

Crunchy-Postgres-Exporter

Table of Contents

1. Task requirement.....	2
1. Environment details.....	2
2. List of tools and technologies.....	2
2. Definition of tools.....	2
1. Podman.....	2
2. PostgreSQL.....	2
3. Command for the setup or configuration.....	3
1. Create Pod name crunchy-postgres for all the 4 container:.....	3
2. Create a directory for mounting the persistent storage (volume mount) for the postgres data.....	3
3. check the created pod status.....	3
4. Create Postgres container in this pod which was created with name crunchy-postgres. 4	
5. check the container status.....	5
6. Do Changes in postgresql.conf configuration file: We will need to modify your postgresql.conf configuration file to tell PostgreSQL to load shared libraries.....	5
7. Pull crunchy-postgres-exporter image:.....	6
8. Copy setup.sql from Container to Host:.....	7
9. Remove the Test Container:.....	7
10. Go inside the postgres container & Push setup.sql in postgres database:.....	8
11. Create Extension:.....	8
12. Here We will login in postgres database and then Create password for user ccp_monitoring and also will create database name yogendra.....	9
13. Now create crunchy-postgres-exporter container:.....	9
14. Check metrics:.....	10
15. Create prometheus container:.....	11
16. Configure the prometheus.yml file.....	12
17. Create grafana container for for Visualisation the metrics data.....	14

1.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 visualised through the open source data visualizer Grafana.

1. Environment details

- podman version 3.4.2

2. List of tools and technologies

- Podman
- Postgres Version 12

2.Definition of tools

1. Podman

- **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.

2. PostgreSQL

- **PostgreSQL**, also known as Postgres, is a free and open-source relational database management system emphasising 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.

3. Command for the setup or configuration

1. Create Pod name **crunchy-postgres** for all the 4 container:

```
manoj@keen:~$ podman pod create --name crunchy-postgres --publish 9090:9090 --publish 9187:9187 --publish 5432:5432 --publish 3000:3000
```

- **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 a directory for mounting the persistent storage (volume mount) for the postgres data.

```
manoj@keen:~$ mkdir -p shiksha_portal/crunchy/postgres/data
```

3. check the created pod status

```
manoj@keen:~$ podman pod ps
POD ID      NAME                STATUS      CREATED      INFRA ID      # OF
CONTAINERS
c3bfa7f976f5  crunchy-postgres    Created    8 seconds ago  9898b7a98e9a  1
manoj@keen:~$
```

4. Create Postgres container in this pod which was created with name `crunchy-postgres`

```
manoj@keen:~$ podman run -d --pod crunchy-postgres --name
postgres_crunchy -e "POSTGRES_DB=postgres" -e "POSTGRES_USER=postgres"
-e "POSTGRES_PASSWORD=redhat" -v
/home/manoj/shiksha_portal/crunchy/postgres/data:/var/lib/postgresql/dat
a docker.io/postgres:12
```

podman run: This command is used to run a new container.

- **-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.

5. check the container status

```
manoj@keen:~$ podman ps
```

```
manoj@keen:~$ podman ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS
8898b7a98e9a   k8s.gcr.io/pause:3.5               /pause                  About an hour ago Up 48 minutes ago 0.0.0.0:3000->3000/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:9090->9090/tcp, 0.0.0.0:9187->9187/tcp
c3bfa7f976f5   infra                               /usr/sbin/sshd -D       About an hour ago Up 48 minutes ago 0.0.0.0:3000->3000/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:9090->9090/tcp, 0.0.0.0:9187->9187/tcp
4a8251ac300    docker.io/library/postgres:12      postgres                About an hour ago Up 48 minutes ago 0.0.0.0:3000->3000/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:9090->9090/tcp, 0.0.0.0:9187->9187/tcp
postgres_crunchy
```

