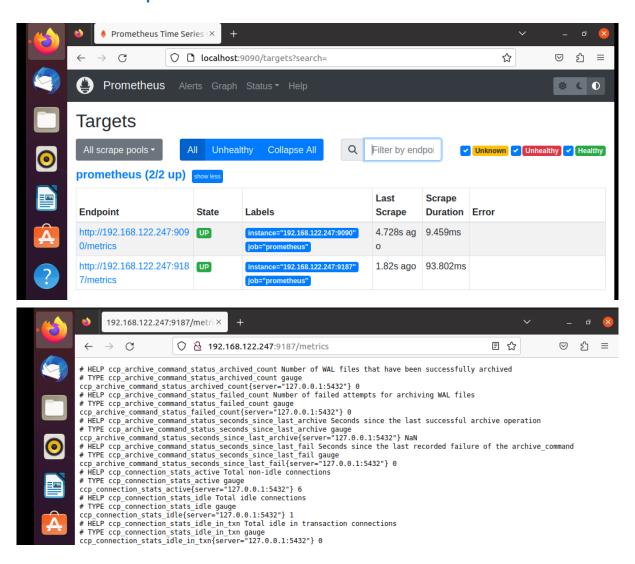
13. Create prometheus container:

```
podman run -itd --pod crunchy-postgres --name prometheus_crunchy -\ 🕌
```

Set the target in prometheus.yml file to get metrics in prometheus

```
manoj@keen:~/shiksha_portal/prometheus_crunchy$ ls
prometheus.yml
manoj@keen:~/shiksha_portal/prometheus_crunchy$ cat prometheus.yml
global:
    scrape_interval: 5s
    external_labels:
        monitor: 'node'
scrape_configs:
    - job_name: 'prometheus'
    static_configs:
        - targets: ['192.168.122.247:9090']
        - targets: ['192.168.122.247:9187']
manoj@keen:~/shiksha_portal/prometheus_crunchy$
```

hit on browser: http://localhost:9090/



14. Create Grafana Container:

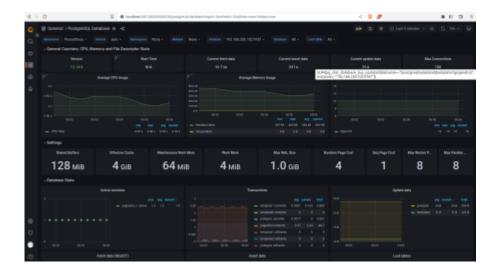
podman run -itd --pod crunchy-postgres --name grafana_crunchy dock 🖳

Hit on browser:

http://localhost:3000/

Select the prometheus as a datasource and import the dashboard.

9628



Test cases list Note: NA

Reference link

https://access.crunchydata.com/documentation/pgmonitor/2.2/exporter/index.html