Performance Analysis and Troubleshooting Methodologies for Databases

Peter Zaitsev, CEO Percona

July 9th, 2019

Montpellier Big Data and Data Science Meetup

PERCONA

Thank you

elium

PERCONA

Enjoying South of France

Lets Get to Kow you!

How Many of you are…

PERCONA

Databases and Performance

Databases are frequent Performance Trouble Makers

Why Databases are Painful ?

Generally Non-Linear Scalability

Complex

Often Poorly understood by developers

Performance Work with Databases

Troubleshooting

Capacity Planning

Cost and Efficiency Optimization

Change Management

Points of View

BlackBox – "Application Developer"

WhiteBox – "DBA, Ops"

Developer Point of View

Database as a Blackbox

I throw queries at it and it responds

DBaaS bring this “promise” to OPS too

BlackBox Success Criteria for Databases

Availability

Response Time

Correctness

Cost

Ops Point of View

Load

Resource Utilization

System/Hardware Problems

Scaling/Capacity Planing

PERCONA

Methodologies for Performance Troubleshooting and Analyses

Typical Default

Troubleshooting by Random Googling

PERCONA

Problems with Typical Approach

Hard to Assure Outcome

Hard to Train People

Hard to Automate

PERCONA

Methodologies Save the Day

USE (Utilization, Saturation, Errors) Method by Brendan Gregg

RED (Rate, Errors(Rate), Duration) Method Tom Wilkie

Golden Signals (Latency - Traffic - Errors - Saturations) Method by Rob Ewaschuk

USE Method

USE Method Basics

Developed to Troubleshoot Server Performance Issues

Resolve 80% of problems with 5% of Effort

Operating System Specific Checklists Available

USE Method in One Sentence

"For every resource, check utilization, saturation, and errors."

USE Method Terminology Defitinions

Resource	• all physical server functional components (CPUs, disks, busses, ...)
Utilization	• the average time that the resource was busy servicing work
Saturation	• the degree to which the resource has extra work which it can't service, often queued
Errors	• the count of error events

USE Method Resources

CPUs: sockets, cores, hardware threads (virtual CPUs)

Memory: capacity

Network interfaces

Storage devices: I/O, capacity

Controllers: storage, network cards

Interconnects: CPUs, memory, I/O

USE Method with Software

Same Basic Resources Apply

Additional Software Resources Apply

Mutex Locks

File Descriptors

Connections

USE Method Benefits

Proven Track Record

Broad Applicability

Detailed Checklists Available

USE Method Drawbacks

Requires Good Understanding of System Architecture

Requires Access to Low Level Resources Monitoring

Hard to apply in Service “Blackbox” environments

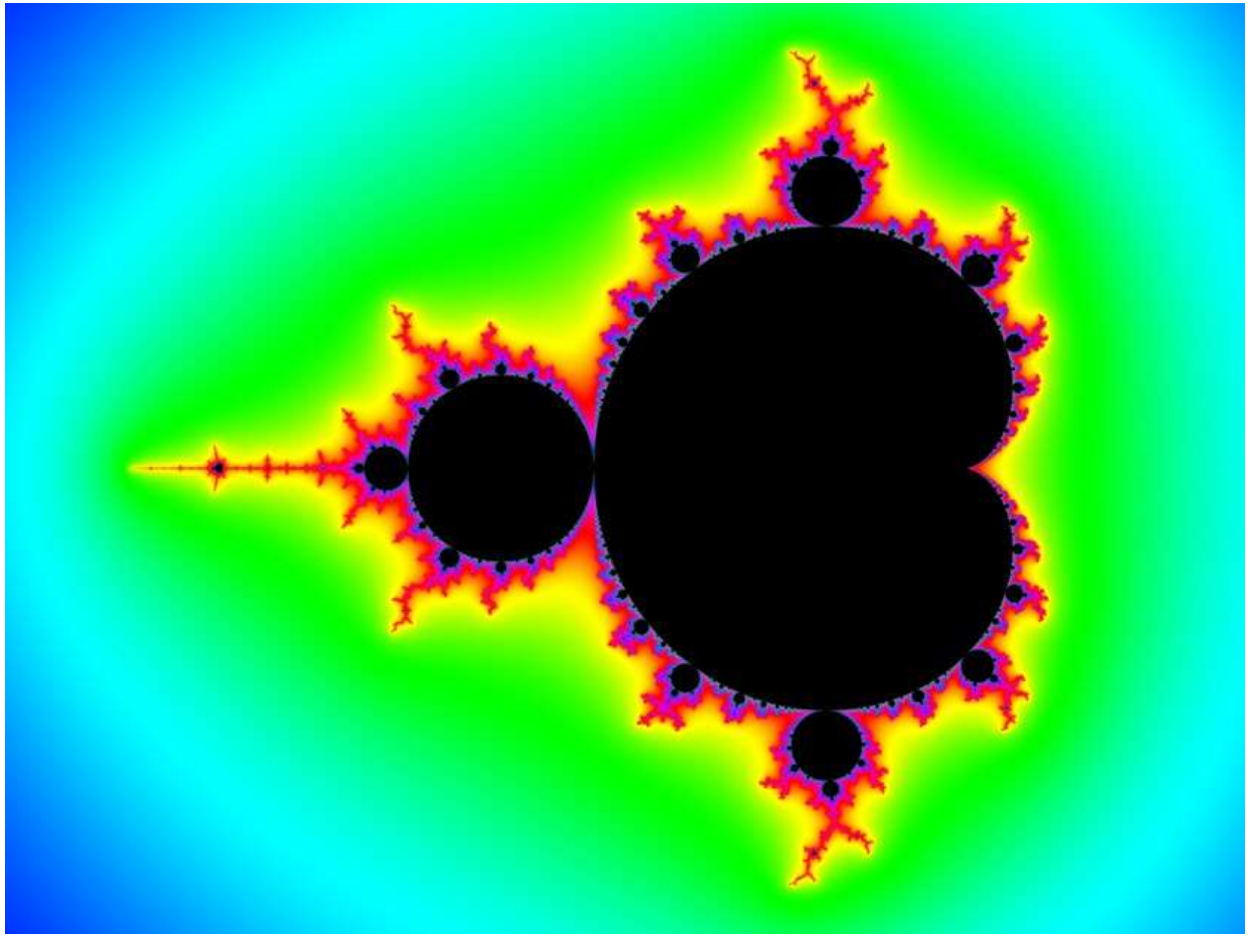
Cloud Computing

Limits in place to isolate Tenants

Dynamic Resource Management

True “Hardware” properties on Hypervisor level only

Understanding Queueing



Many Different Levels

Process running on CPU	• Actually waiting on the Memory
Process Running from User Standpoint	• May be put to queue due to CPU saturation
Disk request issued to EBS	• Can be waiting on network

