MinervaDB

Monitoring PostgreSQL Infrastructure Operations

About MinervaDB

Enterprise-class open source consulting, 24*7 support and remote DBA services provider for MySQL, MariaDB, MyRocks, PostgreSQL and ClickHouse with core expertise in performance, scalability, high availability and database reliability engineering.

What do we monitor in PostgreSQL infrastructure operations ?

- PostgreSQL infrastructure throughput and latency (query performance by response time)
- PostgreSQLTransaction Monitoring:
    - Transaction conflicts and deadlocks
    - Transaction locks
    - Transaction duration
    - Transaction performance
- PostgreSQL configuration / settings
- PostgreSQL connection monitoring
    - Monitoring active connections
    - PostgreSQL connection health check
- Monitoring Postgresql disk I/O operations
- PostgreSQL Monitoring Tools
    - Grafana - Monitoring dashboard to build charts and matrices from time-series data in Prometheus
    - Prometheus DB - Time-series database for PostgreSQL performance monitoring
    - Percona Monitoring Management (PMM) API
    - Query Analytics (QAN) API for storing and querying data collected from PMM client
    - Query Analytics (QAN) web application for monitoring query performance and throughput from a single dashboard

Percona Monitoring Management (PMM) tools for PostgreSQL

Percona Monitoring Management (PMM) tools is client-server monitoring infrastructure built on PMM Server and PMM Client:

PMM Server - PMM Server is the central repository of performance data of PostgreSQL, The data is presented on website infrastructure in the form of tables, dashboards and charts.
PMM Client - PMM Client is installed on PostgreSQL nodes you want to monitor, PMM Client ships performance metrics of both system resource usage and response-time / latency of queries executed in the respective instances.

Installation and configuration of PMM Server

You can install and configure PMM Server following any of the options below:

- PMM Server on Docker
- PMM Server as Virtual Appliance
- PMM Server from Amazon Marketplace

P.S. - We strongly recommend using PMM Server on Docker to reduce operational complexities and efficiency so in this post we have covered PMM Server installation and configuration on Docker.

MinervaDB

Pulling PMM Server Docker image

You pull the latest version from Docker Hub:

```
$ docker pull percona/pmm-server:1
```

Creating PMM Data container

You can create a persistent container for PMM Data running the command below:

```
$ docker create \
   -v /opt/prometheus/data \
   -v /opt/consul-data \
   -v /var/lib/mysql \
   -v /var/lib/grafana \
   --name pmm-data \
   percona/pmm-server:1 /bin/true
```

Creating and launching PMM Server container

To create and launch PMM Server container use docker run command as below:

```
$ docker run -d \
   -p 80:80 \
   --volumes-from pmm-data \
   --name pmm-server \
   --restart always \
   percona/pmm-server:1
```

Installation and configuration of PMM Client

PMM Client is a package of agents and exporters installed on a database host that you want to monitor. We have PMM client distribution available for both Debian and RedHat based Linux distributions:

Debian based GNU/Linux distributions:
Most of the Debian systems come with pre-installed packages of wget, gnupg2 and lsb-release but these packages may be missing from Docker base images. In this case, install them manually before running dpkg:

```
$ sudo apt-get update
$ sudo apt-get install -y wget gnupg2 lsb-release
```

Fetch the repository package:

```
$ wget https://repo.percona.com/apt/percona-release_latest.generic_all.deb
```

Install the downloaded repository package using **dpkg**:

```
$ sudo dpkg -i percona-release_latest.generic_all.deb
```

Install PMM client package:

```
$ apt-get install pmm-client
```

MinervaDB

Installing PMM Client packages on RPM-based Linux distributions

Download repository for RPM-based PMM-Client package:

```
$ sudo yum install https://repo.percona.com/yum/percona-release-latest.noarch.rpm
```

Install PMM Client on RPM-based Linux distributions:

```
yum install pmm-client
```

Configuring PostgreSQL for PMM Monitoring

Enable PostgreSQL configuration parameter track_io_timing to record READ-WRITE transactions statistics data from PostgreSQL infrastructure, You can either do this on a configuration file or by executing the query below on a running production instance:

```
ALTER SYSTEM SET track_io_timing=ON;
SELECT pg_reload_conf()
```

It's strongly recommended to configure PostgreSQL with SUPERUSER privileges to record maximum meaningful performance statistics to troubleshoot efficiently, You can do this by running following command on a standalone PostgreSQL instance:

```
CREATE USER postgres WITH SUPERUSER ENCRYPTED PASSWORD 'postgres';
```

If you are monitoring PostgreSQL on RDS or Amazon Aurora for PostgreSQL:

```
CREATE USER postgres WITH rds_superuser ENCRYPTED PASSWORD 'postgres';
```

Connecting PMM Clients to PMM Server
After successfully installing and configuring PMM Server, We can configure each PMM client to which PMM Server it should send its data to. To connect a PMM client with PMM server you can execute the following commands from PMM client as root user or by using sudo command :

```
$ pmm-admin config --server <IP ADDRESS OF PMM SERVER>:8080
```

Configure PMM Client in PostgreSQL Database Instance to to export performance metadata:

```
sudo pmm-admin add postgresql --user=postgres --password=postgres
```

Conclusion

When you are operating a business critical PostgreSQL infrastructure, The monitoring systems plays a very important role in proactively troubleshooting performance issues. It's technically very expensive to guarantee database infrastructure operations reliability without visibility to PostgreSQL latency and throughput dashboard. We strongly recommend our customers to invest in PostgreSQL performance monitoring systems.