PostgreSQL Monitoring

Using modern software stacks

Roman Fišer

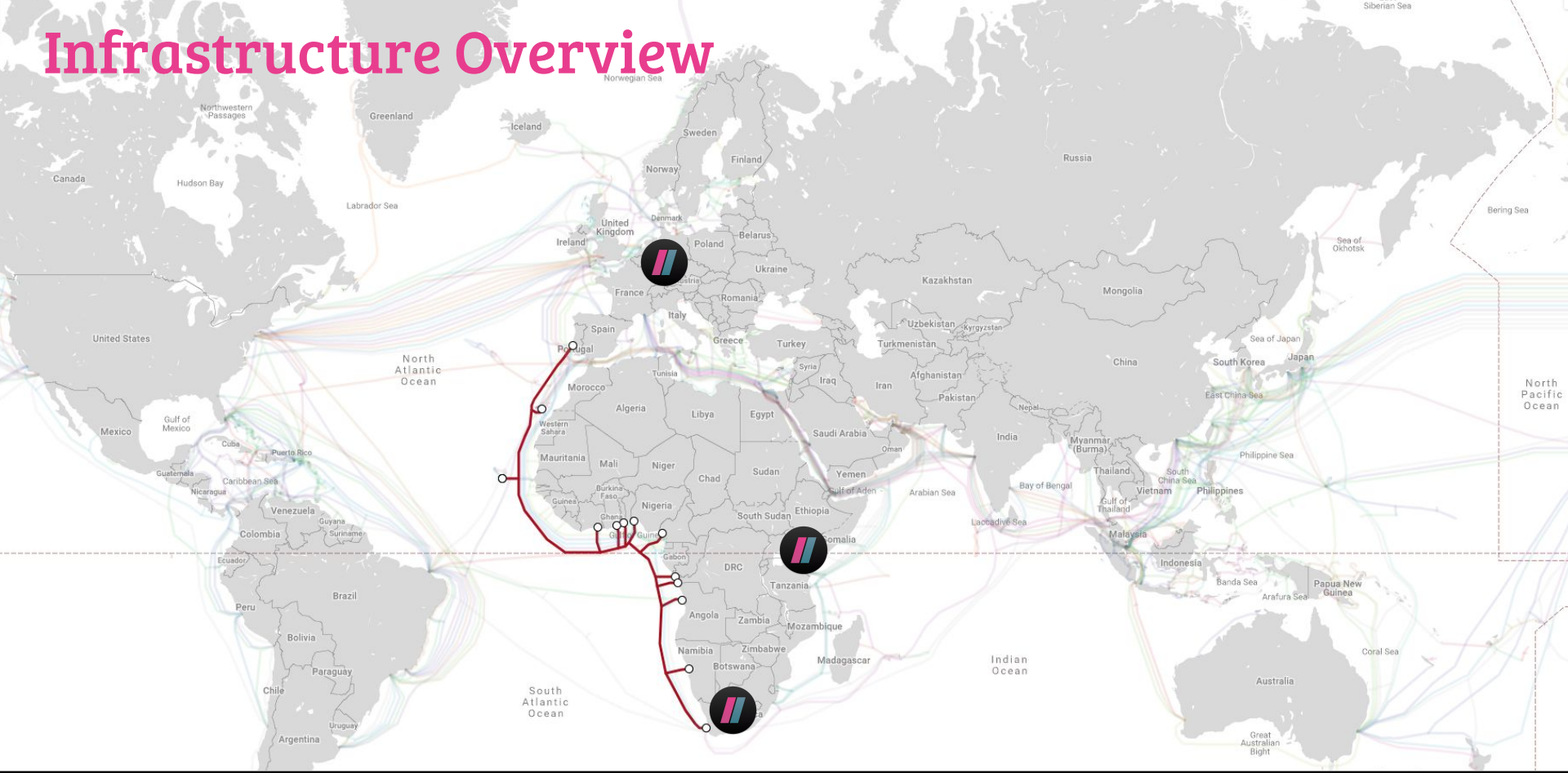
About me

- Roman Fišer
- Head of Infrastructure
- Showmax Engineering
- roman.fiser@showmax.com

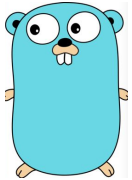
What is Showmax?