USE Method for Linux

component	type	metric
CPU	utilization	system-wide: `vmstat 1`, "us" + "sy" + "st"; `sar -u`, sum fields except "%idle" and "%iowait"; `dstat -c`, sum fields except "idl" and "wai"; per-cpu: `mpstat -P ALL 1`, sum fields except "%idle" and "%iowait"; `sar -P ALL`, same as `mpstat`; per-process: `top`, "%CPU"; `htop`, "CPU%"; `ps -o pcpu`; `pidstat 1`, "%CPU"; per-kernel-thread: `top`/`htop` ("K" to toggle), where VIRT == 0 (heuristic). [1]
CPU	saturation	system-wide: `vmstat 1`, "r" > CPU count [2]; `sar -q`, "runq-sz" > CPU count; `dstat -p`, "run" > CPU count; per-process: /proc/PID/schedstat 2nd field (sched_info.run_delay); `perf sched latency` (shows "Average" and "Maximum" delay per-schedule); dynamic tracing, eg, SystemTap schedtimes.stp "queued(us)" [3]
CPU	errors	`perf` (LPE) if processor specific error events (CPC) are available; eg, AMD64's "04Ah Single-bit ECC Errors Recorded by Scrubber" [4]
Memory capacity	utilization	system-wide: `free -m`, "Mem:" (main memory), "Swap:" (virtual memory); `vmstat 1`, "free" (main memory), "swap" (virtual memory); `sar -r`, "%memused"; `dstat -m`, "free"; `slabtop -s c` for kmem slab usage; per-process: `top`/`htop`, "RES" (resident main memory), "VIRT" (virtual memory), "Mem" for system-wide summary
Memory capacity	saturation	system-wide: `vmstat 1`, "si"/"so" (swapping); `sar -B`, "pgscank" + "pgscand" (scanning); `sar -W`; per-process: 10th field (min_flt) from /proc/PID/stat for minor-fault rate, or dynamic tracing [5]; OOM killer: `dmesg | grep killed`
Memory capacity	errors	`dmesg` for physical failures; dynamic tracing, eg, SystemTap uprobes for failed malloc()s

http://www.brendangregg.com/USEmethod/use-linux.html

Percona Monitoring and Management

100% Free and Open Source

Purpose Build for Open Source Database Monitoring

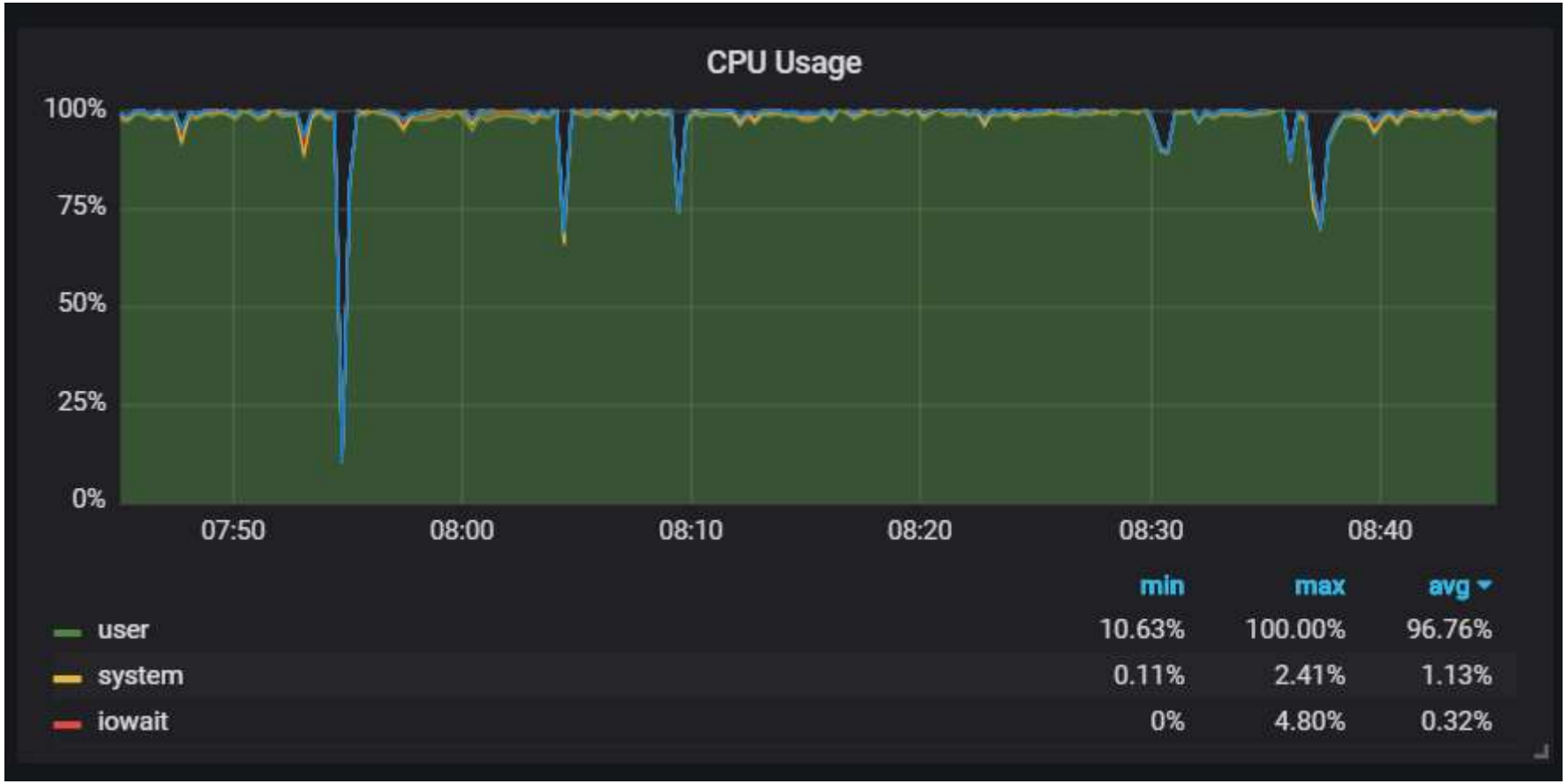
Based on leading Open Source Technologies – Grafana, Prometheus

