Open Source Database Performance Optimization and Monitoring with PMM

Fernando Laudares, Vinicius Grippa, Michael Coburn
Percona
Fernando Laudares

- Senior Support Engineer
- MySQL & PostgreSQL expert
Vinicius Grippa

- Support Engineer at Percona since 2017
- Working with MySQL for over 5 years
- Started with SQL Server
- Working with databases for 7 years
Michael Coburn

- Product Manager for PMM (as well as for Percona Toolkit)
- At Percona for 6 years across multiple MySQL roles
  - Principal Architect, Managing Consultant, Technical Account Manager
Goals of Today's Tutorial

1. Understand the components of PMM
   - pmm-client - Client tools & agents you install on each server
   - PMM Server
     - Prometheus, Grafana, Query Analytics, Metrics Monitor

2. Install PMM Server at your site
   - Docker (today's method)
   - OVA (Open Virtualization Format) - VMware, VirtualBox, etc
   - Amazon AMI from the AWS Marketplace

3. Review queries using Query Analytics

4. Analyze performance using Metrics Monitor
Agenda

- Fundamentals
- Part 1 - Installation and Configuration
- Part 2 - Query Analytics
- Part 3 - Metrics Monitor
- Questions
Fundamentals
What is PMM

• Free, Open Source database troubleshooting and performance optimization platform for MySQL, MongoDB, and PostgreSQL
  ○ We also support:
    ■ ProxySQL
    ■ Amazon RDS MySQL & Aurora MySQL
    ■ Remote MySQL & PostgreSQL instances

• Runs in your secure environment (this is not a SaaS product!) and on your equipment

• Secured with SSL between client and server
PMM Distribution Methods

1. docker
   - `docker pull percona/pmm-server:latest`

2. Virtual Appliance
   - Supports VMware, RedHat Virtualization, Microsoft Systems Center
   - … and VirtualBox!

3. AWS Marketplace
   - Production-ready AMI running in EC2
AWS Marketplace

- Deploy directly to EC2
- Running CentOS 7

https://aws.amazon.com/marketplace/pp/B077J7FYGX
PMM Architecture

- **pmm-client** (eg. MySQL host)
  - mysqld_exporter - MySQL metrics
  - node_exporter - Linux/OS metrics
  - qan-agent - Query Analytics

- **PMM Server**
  - Query Analytics
    - QAN API & QAN Application
  - Metrics Monitor
    - Prometheus
    - Grafana
PMM Server Components

● Metrics Monitor
  ○ Prometheus
    ■ Timeseries database
    ■ Powerful PromQL query language
  ○ Grafana
    ■ Visualization platform

● Query Analytics
  ○ View query performance in real-time
  ○ Aggregated by **queries consuming most amount of time** in the database
  ○ Query drill-down for individual query performance (MySQL & MongoDB)
    ■ MySQL: Rows read & scanned, Query time & count, InnoDB statistics (w/ Percona Server)
    ■ MongoDB: Query time & count, Docs returned, Response length, Docs scanned
pmm-client Components

- **pmm-admin**
  - Command-line tool for client management
- **node_exporter**
  - Agent that exports Linux metrics
- **mysql_exporter, mongodb_exporter, postgres_exporter, proxysql_exporter**
  - Agents that export server metrics
- **qan-agent**
  - Agent that collects query metrics from MySQL Slow Log or PERFORMANCE_SCHEMA, MongoDB profile collection (system.profile)
Prometheus Data Collection

- Prometheus server asks Consul for which services & instances to query
  - by IP address and port
  - Example: curl https://192.168.56.3:42000/metrics
- Prometheus exporter performs data collection upon curl request
- Exporter generates text exposed via web server at :42002/metrics

```
[root@ps57r ~]# curl -s -k https://10.91.136.33:42002/metrics-hr | grep mysql | head -8
# HELP mysql_exporter_collector_duration_seconds Collector time duration.
# TYPE mysql_exporter_collector_duration_seconds gauge
mysql_exporter_collector_duration_seconds{collector="collect.global_status"} 0.019977679
mysql_exporter_collector_duration_seconds{collector="collect.info_schema.innodb_metrics"} 0.019977679
mysql_exporter_collector_duration_seconds{collector="connection"} 0.006224816
# HELP mysql_exporter_hr_last_scrape_error Whether the last scrape of metrics from MySQL resulted in an error (1 for error, 0 for success).
# TYPE mysql_exporter_hr_last_scrape_error gauge
mysql_exporter_hr_last_scrape_error 0
```
Part 1