- VOD (Video On Demand) service
- Focusing on clients in Africa
- Engineering in Prague
- Based on Open source technologies

Infrastructure Overview



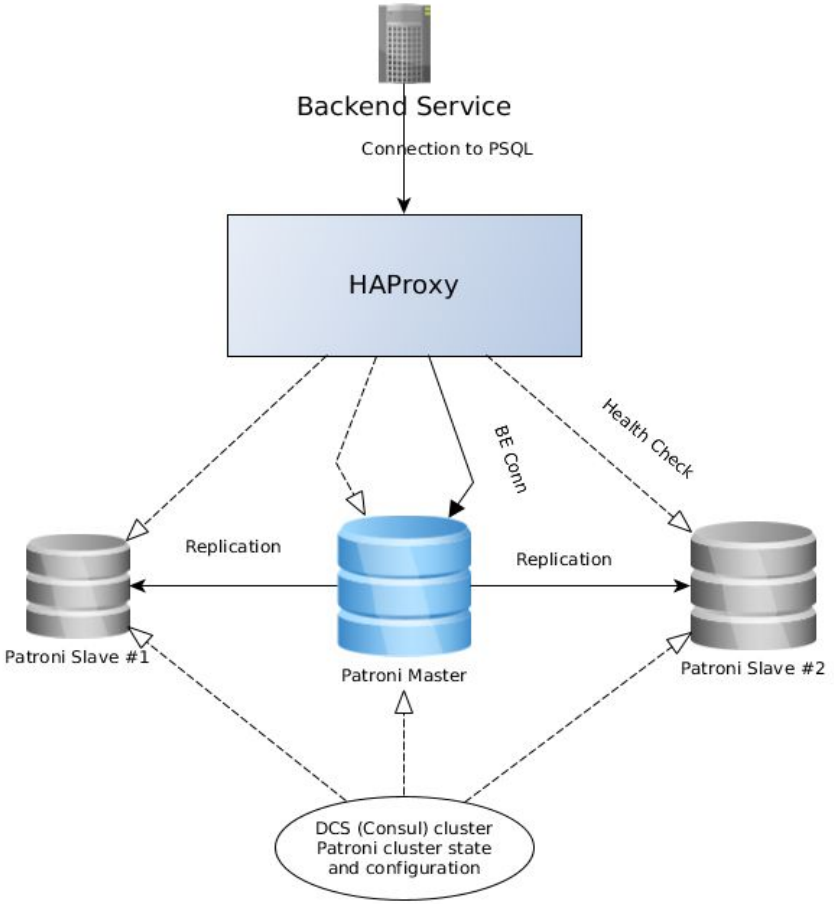
Open Source



docker
PostgreSQL
puppet
HAPROXY
Stack
NGINX
VARNISH
SOFTWARE

PostgreSQL

- Cluster management handled by **Patroni**
  - High-availability
    - Automatic master/slave election
    - Auto-failover
  - Streaming replication
  - Written in python & Open Source
    - Multiple patches to upstream
- Secrets management w Vault
- Barman for Backups



PostgreSQL @ Showmax

- Backend - data store for Showmax business entities
  - Microservice architecture, each DB has RESTful microservice
- CMS
  - Stores CMS in PostgreSQL (then denormalized to Elastic)
  - Cache invalidations
- Analytics DWH
  - Copy over from all databases
  - Events digestion

What should the monitoring provide?

- Alerting based on Global properties
  - Do I have at least one operational read-only replica?
- Alerting based on historical data
  - Compare current data with data 7 days old
- Alerting based on comparison between services
  - Ratio of rates of transactions to application requests

What type of metrics are important?

- Four SRE Golden Signals
  - Latency
    - The time it takes to service a request
  - Traffic
    - A measure of how much demand is being placed on your system
  - Error
    - The rate of requests that fail
  - Saturations
    - How "full" your service is

Latency

- pg_stat_statements
  - Average call time
  - Maximum call time
- Replication delay

Traffic

- Pg_stat_statements
  - Calls
  - Returned rows
- Network traffic
- System IO Statistics (IOPS, traffic)

Errors

- Rollback / Commit ratio
- Deadlocks

Saturation

- num_backends / max_connections
- High IO utilization (iowait, await)
- High CPU utilization
- Checkpoints
- Tempfile usage
- Disk usage (free space, inodes)