Easy to Set up

PERCONA

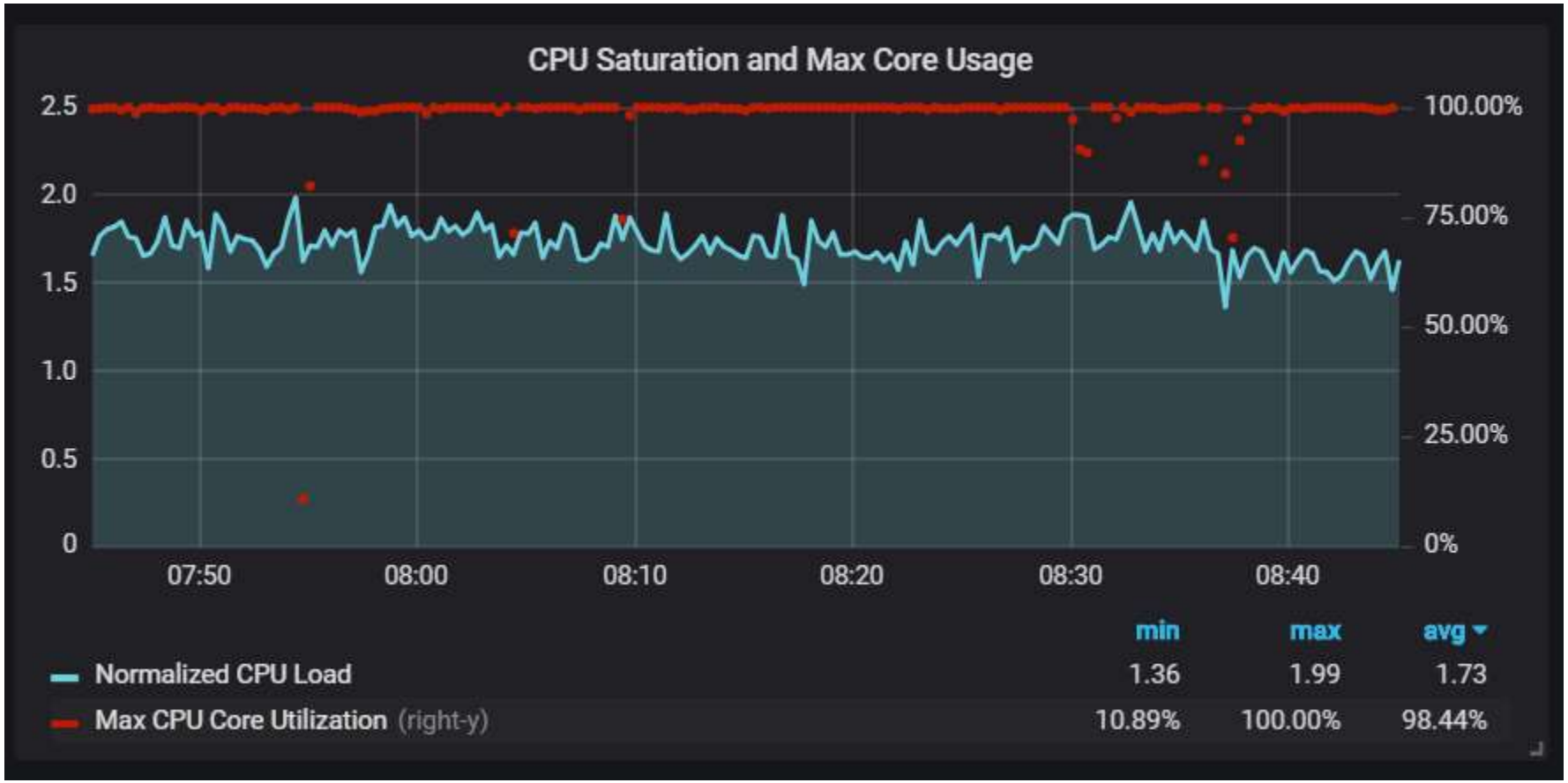
Monitoring and Management

CPU Utilization



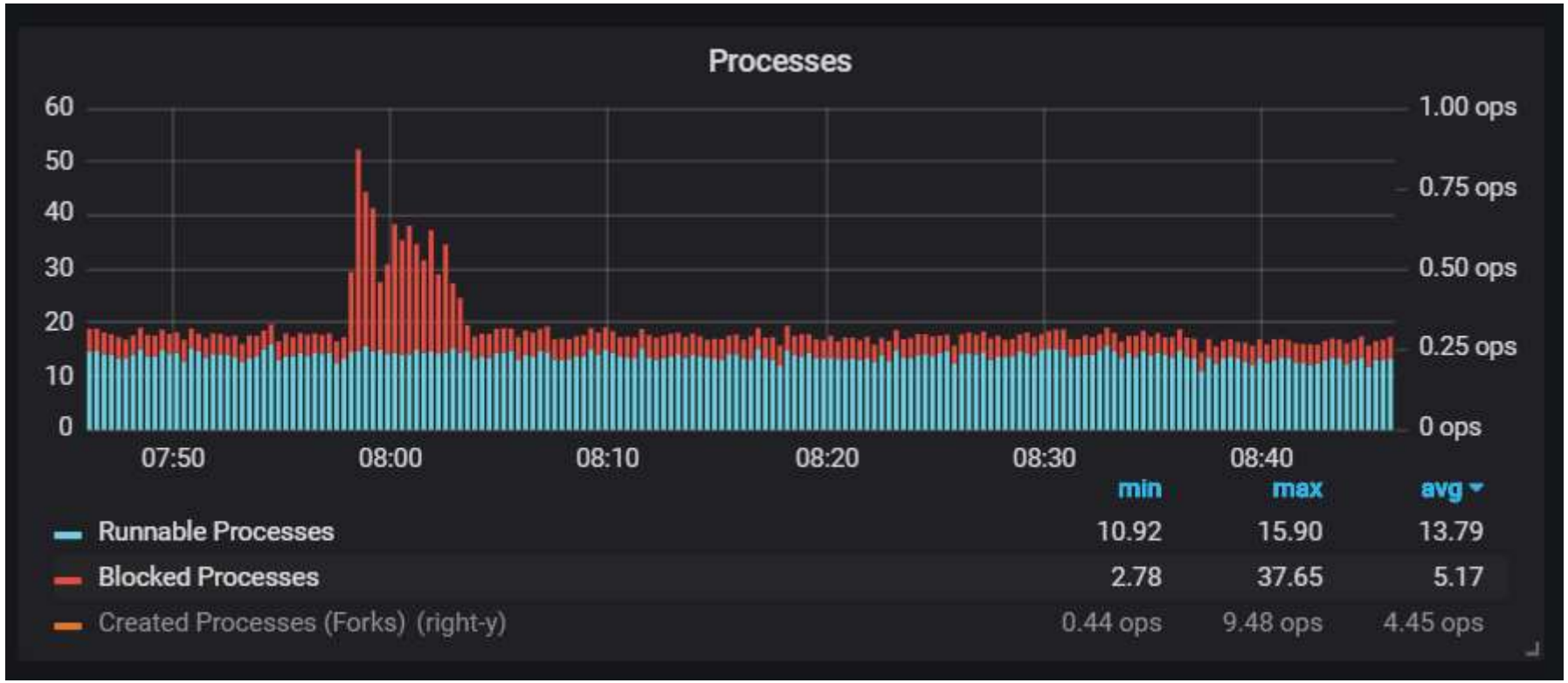
PERCONA

CPU Saturation



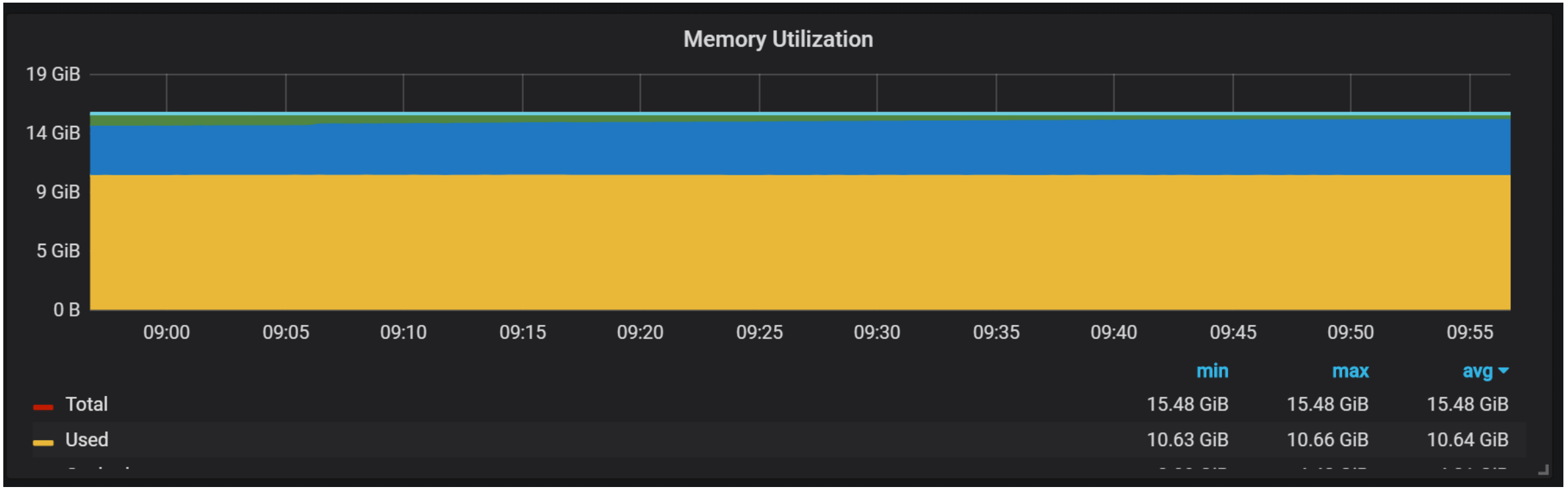