6. Do Changes in `postgresql.conf` configuration file:

We will need to modify your `postgresql.conf` configuration file to tell PostgreSQL to load shared libraries.

```
manoj@keen:~/shiksha_portal/crunchy/postgres$ sudo su
root@keen:/home/manoj/shiksha_portal/crunchy/postgres# cd data/
root@keen:/home/manoj/shiksha_portal/crunchy/postgres/data# ls
base      pg_commit_ts  pg_hba.conf   pg_logical    pg_notify
pg_serial  postgresql.conf  postmaster.pid

root@keen:/home/manoj/shiksha_portal/crunchy/postgres/data# echo
"shared_preload_libraries = 'pg_stat_statements,auto_explain'" >>
postgresql.conf
```

We can also do this configuration changes with `vim postgresql.conf` and search for `shared_preload_libraries` and uncomment this also add the below lines

```
# - Shared Library Preloading -

shared_preload_libraries = 'pg_stat_statements,auto_explain'    # (change requires restart)
#local_preload_libraries = ''
#session_preload_libraries = ''
#jit_provider = 'llvmjit'                                         # JIT library to use

# - Other Defaults -
```

7. Pull crunchy-postgres-exporter image:

Before pulling this image

(`registry.connect.redhat.com/crunchydata/crunchy-postgres-exporter:latest`)

You have redhat user id and password otherwise it will show error like this which is given below;

```
manoj@keen:~$ podman pull
```

```
registry.connect.redhat.com/crunchydata/crunchy-postgres-exporter:latest
```

```
Trying to pull registry.connect.redhat.com/crunchydata/crunchy-postgres-exporter:latest...
```

Error: initializing source

docker://registry.connect.redhat.com/crunchydata/crunchy-postgres-exporter:latest: unable to retrieve auth token: invalid username/password: unauthorized: Please login to the Red Hat Registry using your Customer Portal credentials. Further instructions can be found here:

<https://catalog.redhat.com/software/containers/crunchydata/crunchy-postgres-exporter/5f7de9d52937386820422b33?architecture=amd64&image=65319c1b97b29699bd3cf235>

Please try above link to solve this issue

```
manoj@keen:~$ podman login registry.connect.redhat.com -u
manoj@fosteringlinux.com -p 123455
Login Succeeded!
```

```
manoj@keen:~$ podman pull
registry.connect.redhat.com/crunchydata/crunchy-postgres-exporter:latest

Trying to pull
registry.connect.redhat.com/crunchydata/crunchy-postgres-exporter:latest
Getting image source signatures
Copying blob 07e9ce81867b done
Copying blob 3a0c43549655 done
Writing manifest to image destination
Storing signatures
```

Then Run the container with this image

```
manoj@keen:~$ podman run -itd --pod crunchy-postgres --name crunchy -e
EXPORTER_PG_PASSWORD=redhat615904c619c5
registry.connect.redhat.com/crunchydata/crunchy-postgres-exporter
```

Again check container status

8. Copy `setup.sql` from Container to Host:

Copy the `setup.sql` file from the `/opt/cpm/conf/pg12/` directory within the `crunchy` container to your current working directory on the host machine:

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

```
manoj@keen:~$ podman cp crunchy:/opt/cpm/conf/pg12/setup.sql .
```

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

NOTE:- Don't miss the `(.)` which is taking place after `setup.sql` because it represents a copy in your current directory.

9.Remove the Test Container:

Remove the `crunchy` container:

```
manoj@keen:~$ podman rm -f crunchy
```

Here, we will copy `setup.sql` from host to inside the postgres container in `/var/lib/postgresql` directory

```
manoj@keen:~$ podman cp /home/manoj/setup.sql  
postgres_crunchy:/var/lib/postgresql
```

10. Go inside the postgres container & Push setup.sql in postgres database:

```
manoj@keen:~$ podman exec -it postgres_crunchy bash

root@crunchy-postgres:/# ls
bin boot dev  docker-entrypoint-initdb.d  etc  home  lib  lib32
lib64 libx32 media mnt  opt  proc  root  run  sbin  srv  sys  tmp
usr  var

root@crunchy-postgres:/# cd var/lib/postgresql/

root@crunchy-postgres:/var/lib/postgresql# ls
data  setup.sql

root@crunchy-postgres:/var/lib/postgresql# psql -h 127.0.0.1 -U postgres
-d template1 < setup.sql
DO
GRANT ROLE
GRANT ROLE
ALTER ROLE
CREATE SCHEMA
```

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.

