PostgreSQL GSoC Contributor Proposal

Summary
self link:
https://github.com/xRay2016

Pgexporter is a Prometheus exporter for PostgreSQL. This project aims to add the option to have a custom file specify the queries run against the PostgreSQL instances.

Owner: xray20161@gmail.com
Contributors: Donglin Xie
Created: 2022.04.15

Project Abstract

In this project, we add the option to have a custom query file to specify the queries run against the PostgreSQL instances. The deliverables include the definition and code to integrate the custom files and the Prometheus metric for each queries.

Problem Background

In the pgexporter, the main process call ***start_metrics*** *function, then add EV_READ event on each metrics_fds. The call back function* ***accept_metrics_cb*** *handle the accept event.*

https://github.com/pgexporter/pgexporter/blob/5f48f3880b48b25dbab80452e09cf393e2a95fa9/src/main.c#L742

When the accept event comes, try to accept on the socket and then fork a subprocess to call ***pgexporter_prometheus.***

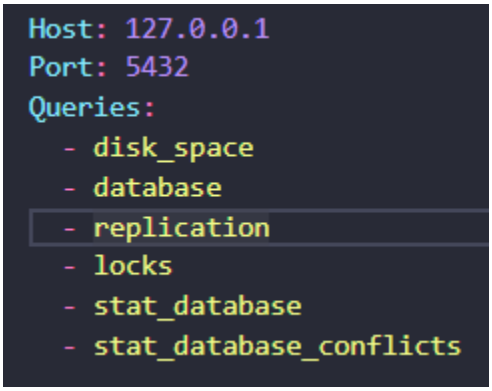
https://github.com/pgexporter/pgexporter/blob/5f48f3880b48b25dbab80452e09cf393e2a95fa9/src/libpgexporter/prometheus.c#L80

In ***pgexporter_prometheus****, the subprocess read http request and reply with the corresponding pages. The metric includes various information, for example version, disk space etc. This project introducing a custom file to specify these queries.*

Design Ideas

In my opinion, the project could be divided into three parts.

1. *the custom file (YAML format) definition and parse*
2. *add custom query part to the function* ***metrics_page()***
3. *add more Prometheus metric*



nihaoFigure 1 The sample custom YAML file

From the sample custom file, it specify the query of PostgreSQL(127.0.0.1:5432). In main function, the custom file is parsed to the specific query rules. Then in metrics_page() function, the Prometheus metric is generated with the specific rules.

Deliverables

1. *Define and integrate loading of the format - maybe YAML - into pgexporter*
2. *Create Prometheus metric for each of the defined queries*
3. *Detailed code documentation*

Schedule of Deliverables (timeline)

May 20 - June 1

- *Design the custom file format(YAML), and learn the Prometheus metric format*

June 1 - July 27 (Phase I)

- *Code for custom file integration*

- *(3 weeks) develop the YAML parse part of the custom file*
- *(3 weeks) develop the custom queries according to the custom file*
- *(1 week) develop the Prometheus metric for some query*
- *(2 week) make unit test, fix bugs and document. reserve one week to prevent accidents*

June 27 – September 11 (Phase I)

- *the reserved weeks I would like to Improve infrastructure on creating new queries and their representation in the Prometheus interface*

About Me

My name is Donglin Xie, a student pursuing a master degree in Computer Science at Zhejiang University, China. I am passionate about programing and open source. I hope I can get the opportunity to contribute to the PostgreSQL community.
I have abundant experience in C/C++, linux and network programing. Here are some of my experience.

1. Tencent Rhino Bird Open Source Training Program-Tars(C++)
(1) Realizes support for the subset function in Tarscpp. It can distribute traffic to different Subsets based on the configured Subset flow rules, achieving more effective development and testing (grayscale testing).
(2) Realize the transparent transmission of TARS_ROUTE_KEY in the Tars structure.
(3) Realize proportional (consistent hashing), select according to request parameters (regular matching) and subset of random rules
github: https://github.com/TarsCloud/TarsCpp
2. High-performance Web server based on C++(C++)
(1) A single Reactor multi-threaded model is realized by using multiple IO multiplexing technology Epoll and thread pool
(2) Use regular matching and state machines to parse HTTP request messages to achieve static resource processing requests
(3) Based on the timer implemented by the heap, close the timeout inactive connection
(4) Realizes the database connection pool of the RAII mechanism, reducing the overhead of establishing and closing database connections
(5) A blocking queue is used to implement an asynchronous log system to record the running status of the server
github: https://github.com/xRay2016/CppWebServer

3. ByteDance Back-end Training Camp: Red Packet Rain Project(Go)
(1) Use Redis to cache Mysql data, realize efficient data reading, and introduce Kafka to write to Mysql asynchronously

(2) The Bloom filter is introduced in the local cache, to reduce the redis network request as much as possible and improve the throughput
(3) The token bucket algorithm is used to realize the current limit of the interface
(4) Use Hystrix-go to deal with the surge in traffic
github: https://github.com/MySuperSoul/teccamp-envelop-rain

Availability Scheduler and Others

I can commit more than 40 hours every week to achieve the goals and deliverables. In weekdays, the working hours are from 6pm to 11pm. The working hours in weekend are from 10am to 9pm. Every weeks, I will make a progress report to the mentor.

The contribution to the open-source project is not limited in GSoC. I would like to continue working on the open-source project. It is a challenging but interesting thing. After the GSoC, do more thing for the PostreSQL community.