PERCONA

Process View



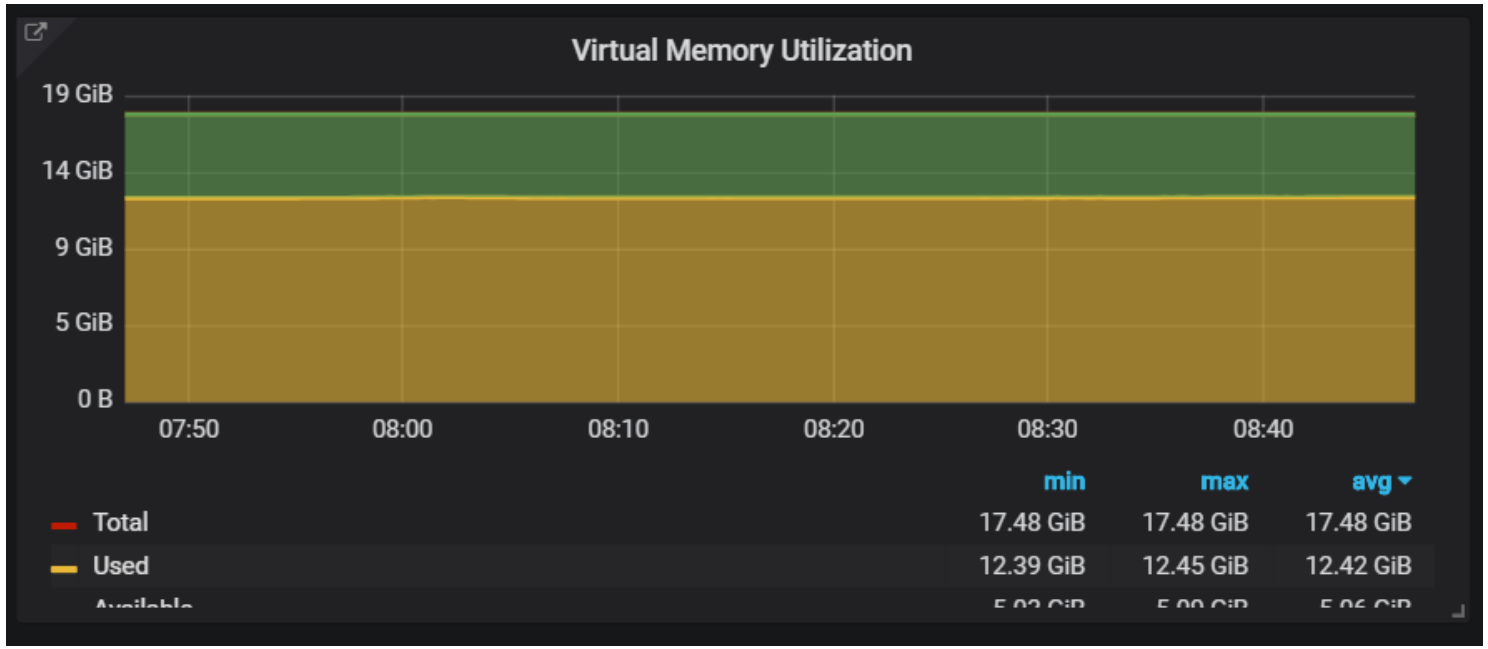
PERCONA

Physical Memory



PERCONA

Virtual Memory

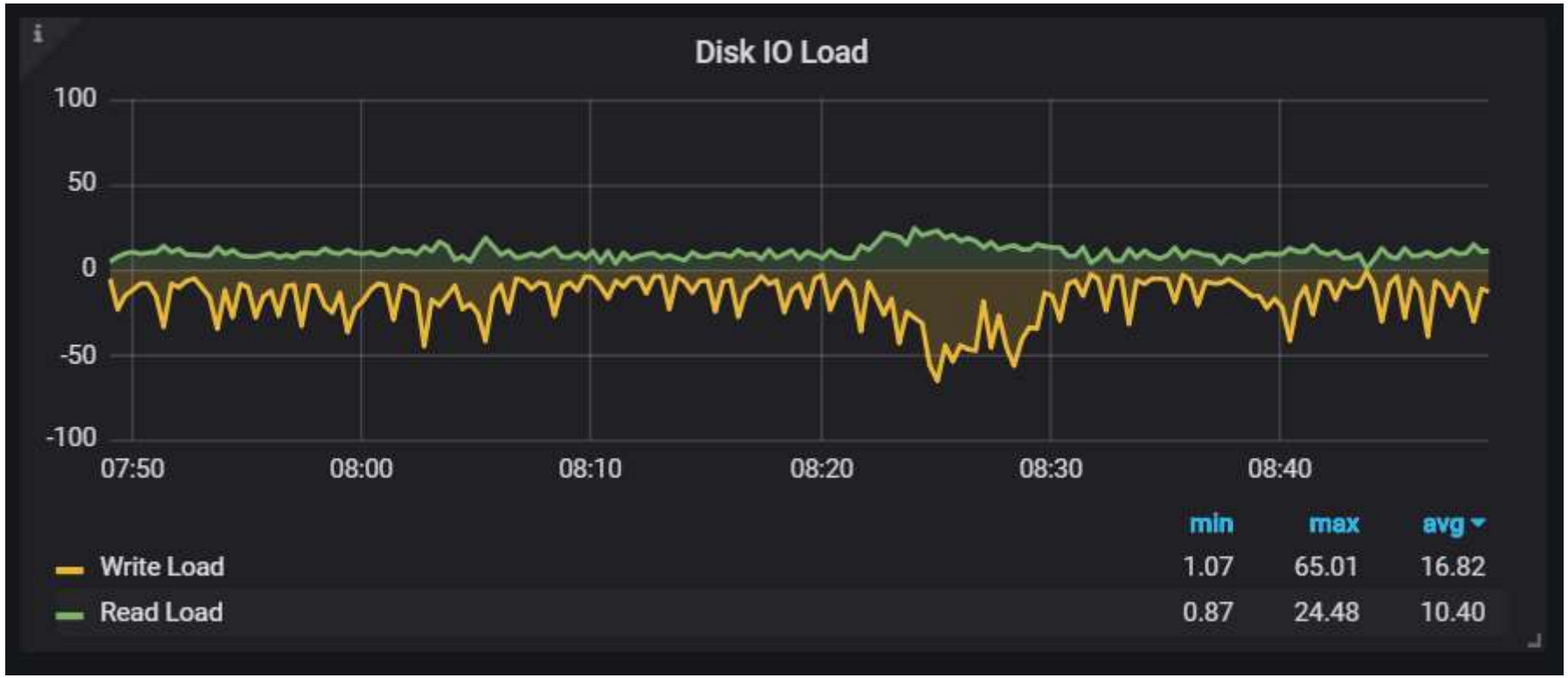


PERCONA

Swap Activity