Other important metrics

- Long running idle in transaction
- Blocked autovacuum
- Error events in PostgreSQL log
  - Server crashes
  - I/O Errors
  - Data corruption
  - Index corruption

Tools - Prometheus Stack

- Prometheus
  - Prometheus is a time-series database. Suitable for white-box monitoring
- Alert Manager
  - Part of the Prometheus project. Used for Alerting.
- Exporters
  - Patroni_exporter, Postgres_exporter - Exports PostgreSQL metrics
  - Node_exporter - Expose OS metrics
- Grafana
  - Web frontend for Prometheus data

Prometheus vs Nagios/Icinga

Nagios/Icinga

- Focus on black-box monitoring
- Checks usually complicated bash scripts
- Can’t base alerts on relations between different metric types

Prometheus

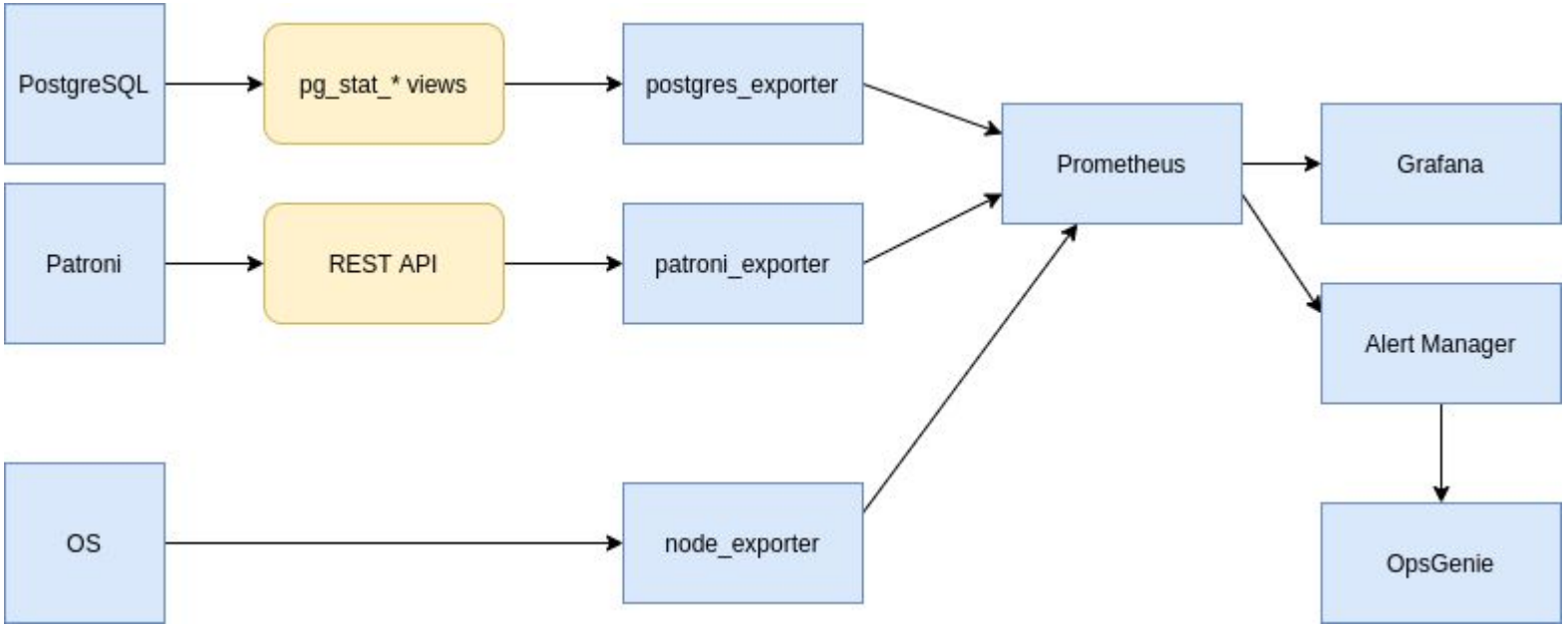
- Promotes white-box monitoring
- High-performance TSDB
- Cloud-native ready with multiple service discovery providers
- Standardized interface for exporters

Prometheus

- Monitoring based on metrics with metadata (CPU, RAM, disk IO, disk utilization, etc.)
- Custom labels for metrics
- Functions to filter, change, remove …. metadata while fetching them
- Multiple exporters - expose data via HTTP API
- Effective data fetching:
  - Based on intervals measured in seconds
  - Million of data points
- Notifications can be reported via Email, Slack, etc.