11. Create Extension:

```
root@crunchy-postgres:/var/lib/postgresql# psql -h 127.0.0.1 -U postgres
-d template1 -c "CREATE EXTENSION pg_stat_statements;"
CREATE EXTENSION
```

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.

12. Here We will login in postgres database and then Create password for user ccp_monitoring and also will create database name yogendra

```
root@crunchy-postgres:/var/lib/postgresql# psql -h 127.0.0.1 -U postgres
-d postgres
psql (12.16 (Debian 12.16-1.pgdg120+1))
Type "help" for help.
postgres=# \password ccp_monitoring
Enter new password for user "ccp_monitoring":      (I am giving here
redhat)
Enter it again:
postgres=# create database yogendra;
CREATE DATABASE
```

- **psql**: This is the command-line utility for interacting with PostgreSQL databases. It allows you to run SQL queries, manage databases, and perform various database-related tasks.
- **-h 127.0.0.1**: This flag specifies the host where the PostgreSQL database is running.
- **-U postgres**: This flag specifies the username to use when connecting to the database. In this case, you're connecting as the PostgreSQL superuser "postgres."
- **-d postgres**: This flag specifies the name of the database to connect to. In this case, you're connecting to a database named "postgres."

13. Now create crunchy-postgres-exporter container:

```
manoj@keen:~$ podman run -itd --pod crunchy-postgres --name crunchy -e
EXPORTER_PG_PASSWORD=redhat -e EXPORTER_PG_HOST=127.0.0.1 -e
EXPORTER_PG_USER=ccp_monitoring -e
DATA_SOURCE_NAME=postgresql://ccp_monitoring:redhat@127.0.0.1:5432/yogen
dra?sslmode=disable 83a59722eb87
```

- **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.

14. Check metrics:

```
manoj@keen:~$ curl localhost:9187/metrics | grep query
```

- **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.

```
manoj@keen:~$ curl localhost:9187/metrics | grep query
# HELP ccp_connection_stats_max_blocked_query_time Length of time in seconds of the longest running query that has been blocked by a heavyweight lock
# TYPE ccp_connection_stats_max_blocked_query_time gauge
ccp_connection_stats_max_blocked_query_time{server="127.0.0.1:5432"} 0
# HELP ccp_connection_stats_max_query_time Length of time in seconds of the longest running query
# TYPE ccp_connection_stats_max_query_time gauge
ccp_connection_stats_max_query_time{server="127.0.0.1:5432"} 0.009827
ccp_pg_stat_statements_top_max_exec_time_ms{dbname="postgres",query="create database yogendra",queryid="-638912307742083007",role="postgres",server="127.0.0.1:5432"} 142.076245
ccp_pg_stat_statements_top_max_exec_time_ms{dbname="template1",query="CREATE EXTENSION pg_stat_statements",queryid="2643756286501377769",role="postgres",server="127.0.0.1:5432"} 9.560248
ccp_pg_stat_statements_top_max_exec_time_ms{dbname="template1",query="CREATE FUNCTION monitor.sequence_status(",queryid="-1820694682622908088",role="postgres",server="127.0.0.1:5432"} 9.913726
ccp_pg_stat_statements_top_max_exec_time_ms{dbname="template1",query="CREATE INDEX ON monitor.pg_hba_checksum",queryid="-6382206587586908043",role="postgres",server="127.0.0.1:5432"} 4.53452
ccp_pg_stat_statements_top_max_exec_time_ms{dbname="template1",query="CREATE INDEX ON monitor.pg_settings_chec",queryid="-1201874148267286380",role="postgres",server="127.0.0.1:5432"} 3.337925
```

```
# HELP pg_settings_track_activity_query_size_bytes Sets the size reserved for pg_stat_activity.query, in bytes. [Units converted to bytes.]
# TYPE pg_settings_track_activity_query_size_bytes gauge
pg_settings_track_activity_query_size_bytes{server="127.0.0.1:5432"} 1024
# HELP pg_settings_work_mem_bytes Sets the maximum memory to be used for query workspaces. [Units converted to bytes.]
00 134k 0 134k 0 0 899k 0 ---:---:---:---:---:---: 899k
manoj@keen:~$
```