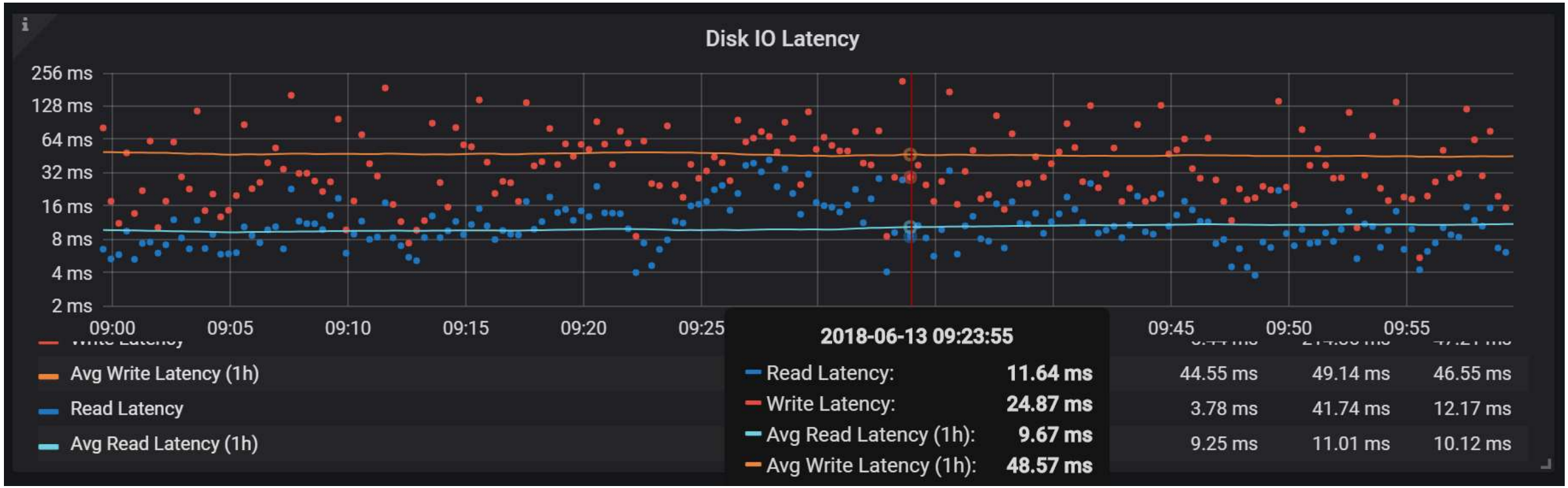
PERCONA

Disk Load



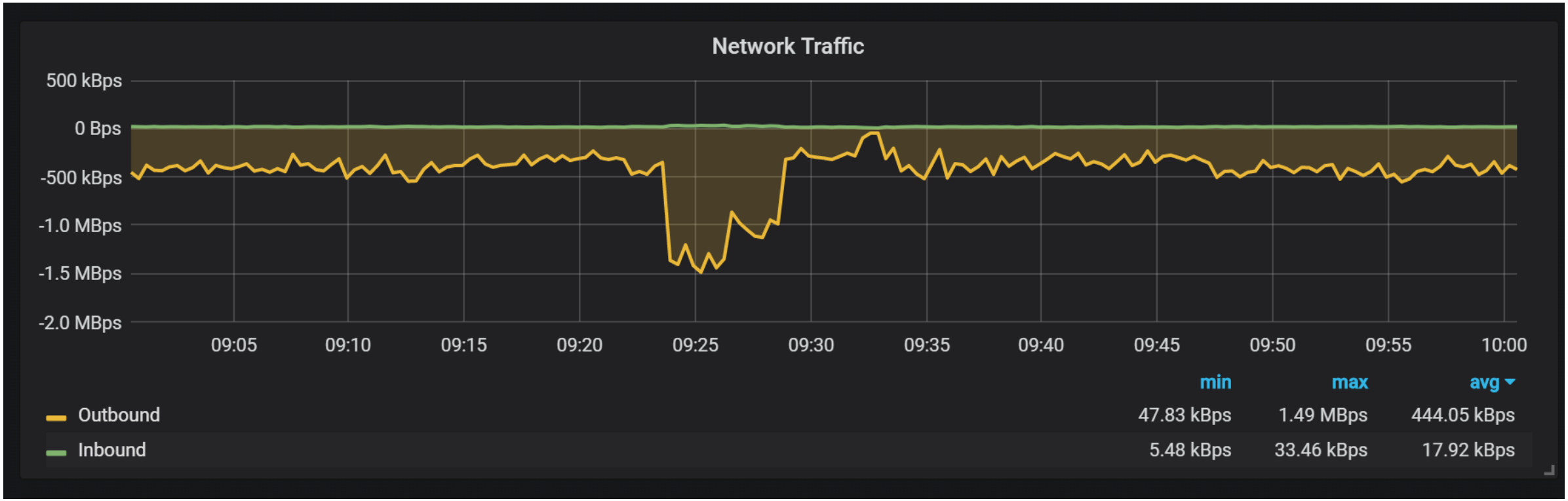
PERCONA

Disk IO Latency



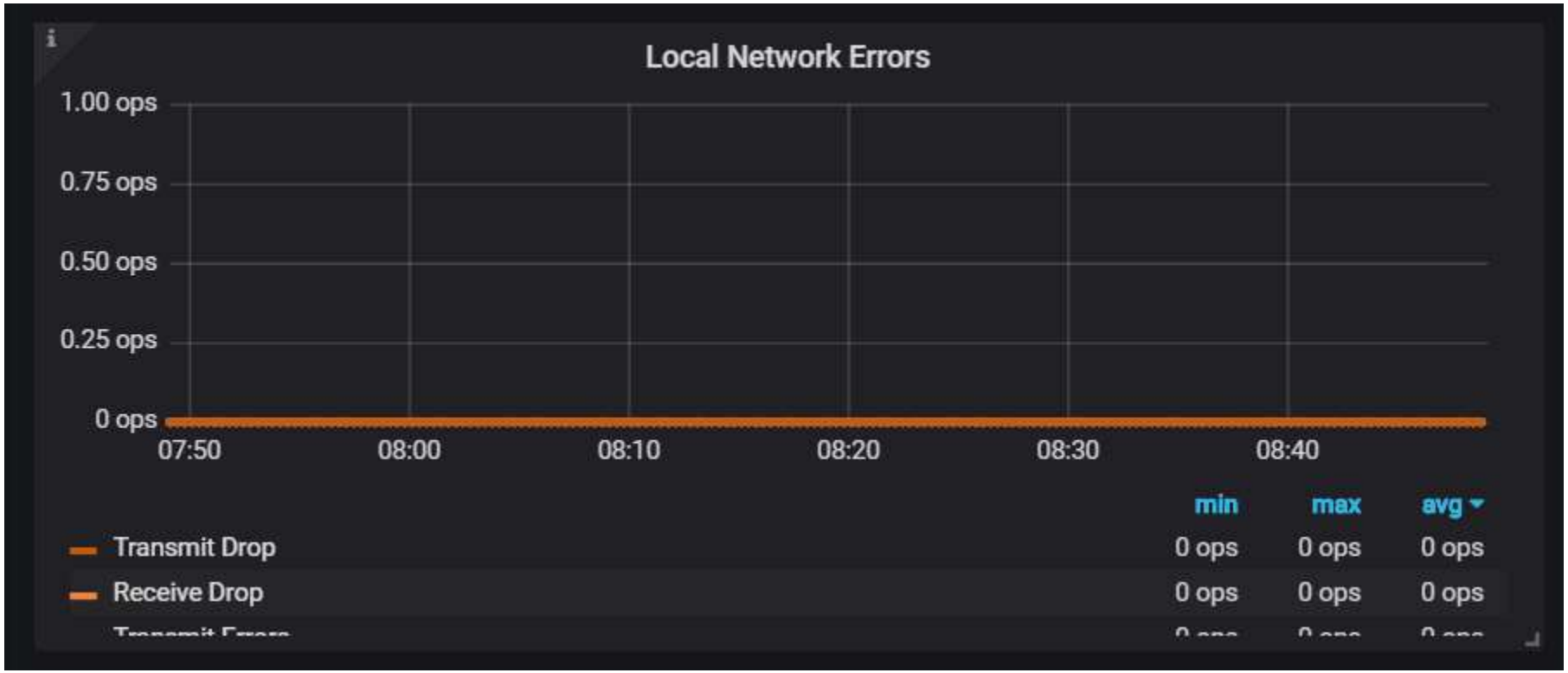
PERCONA

Network “Utilization”