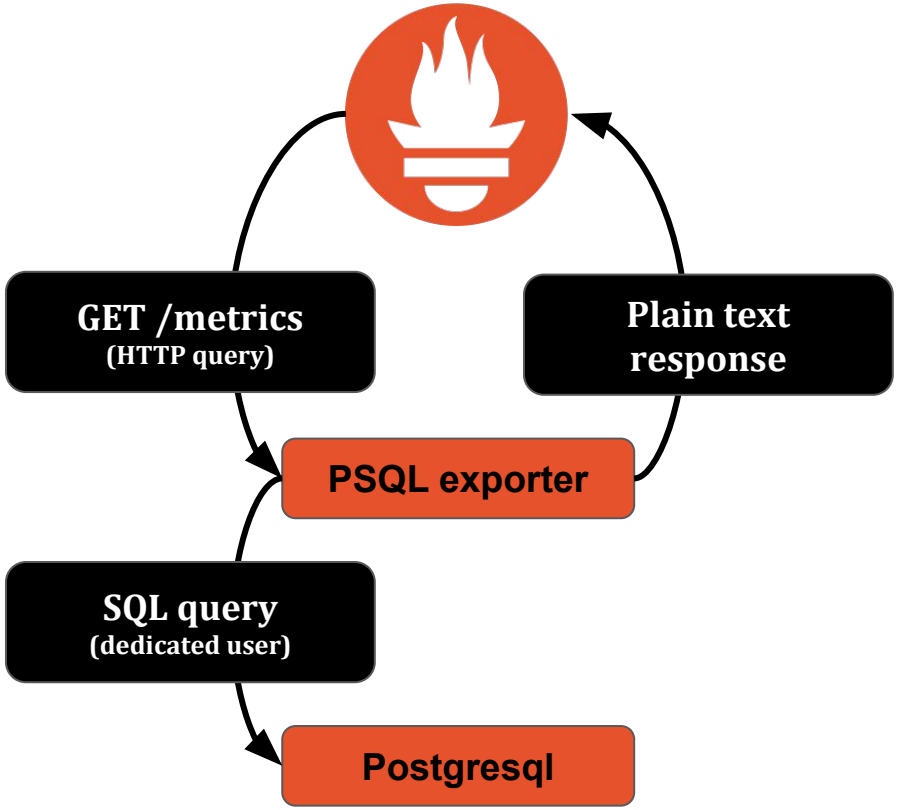
Prometheus
https://prometheus.io

Prometheus Pipeline

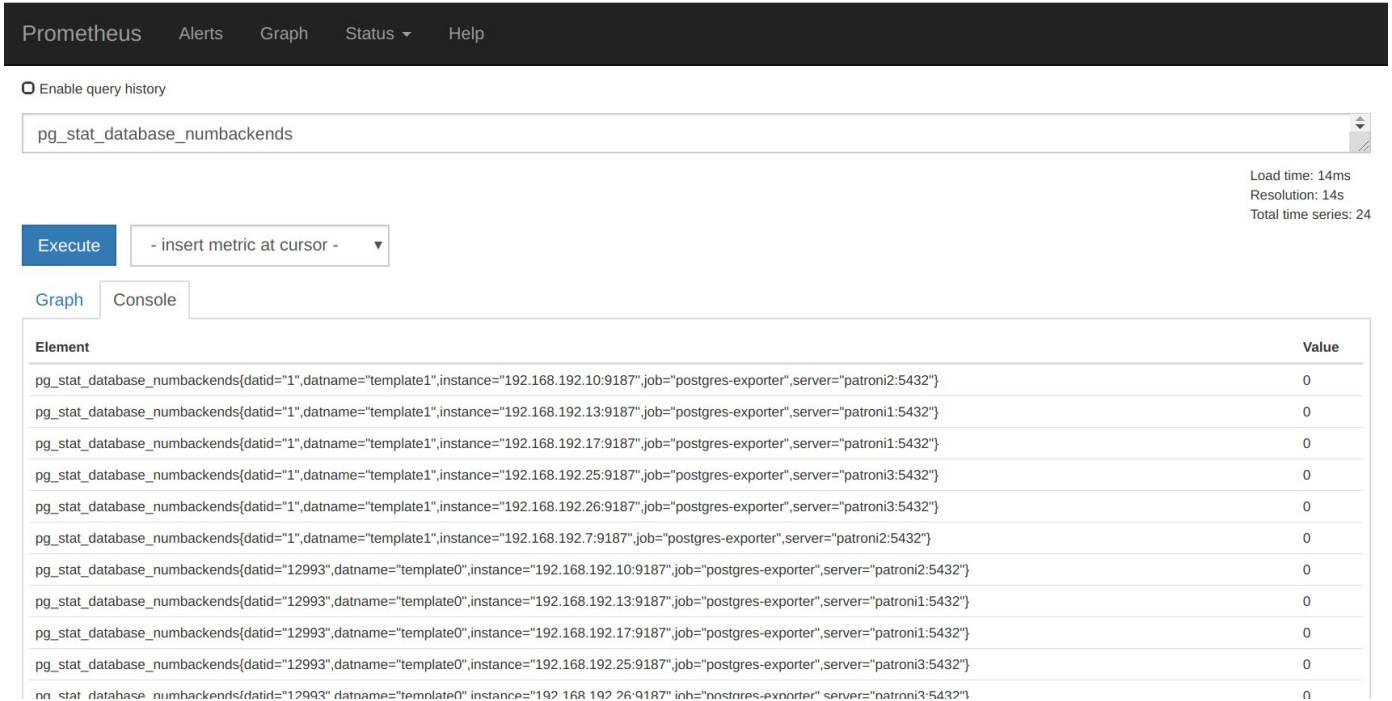


Prometheus