NOW,for prometheus container we have to make prometheus_crunchy directory and prometheus.yml file inside directory

```
manoj@keen:~/shiksha_portal$ mkdir prometheus_crunchy
manoj@keen:~/shiksha_portal$ ls
crunchy prometheus_crunchy
manoj@keen:~/shiksha_portal$ cd prometheus_crunchy/
manoj@keen:~/shiksha_portal/prometheus_crunchy$ touch prometheus.yml
```

15. Create prometheus container:

```
manoj@keen:~$ podman run -itd --pod crunchy-postgres --name
```

```
prometheus_crunchy -v
/home/manoj/shiksha_portal/prometheus_crunchy/prometheus.yml:/etc/promet
heus/prometheus.yml docker.io/prom/prometheus
```

```
Trying to pull docker.io/prom/prometheus:latest...
Getting image source signatures
Copying blob d5c4df21b127 done
Copying blob 2f5f7d8898a1 done
Copying blob ea6cf9f81dfe done
Copying config 3b907f5313 done
Writing manifest to image destination
Storing signatures
5913336df29e649000a51a838d8bf16fd4b4eb68245391205353232e1b147d2b
```

Check all container status

```
manoj@keen:~$ podman ps
CONTAINER ID   IMAGE                                     COMMAND                  NAMES        CREATED        STATUS
PORTS
9898b7a98e9a   k8s.gcr.io/pause:3.5                   /pause                  kube-pause    4 hours ago    Up 4 hours ago
0.0.0.0:3000->3000/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:9090->9090/tcp, 0.0.0.0:9187->9187/tcp c3bfa7f976f5-infra    4 hours ago    Up 4 hours ago
e4a8251ac300   docker.io/library/postgres:12          postgres              postgres      4 hours ago    Up 4 hours ago
0.0.0.0:3000->3000/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:9090->9090/tcp, 0.0.0.0:9187->9187/tcp postgres_crunchy
642990c7daee   registry.connect.redhat.com/crunchydata/crunchy-postgres-exporter:latest /opt/cpm/bin/star...  23 minutes ago  Up 23 minutes ag
o 0.0.0.0:3000->3000/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:9090->9090/tcp, 0.0.0.0:9187->9187/tcp crunchy
5913336df29e   docker.io/prom/prometheus:latest       --config.file=/et...    18 seconds ago  Up 19 seconds ag
o 0.0.0.0:3000->3000/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:9090->9090/tcp, 0.0.0.0:9187->9187/tcp prometheus_crunchy
manoj@keen:~$
```

16. Configure the prometheus.yml file

```
manoj@keen:~/shiksha_portal/prometheus_crunchy$ cat prometheus.yml
```

```
global:
  scrape_interval: 5s
  external_labels:
    monitor: 'postgres'
scrape_configs:
  - job_name: 'prometheus'
    static_configs:
      - targets: ['192.168.122.199:9090']
      - targets: ['192.168.122.199:9187']
```

```

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$