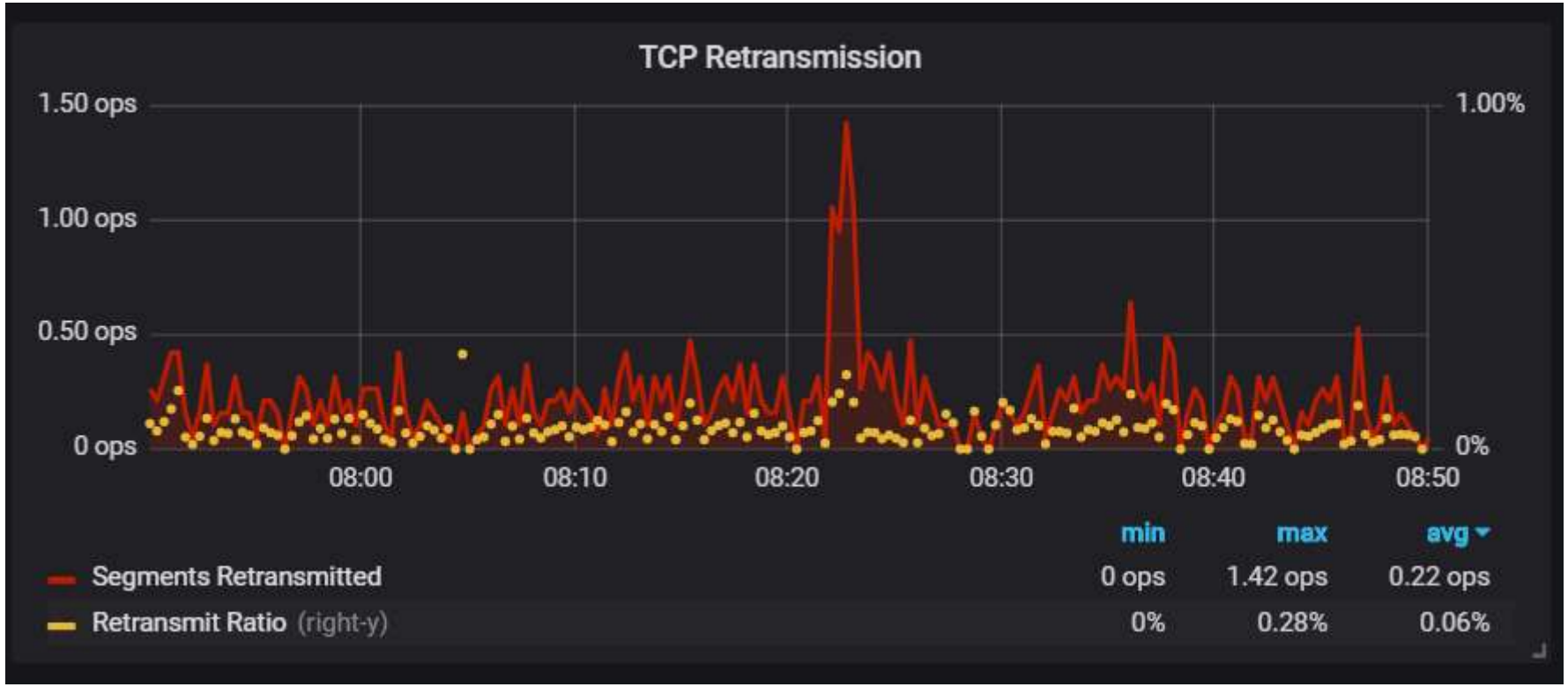
PERCONA

Network: Local



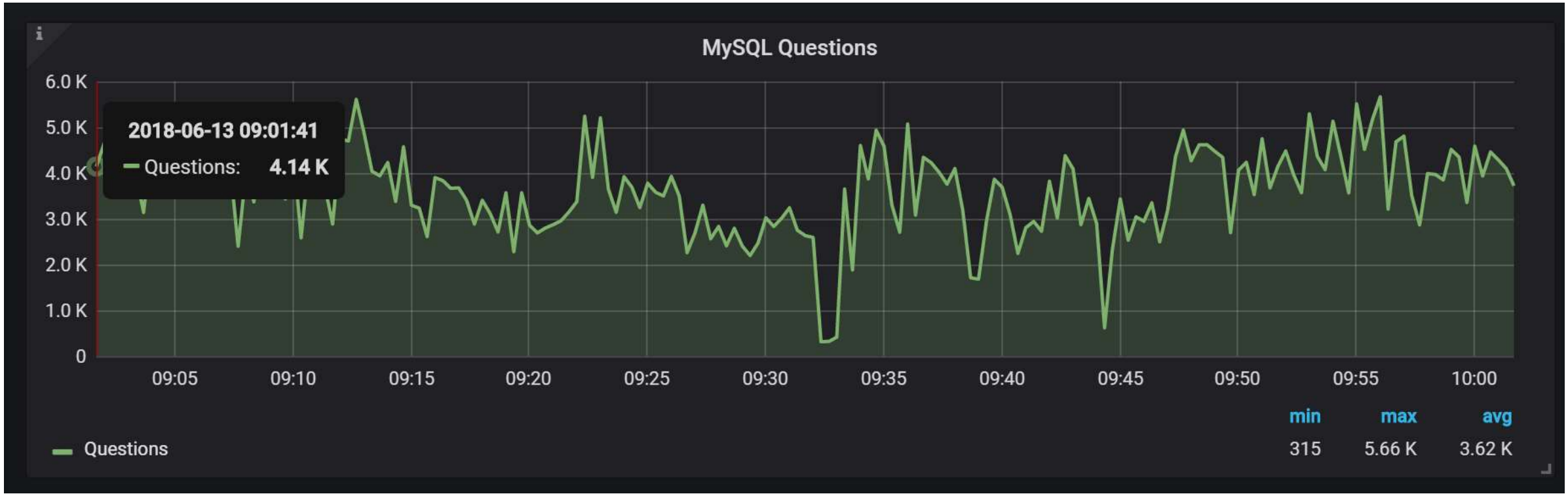
PERCONA

Netwok: “Global”



