Open Source Database Performance Optimization and Monitoring with PMM

Vinnie Grippa, Charly Batista, Michael Coburn
Percona
Vinicius Grippa

- Support Engineer at Percona since 2017
- Working with MySQL for over six years
- Working with databases for over nine years
- Speaker at PL 2018 and meetups about MySQL/MongoDB
Charly Batista

- Senior Support Engineer at Percona
- MySQL and PostgreSQL expert
- Working with databases for over 15 years
- Speaker at Percona Live and meetups about Database and Development
Michael Coburn

- Product Manager for PMM (and Percona Toolkit)
- At Percona for six 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 and 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
5. Deploy PMM 2 (Optional)
Tutorial Requirements

• Laptop
• ssh client
• web browser
Agenda

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

- A free, Open Source database troubleshooting and performance optimization platform for MySQL, MongoDB, and PostgreSQL
  - We also support:
    - ProxySQL
    - Amazon RDS MySQL and Aurora MySQL
    - Remote MySQL and 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:1`

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

Search for "pmm" or "Percona Monitoring and Management"

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

- pmm-client (eg. MySQL host)
  - mysqlld_exporter - MySQL metrics
  - node_exporter - Linux/OS metrics
  - qan-agent - Query Analytics
- PMM Server
  - Query Analytics
    - QAN API and 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 and MongoDB)
    ■ MySQL: Rows read and scanned, Query time and count, InnoDB statistics (w/ Percona Server)
    ■ MongoDB: Query time and 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

- **mysqld_exporter, mongodbExporter, postgresExporter, proxysqlExporter**
  - 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 and 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"} 0.006224816
mysql_exporter_collector_duration_seconds{collector="connection"} 2.1584e-05
# 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 One

Installation and configuration
Environment Notes

- ssh root@<pmm-server>
  - Select an IP from the document list for your instance

- Assumptions
  - Someone ELSE set up the OS, configured the database, and sends load (i.e. Application exists)
  - Someone else installed dependencies (docker daemon)

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

If you get stuck, just grab our attention!
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 --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 --host=localhost --user pmm --password 'pmm'
  - `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

System Activity

CPU Utilization (%)
- Network Activity: min 1.48 MB/s, max 6.59 MB/s, avg 2.65 MB/s
- Disk I/O Activity: min 1.32 MB/s, max 9.00 MB/s, avg 2.40 MB/s
- CPU Utilization: min 9.46%, max 26.06%, avg 14.48%

Database Server Activity

Threads
- Queries (right-y): min 203.92 ops, max 521.97 ops, avg 293.41 ops
- Peak Threads Connected: min 15.00, max 39.00, avg 21.03
- Peak Threads Running: min 8.00, max 22.00, avg 13.32

PMM Query Analytics

Top 10 of 662 Queries by % Grand Total Time (%GT)

<table>
<thead>
<tr>
<th>#</th>
<th>Query Abstract</th>
<th>Load</th>
<th>Count</th>
<th>Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SELECT geoip countries</td>
<td>0.66</td>
<td>2.73 QPS</td>
<td>117.72k</td>
</tr>
<tr>
<td>2</td>
<td>INSERT clientsoftwareversion</td>
<td>0.05</td>
<td>12.81 QPS</td>
<td>553.21k</td>
</tr>
<tr>
<td>3</td>
<td>SELECT wp_posts</td>
<td>0.05</td>
<td>4.61 QPS</td>
<td>199.20k</td>
</tr>
<tr>
<td>4</td>
<td>SELECT wp_posts wp_term_relationships wp_t...</td>
<td>0.05</td>
<td>0.35 QPS</td>
<td>15.11k</td>
</tr>
</tbody>
</table>
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
### SELECT myisam.sblast1

<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>4.32 ms avg</td>
</tr>
<tr>
<td>Query Time</td>
<td>6.02 load</td>
<td>3 days, 0:14:03 28.26% of total</td>
<td>432 ms avg</td>
</tr>
<tr>
<td>Lock Time</td>
<td>0.89 (avg load)</td>
<td>10:43:19 23.38% of total 13.64% of qu...</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...</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 a...</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 ...</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 ...</td>
<td>0.17 avg</td>
</tr>
</tbody>
</table>
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

INDEXES

<table>
<thead>
<tr>
<th>KeyName</th>
<th>Type</th>
<th>Unique</th>
<th>Packed</th>
<th>Column</th>
<th>Cardinality</th>
<th>Collation</th>
<th>Null</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRIMARY</td>
<td>BTREE</td>
<td>Yes</td>
<td>No</td>
<td>id</td>
<td>98629992</td>
<td>A</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>k_1</td>
<td>BTREE</td>
<td>No</td>
<td>No</td>
<td>k</td>
<td>28764292</td>
<td>A</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>
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 and PostgreSQL
Add Instances
# 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.cg8slbmxcsve.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-mysq156</td>
<td>rds-mysq156.cg8slbmxcsve.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

Add remote MySQL Instance

- Hostname
- Name (default: Hostname)
- (default: 3306)
- Username
- Password

Add remote PostgreSQL Instance

- Hostname
- Name (default: Hostname)
- Port (default: 5432)
- Username
- Password

Add instance
Part Three - Using Metrics Monitor
Grafana in a Nutshell

- Open Source data visualisation tool
- Popular data sources
  - 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="ps 57",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?

- Explore Cluster health
  - MongoDB Cluster Summary
- Understand ReplSet health
  - MongoDB ReplSet
- Review server-at-a-time metrics
  - MongoDB Overview
- 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

- 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 and 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
Almost the End... of PMM 1
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`
PMM 2 Beta

- Do you want to deploy PMM 2? (optional)
- We'll upgrade PMM Server, then clients, then add the services back
- Note: This is not an in-place upgrade - this is a "start from scratch" deployment
Server Configuration - Docker

- Identify and stop existing PMM Server
  - `docker ps -a`
  - `docker stop pmm-server`

- Create docker storage container
  - `docker create -v /srv --name pmm-data-2 perconalab/pmm-server:2.0.0-beta1 /bin/true`

- Run the container

- Confirm server is running
  - `http://<pmm-server>`
pmm2-client Installation

- Stop and remove pmm-client
  - `pmm-admin stop --all`
  - `yum -y erase pmm-client`

- Configure for Percona Experimental Repository
  - `percona-release disable all`
  - `percona-release enable original experimental`

- Install client
  - `yum -y install pmm2-client`
Adding Client Services

- Advertise the client to PMM Server
  - `pmm-admin config --server-insecure-tls --server-address=<IP Address>:443`

- Add services
  - `pmm-admin add mysql --use-slowlog --username=pmm --password=percona18live`
  - `pmm-admin add mongodb --use-profiler --use-exporter --username=mongoadmin --password=mongoadmin`
  - `pmm-admin add postgresql --username=pmm --password=pmm`
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

What would you like to see in PMM?
Any 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? Please let us know!
Thank You to Our Sponsors
Rate My Session

Introducing gh-ost: triggerless, painless, trusted online schema migrations

Rate & Review

Tap to rate and review.