```

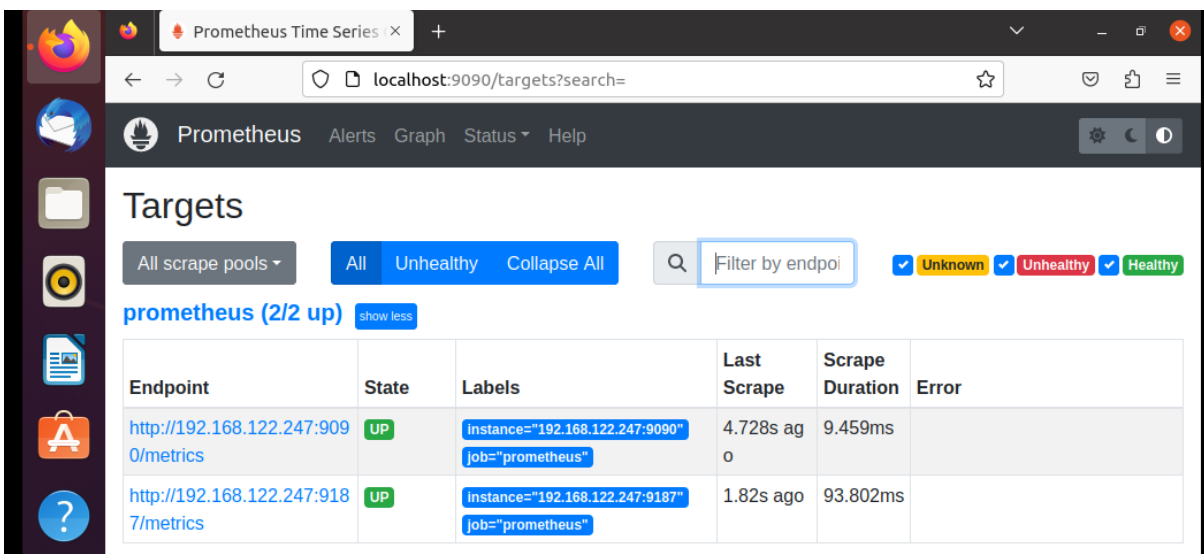
Set the target in prometheus.yml file to get metrics in prometheus and hit on browser(after restart the prometheus container) :

http://localhost:9090/

```

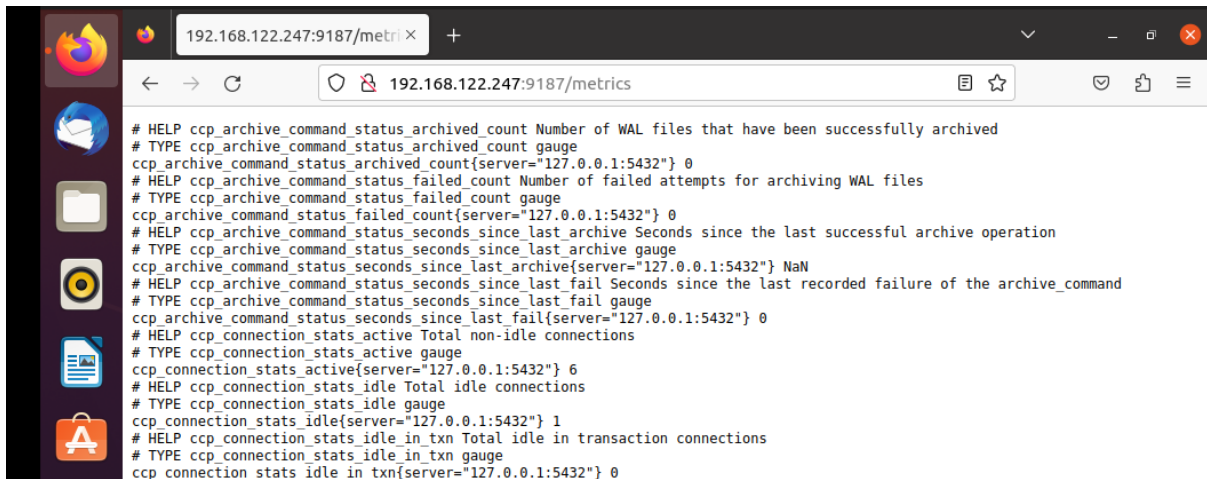
manoj@keen:~/shiksha_portal/prometheus_crunchy$ podman restart
prometheus_crunchy