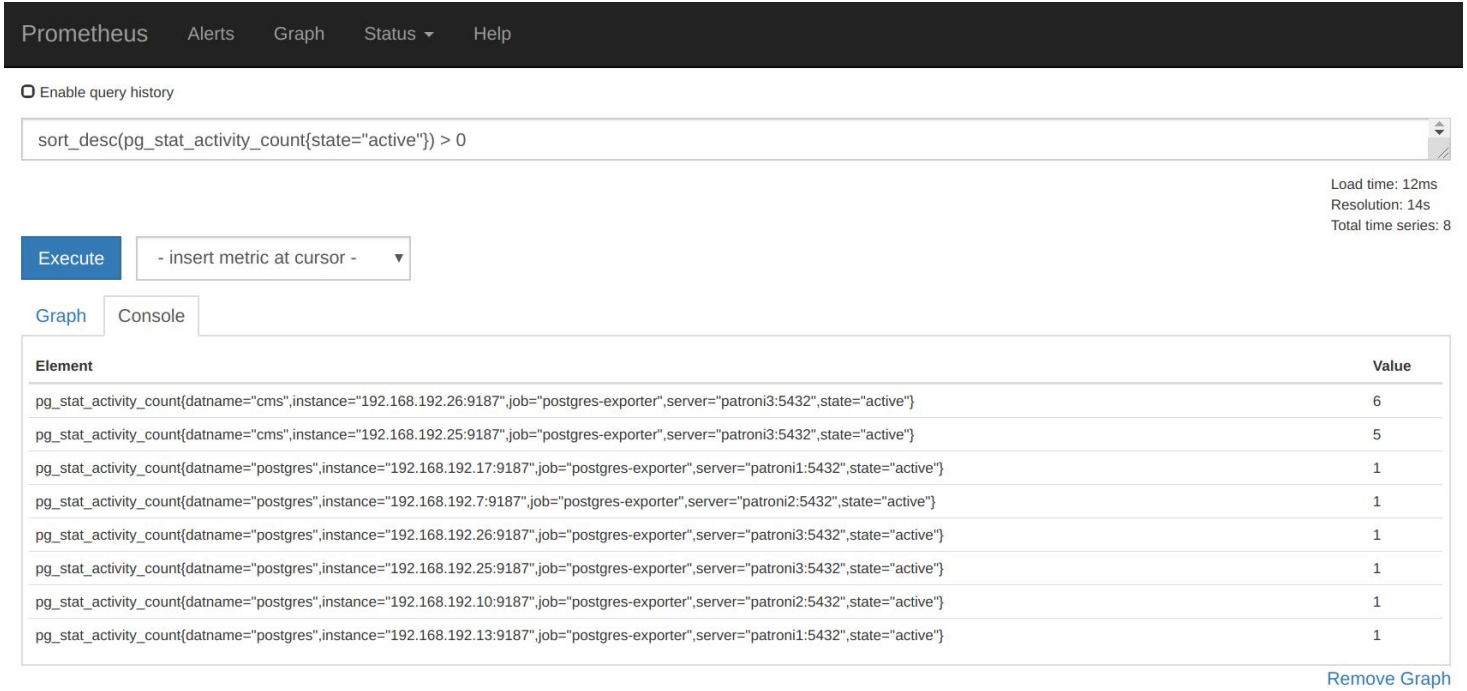

How does it work?


GET /metrics
(HTTP query)
Plain text
response
PSQL exporter
SQL query
(dedicated user)
Postgresql