Installation and configuration
Environment Notes

- **Authentication**
  - centos / percona
- **ssh percona@<pmm-server>**
  - See handout for your IP address
- **Assumptions**
  - Someone ELSE set up the OS, configured the database, and sends load (i.e. Application exists)
  - Someone else installed dependencies (docker daemon)

- If you get stuck, just grab our attention!!

- **What is deployed**
  - 16 cores, 32GB RAM
  - 9 virtual machines (VirtualBox)
    - 3 x PXC
    - 1 x MySQL
    - 3 x MongoDB
    - 2 x PostgreSQL
Server Configuration - Docker method

● Create docker storage container
  ○ `sudo docker create \
    -v /opt/prometheus/data \
    -v /opt/consul-data \
    -v /var/lib/mysql \
    -v /var/lib/grafana \
    --name pmm-data \
    percona/pmm-server:latest /bin/true`
Server Configuration - Docker method

- **Start docker container**
  - `sudo docker run -d \
    -p 80:80 \
    --volumes-from pmm-data \ 
    --name pmm-server \ 
    --restart always \ 
    percona/pmm-server:latest`

- **Confirm Server is running**
  - `http://<pmm-server>`
Client Configuration

- **Install pmm-client**
  - `yum -y install pmm-client`

- **Connect client to PMM Server**
  - `pmm-admin config --server 10.0.0.13`
Adding MySQL services

- `pmm-admin add mysql:metrics --user root --password percona18live`
- This will set up the following three services:
  - `linux:metrics`
  - `mysql:metrics`
  - `mysql:queries`
Adding MongoDB services

- `pmm-admin add mongodb --uri mongodb://mongoadmin:mongoadmin@localhost:2700/admin --cluster MongoCluster mongo1-2700`
  - `linux:metrics`
  - `mongodb:metrics`
  - `mongodb:queries`
Adding PostgreSQL services

- `pmm-admin add postgresql`
  - `linux:metrics`
  - `postgresql:metrics`
Confirming it all Works

- PMM Server: http://<pmm-server>/
- Prometheus http://<pmm-server>/prometheus

- Do they work? Great - take a break! Stretch your legs
- **No?** Let's Troubleshoot (next slide…)


Troubleshooting PMM

- Check for any red fields:
  - `sudo pmm-admin list`
  - `sudo pmm-admin check-network`
- Restarting one or all components
  - `sudo pmm-admin restart linux:metrics pmm-client`
  - `sudo pmm-admin restart --all`
- Logs are in `/var/log/pmm-*`.log
- Check targets status in Prometheus
  - `http://<pmm-server>/prometheus/targets`
Query Analytics

Examining queries in depth
Query Analytics Dashboard
Query Analytics Overview

- **Query Abstract**
  - Query pattern with placeholders

- **ID**
  - Unique fingerprint, used for query group by

- **Load**
  - Grand Total Time - percentage of time that MySQL server spent executing the query