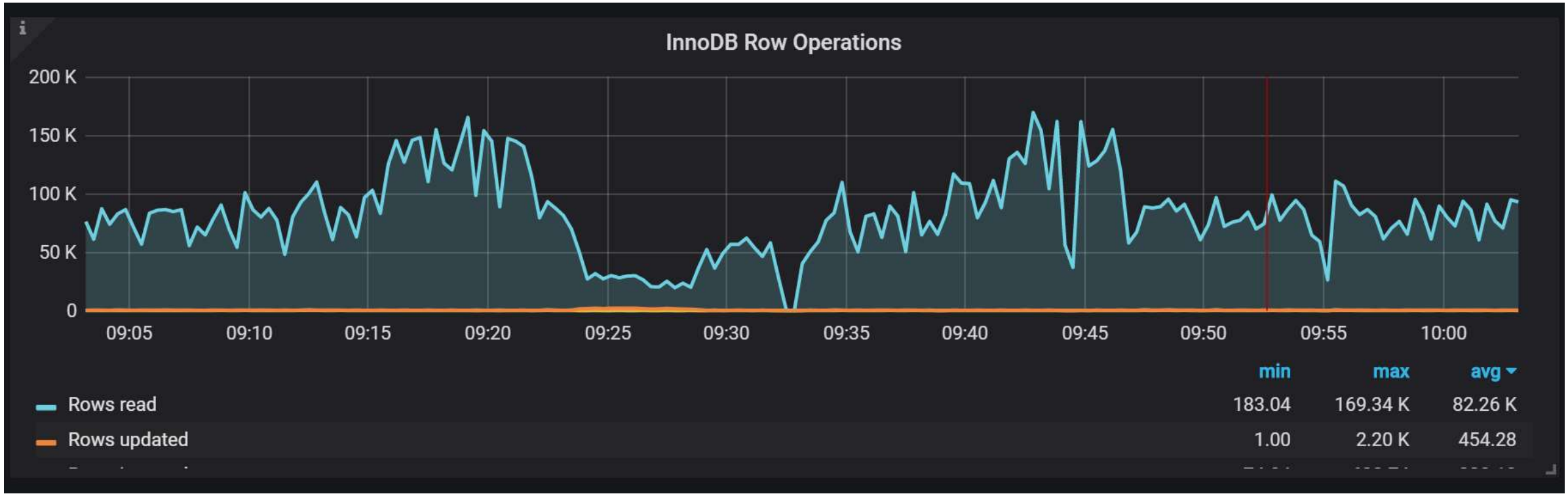
PERCONA

MySQL Questions



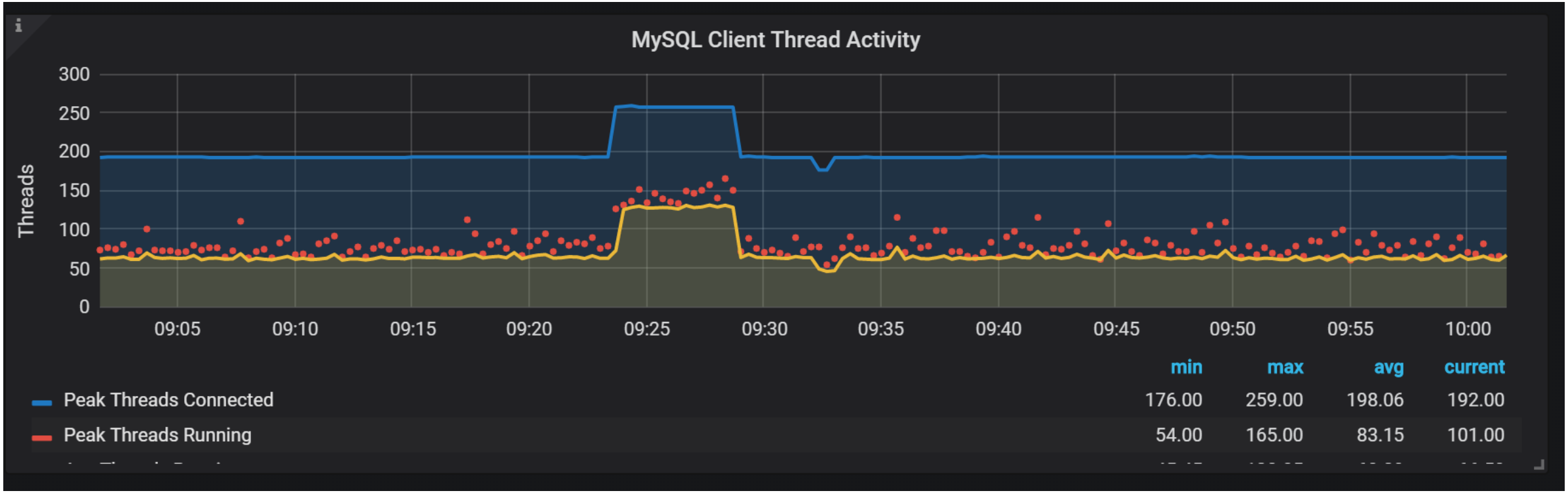
PERCONA

Lower Level Work Done



PERCONA

MySQL Saturation



PERCONA

RED Method

RED Method Focus

Microservices

“Cattle not Pets”

Mapping to Resources can be fluid

RED Method

For every Service Check Request check these are within SLO

- Rate
- Error (Rate)
- Duration (Distribution)

RED Method for Databases

Looking at Service Level

Looking at Individual Database Servers

Can be applied to Components/Resources

Can be applied to individual Types of Queries

PERCONA

RED Method Benefits

Easily maps to what Developers Care About

Does not require as deep understanding of architecture

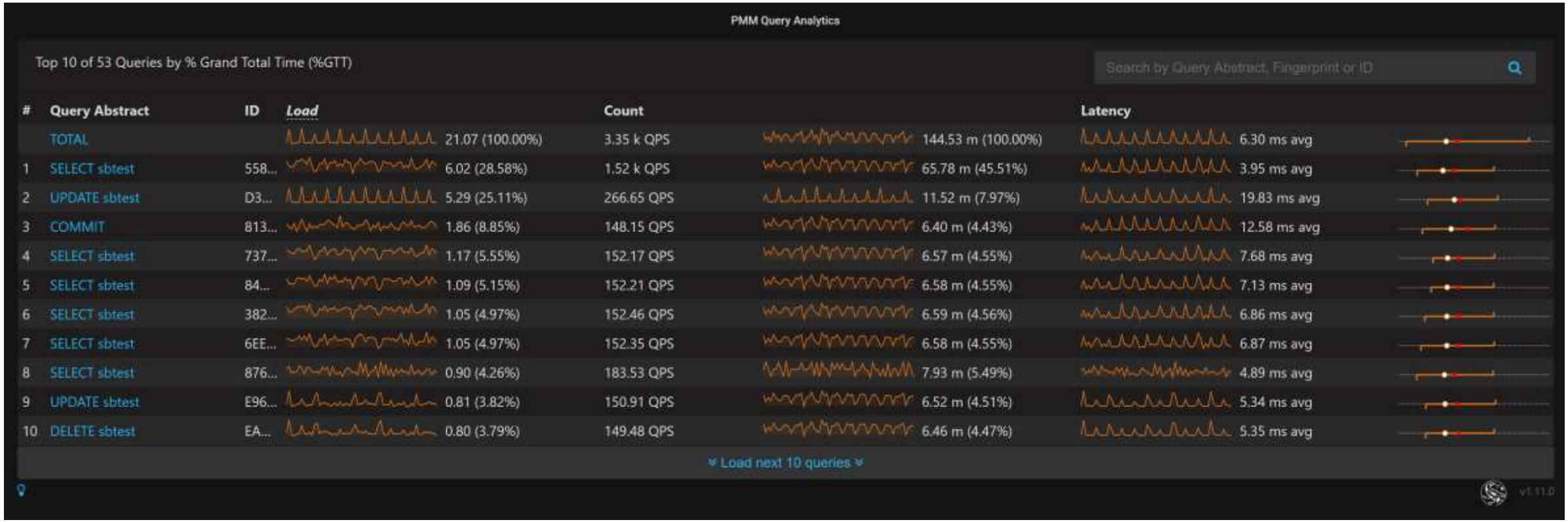
Does not need access to low lever resource monitoring