```



The screenshot shows the Prometheus web interface in a browser window. The address bar shows the URL `localhost:9090/targets?search=`. The page title is "Prometheus Time Series". The main heading is "Targets". Below the heading, there are filters for "All scrape pools", "All", "Unhealthy", and "Collapse All". There is also a search bar labeled "Filter by endpoint" and three status filters: "Unknown", "Unhealthy", and "Healthy". The status filters are all checked. Below the filters, it says "prometheus (2/2 up)" with a "show less" link. A table displays the targets:

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://192.168.122.247:9090/metrics	UP	instance="192.168.122.247:9090" job="prometheus"	4.728s ago	9.459ms	
http://192.168.122.247:9187/metrics	UP	instance="192.168.122.247:9187" job="prometheus"	1.82s ago	93.802ms	



17. Create grafana container for for Visualisation the metrics data

```
manoj@keen:~$ podman run -itd --pod crunchy-postgres --name
grafana_crunchy docker.io/grafana/grafana
Trying to pull docker.io/grafana/grafana:latest...
Getting image source signatures
Copying blob 8a49fdb3b6a5 done
Copying blob 3e825cba1e15 done
Copying blob 8b851b1c9ffe done
Copying blob 4e672be270b9 done
Copying blob 9fe6ebc5ed46 done
Copying blob ab577562f274 done
Copying blob 26e78b642c59 done
Copying blob 3c3e774c3854 done
Copying blob 1a0bceefe858 done
Copying config 6fc2a77217 done
Writing manifest to image destination
Storing signatures
205f113d8ab6c4e3930b950a06eb2f75dd52e9037a46fe51dc3ec6dd63b6b162
```

Now check all container status

CONTAINER ID	IMAGE	NAMES	COMMAND	CREATED	STATUS	PORTS
9898b7a98e9a	k8s.gcr.io/pause:3.5			5 hours ago	Up 4 hours ago	0.0.0.0:3000->3000/tcp, 0.0.0.0:5432-
>5432/tcp, 0.0.0.0:9090->9090/tcp, 0.0.0.0:9187->9187/tcp	e4a8251ac300	docker.io/library/postgres:12	postgres	5 hours ago	Up 4 hours ago	0.0.0.0:3000->3000/tcp, 0.0.0.0:5432-
>5432/tcp, 0.0.0.0:9090->9090/tcp, 0.0.0.0:9187->9187/tcp	642990c7dae	registry.connect.redhat.com/crunchydata/crunchy-postgres-exporter:latest	/opt/cpn/bin/star...	56 minutes ago	Up 56 minutes ago	0.0.0.0:3000->3000/tcp, 0.0.0.0:5432-
>5432/tcp, 0.0.0.0:9090->9090/tcp, 0.0.0.0:9187->9187/tcp	591336df29e	docker.io/prom/prometheus:latest	--config.file=/et...	32 minutes ago	Up 13 minutes ago	0.0.0.0:3000->3000/tcp, 0.0.0.0:5432-
>5432/tcp, 0.0.0.0:9090->9090/tcp, 0.0.0.0:9187->9187/tcp	205f113d8ab6	docker.io/grafana/grafana:latest		5 seconds ago	Up 6 seconds ago	0.0.0.0:3000->3000/tcp, 0.0.0.0:5432-
>5432/tcp, 0.0.0.0:9090->9090/tcp, 0.0.0.0:9187->9187/tcp		grafana_crunchy				

Hit on browser:

<http://localhost:3000/>

Select the prometheus as a datasource and import the dashboard id **9628**