- **Count**
  - QPS, total count during window, % of total

- **Latency**
  - Min, Med, Avg, P95, Max
### MySQL PERFORMANCE_SCHEMA

**SELECT myisam.sbtest1**

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Rate/Sec</th>
<th>Sum</th>
<th>Per Query Stats</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query Count</td>
<td>40.07 (per sec)</td>
<td>1.73 m 17.33% of total</td>
<td></td>
</tr>
<tr>
<td>Query Time</td>
<td>0.35 load</td>
<td>4:14:06 3.54% of total</td>
<td>8.67 ms avg</td>
</tr>
<tr>
<td>Lock Time</td>
<td>&lt;0.01 (avg load)</td>
<td>0:01:44 0.31% of total</td>
<td>0 avg</td>
</tr>
<tr>
<td>Rows Sent</td>
<td>36.45 (per sec)</td>
<td>1.57 m 0.28% of total</td>
<td>0.00 avg</td>
</tr>
<tr>
<td>Rows Examined</td>
<td>36.45 (per sec)</td>
<td>1.57 m 0.19% of total 1.00 per row sent</td>
<td>0.00 avg</td>
</tr>
</tbody>
</table>
### MySQL Slow Log - *Percona Server only*

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Rate/Sec</th>
<th>Sum</th>
<th>Per Query Stats</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query Count</td>
<td>1.56 k (per sec)</td>
<td>67.51 m 45.59% of total</td>
<td></td>
</tr>
<tr>
<td>Query Time</td>
<td>6.02 load</td>
<td>3 days, 0:14:03 28.26% of total</td>
<td>4.32 ms avg</td>
</tr>
<tr>
<td>Lock Time</td>
<td>0.89 (avg load)</td>
<td>10:43:19 27.38% of total 13.64% of query</td>
<td>588.68 µs avg</td>
</tr>
<tr>
<td>Innodb IO Read Wait</td>
<td>2.98 (avg load)</td>
<td>1 days, 11:47:19 40.65% of total 51.1% of query</td>
<td>2.22 ms avg</td>
</tr>
<tr>
<td>Innodb Read Ops</td>
<td>316.16 (per sec)</td>
<td>13.66 m 33.81% of total</td>
<td>0.00 avg</td>
</tr>
<tr>
<td>Innodb Read Bytes</td>
<td>5.18 MB (per sec)</td>
<td>223.77 GB 33.81% of total 16.38 KB average</td>
<td>3.30 KB avg</td>
</tr>
<tr>
<td>Innodb Distinct Pages</td>
<td></td>
<td></td>
<td>1.59 avg</td>
</tr>
<tr>
<td>Query Cache Hits</td>
<td>0.50 (per sec)</td>
<td>21.53 k 0.54% of total 0.03% QC hit ratio</td>
<td></td>
</tr>
<tr>
<td>Rows Sent</td>
<td>1.56 k (per sec)</td>
<td>67.49 m 3.09% of total</td>
<td>0.17 avg</td>
</tr>
<tr>
<td>Bytes Sent</td>
<td>304.73 KB (per sec)</td>
<td>13.16 GB 4.84% of total 195.06 Bytes average</td>
<td>195.00 Bytes avg</td>
</tr>
<tr>
<td>Rows Examined</td>
<td>1.56 k (per sec)</td>
<td>67.49 m 1.01% of total 1.00 per row average</td>
<td>0.17 avg</td>
</tr>
</tbody>
</table>
**EXPLAIN**

### CLASSIC

<table>
<thead>
<tr>
<th>Id</th>
<th>SelectType</th>
<th>Table</th>
<th>Partitions</th>
<th>CreateTable</th>
<th>CreateTable</th>
<th>PossibleKeys</th>
<th>Key</th>
<th>KeyLen</th>
<th>Ref</th>
<th>Rows</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SIMPLE</td>
<td>g</td>
<td>ALL</td>
<td></td>
<td></td>
<td>geoid, iprange</td>
<td></td>
<td></td>
<td></td>
<td>296670</td>
<td>Using where</td>
</tr>
<tr>
<td>1</td>
<td>SIMPLE</td>
<td>c</td>
<td>eq_ref</td>
<td>PRIMARY</td>
<td>PRIMARY</td>
<td>version_check.g.geoname_id</td>
<td>4</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