RED Method Drawbacks

Does not have as much tools and checklists support yet

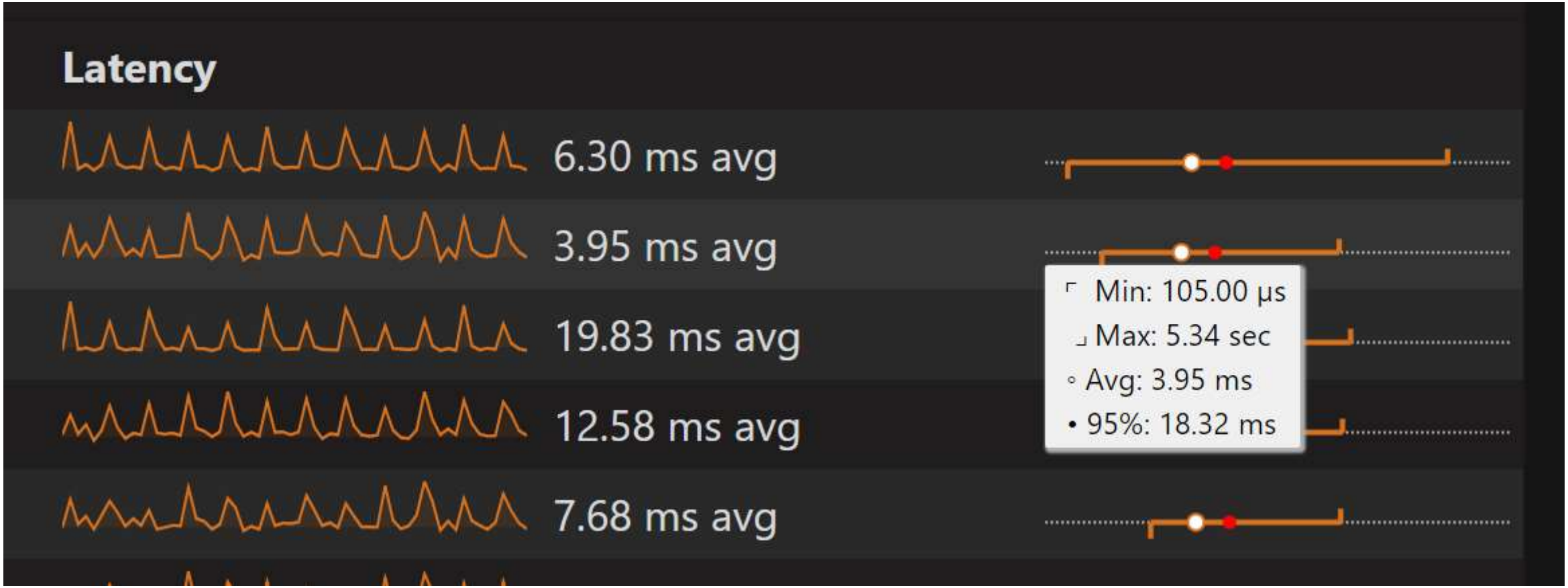
More focused on Answering WHAT rather WHY

Query Response Time



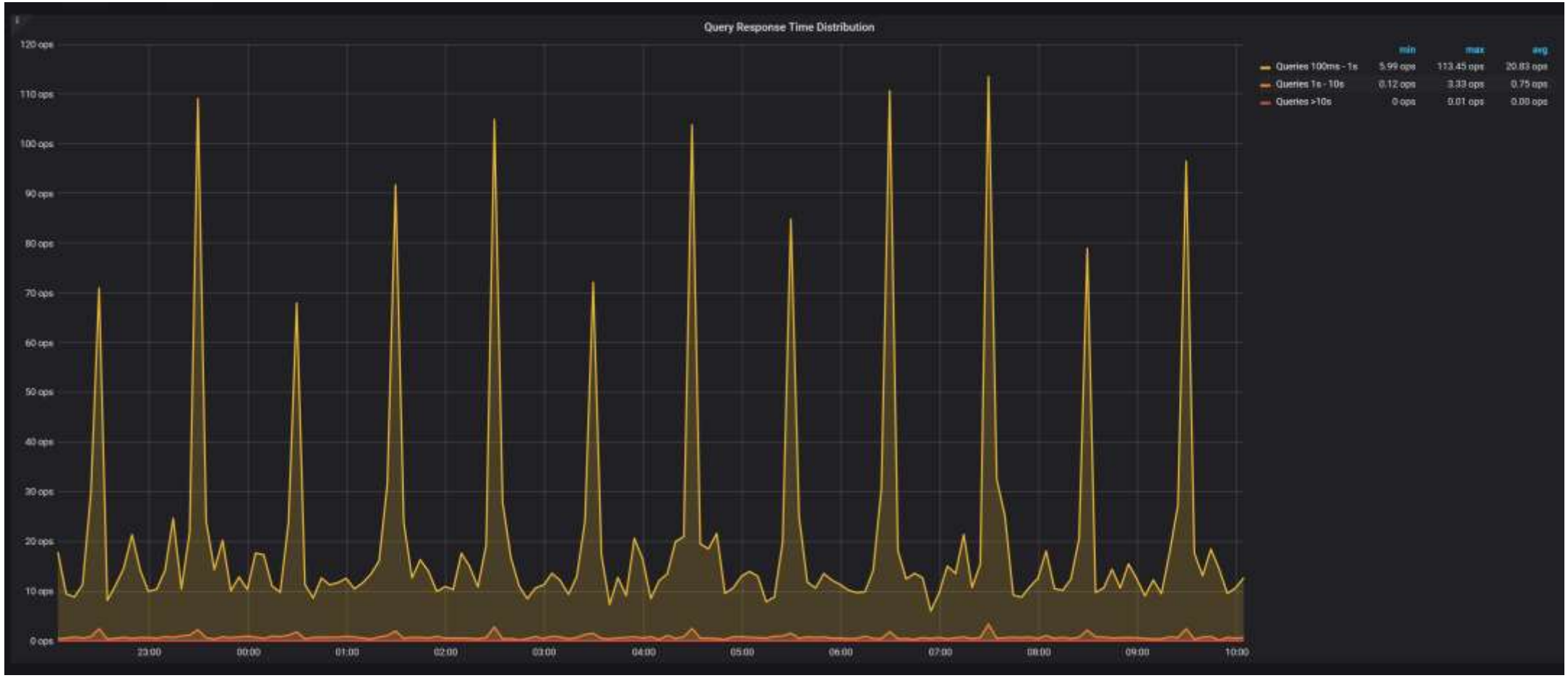
PERCONA

Response Time Details



PERCONA

Slow Queries Summary



PERCONA

Four Golden Signals

Focus

Monitoring Distributed Systems from SRE Book

To Be used for Alerting, Troubleshooting, Trend Analyses

Four Golden Signals

Latency	• Distribution not just Average; Latency for Successful requests vs Errors
Traffic	• How much Demand is being placed on the System
Errors	• Error Codes are Easy; Bad Content is hard
Saturation	• How Full your system “capacity”. Forecast when Possible.

For Golden Signals and Databases

Latency	• Query Response Time Distribution by Digest
Traffic	• Number of Queries of Specific kind Served
Errors	• Error codes but also Wrong Query Responses (hard)
Saturation	• Resource Usage but also Connections; Disk Space etc

Golden Signals Benefits

Methodology Well Tested by Google and Many Others

Good Resource Book Available

Golden Signals Drawbacks

Framework for Monitoring

No specific troubleshooting checklists

Take Aways

Great innovation in Performance Analyses Methods

Significant Overlap – Looking at mainly same things, different way

Check Out http://per.co.na/careers

Call for Papers Now Open!

Submit by July 15

www.percona.com/cfp

PERCONA

LIVE EUROPE

AMSTERDAM

#PERCONALIVE

Thank You!