### JSON

```
JOIN
+- Bookmark lookup
 | +- Table
 | | table c
 | | possible keys PRIMARY
 | +- Unique index lookup
 | | key c->PRIMARY
 | | possible_keys PRIMARY
 | | key_len 4
 | | ref version_check.g.geoname_id
 | | rows 1
+- Filter with WHERE
 | +- Table scan
 | | rows 296670
 | +- Table
 | | table g
 | | possible_keys geoid,iprange
```
CREATE TABLE, TABLE STATUS, and INDEXES

```
CREATE TABLE `sbttest1` (  
    `id` int(10) unsigned NOT NULL AUTO_INCREMENT,  
    `k` int(10) unsigned NOT NULL DEFAULT '0',  
    `c` char(120) NOT NULL DEFAULT '',  
    `pad` char(60) NOT NULL DEFAULT '',  
    PRIMARY KEY (`id`),  
    KEY `k_1` (`k`)  
) ENGINE=InnoDB AUTO_INCREMENT=100000001 DEFAULT CHARSET=latin1 MAX_ROWS=1000000
```
Server Summary Information

● PMM System Summary Dashboard
● Collects and displays per Server:
  ○ pt-summary
  ○ pt-mysql-summary
  ○ pt-mongodb-summary
● Summary can be downloaded from the UI
Amazon RDS and Aurora

MySQL & PostgreSQL
Add Instances

AWS Credentials

Where do I get the security credentials for my Amazon RDS DB instance?

RDS Instances

<table>
<thead>
<tr>
<th>Name</th>
<th>Region</th>
<th>Endpoint</th>
<th>Engine</th>
<th>Enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>mysql57</td>
<td>eu-west-1</td>
<td>mysql57.ckpwom1.xcon.eu-west-1.rds.amazonaws.com:3306</td>
<td>mysql v5.7.19</td>
<td></td>
</tr>
<tr>
<td>aurora1</td>
<td>us-east-1</td>
<td>aurora1.cdy17lq1j7.us-east-1.rds.amazonaws.com:3306</td>
<td>aurora v5.6.10a</td>
<td></td>
</tr>
<tr>
<td>aurora1-us-east-1c</td>
<td>us-east-1</td>
<td>aurora1-us-east-1c.cdy17lq1j7.us-east-1.rds.amazonaws.com:3306</td>
<td>aurora v5.6.10a</td>
<td></td>
</tr>
<tr>
<td>mysql56</td>
<td>us-east-1</td>
<td>mysql56.cdy17lq1j7.us-east-1.rds.amazonaws.com:3306</td>
<td>mysql v5.7.35</td>
<td></td>
</tr>
<tr>
<td>pmm-doc</td>
<td>us-east-1</td>
<td>pmm-doc.cdy17lq1j7.us-east-1.rds.amazonaws.com:3306</td>
<td>aurora v5.6.10a</td>
<td></td>
</tr>
<tr>
<td>pmm-doc-us-east-1c</td>
<td>us-east-1</td>
<td>pmm-doc-us-east-1c.cdy17lq1j7.us-east-1.rds.amazonaws.com:3306</td>
<td>aurora v5.6.10a</td>
<td></td>
</tr>
</tbody>
</table>
# List Instances

## RDS and remote instances

<table>
<thead>
<tr>
<th>Name</th>
<th>Endpoint</th>
<th>Region</th>
<th>Engine</th>
<th>Remove</th>
</tr>
</thead>
<tbody>
<tr>
<td>rds-aurora1</td>
<td>rds-aurora1.cg8slbmxcsvv.us-east-1.rds.amazonaws.com:3306</td>
<td>us-east-1</td>
<td>aurora 5.6.10a</td>
<td></td>
</tr>
<tr>
<td>rds-mysql56</td>
<td>rds-mysql56.cg8slbmxcsvsv.us-east-1.rds.amazonaws.com:3306</td>
<td>us-east-1</td>
<td>mysql 5.6.37</td>
<td></td>
</tr>
</tbody>
</table>
Remote MySQL and PostgreSQL

For when you don't have shell, or run an unsupported platform (eg. MySQL on Windows)
Add Remote Instances
Part 3 - Using Metrics Monitor

Eye candy
Grafana in a Nutshell

- Open Source data visualisation tool
- Popular datasources
  - Prometheus
  - CloudWatch
  - Graphite
  - Elasticsearch
- Templated Variables
  - Define your graph metrics, and let the hosts get filled in automatically
  - GREAT for large, dynamic environments where hosts are considered ephemeral
Prometheus revisited

- Timeseries database - metric name + key/value pairs
  - `mysql_global_variables_innodb_buffer_pool_instances{instance="ps57",job="mysql"} = 8`
  - `mysql_slave_status_slave_io_running{instance="ps57r",job="mysql",master_host="10.91.136.32",master_uuid="9809315d-4d97-11e6-b85e-0007cb03dc86"} = 1`

- Flexible query language - PromQL
- Collection of metrics based on HTTP pull
- Targets identified via service discovery or static configuration files
  - We're using consul in PMM for service discovery
How can I... in general

- Compare servers to each other
  - Cross Server graphs
- Show behaviour now() vs past periods (1 day ago, 1 week ago)
  - Trends Overview dashboard
- Describe Linux and hardware usage
  - System Overview, Network Overview, Disk Performance, CPU Utilization Details
How can I... for MySQL

- At a glance MySQL + Storage Engine
  - MySQL Overview, InnoDB, InnoDB Advanced, MyRocks
- Review High Availability metrics
  - PXC Cluster Overview, ProxySQL Overview, MySQL Replication
- Table statistics*, User statistics*, Query Response Time*
  - Largest tables by rows and size, total DB size, tables by rows read and changed, auto_increment usage (about to hit the limit?)
  - Top users by connection count, network usage, rows read/changed

* Percona Server only
How can I… for MongoDB

● Review server-at-a-time metrics
  ○ MongoDB Overview
● Understand ReplSet health
  ○ MongoDB ReplSet
● Explore Cluster health
  ○ MongoDB Cluster Summary
● Examine per engine activity
  ○ MongoDB WiredTiger
  ○ MongoDB RocksDB
  ○ MongoDB MMAPv1
  ○ MongoDB InMemory
How can I... for PostgreSQL

- Understand PostgreSQL performance
  - PostgreSQL Overview
- More dashboards to come! What else would you like to see?
Annotations

- Visualize Application Events in PMM
  - `pmm-admin annotate "Application deployment v1.3"`
Alerting

- Alerting
  - Cannot use Templated Variables
  - Instead, replace with string constants for instance name
Other features

- **Textfile collector**
  - Ever wanted to create a metric series using bash?

- **MySQL & PostgreSQL Custom Queries**
  - Use SELECTs to create metric series you can then plot
  - Ideas: Application specific metrics, or add InnoDB Group Replication support before Percona does!
Almost the end

Parting thoughts
Advice

- Metrics retention is 30 days
  - We are looking at options to present a longer history
- mysql:metrics are polled at 1s, 5s, and 60s resolutions, and linux:metrics is every 1s
  - On high-latency links you might need to tune METRICS_RESOLUTION upwards
- Don't skimp on resources
  - Prometheus in particular needs a lot of CPU cores and fast disks
- Consider disabling some *_exporter features to minimise performance impact
  - --disable-tablestats, --disable-processlist
- Keep queries in the database (PII, security)
  - --disable-queryexamples
The Future of PMM

- **Query Analytics**
  - aggregation across all servers, new filtering and sorting options, faster performance
  - PostgreSQL support coming in first half 2019
- **Alerting**
  - Integration with Prometheus AlertManager
- **Long term metrics storage**

- **What would you like to see in PMM?**
Questions?

- Michael Coburn michael.coburn@percona.com

- Percona is looking for MongoDB, MySQL, and PostgreSQL rockstars! Be sure to stop by Percona’s booth.

- Do you have any areas or benchmarks you want Percona to talk about in blogs together? Any features or tools you think we should focus on?
Rate My Session

Introducing gh-ost: triggerless, painless, trusted online schema migrations
11:20 - 12:10, Matterhorn 2

Tap the session

Rate & Review

Tap to rate & review

Feedback (optional)

Submit
Thank You Sponsors!!