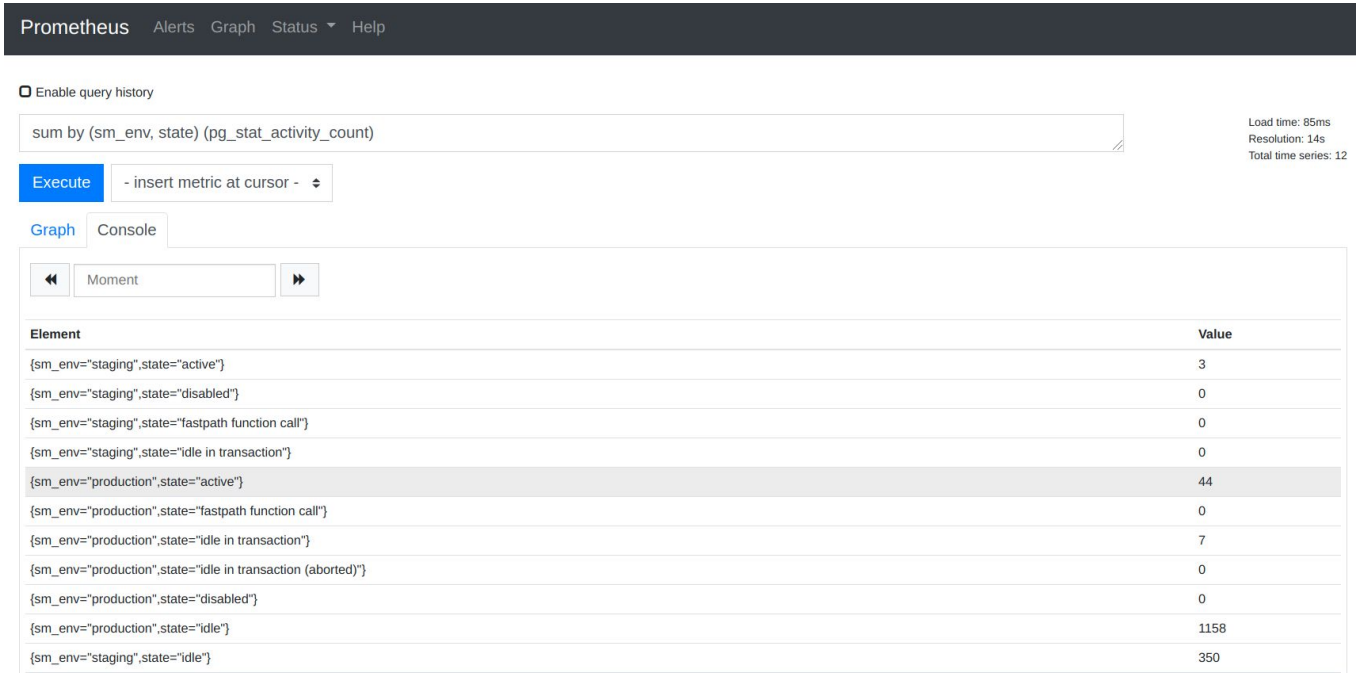
PromQL


Prometheus
Alerts
Graph
Status
Help
Enable query history
pg_stat_database_numbackends
Load time: 14ms
Resolution: 14s
Total time series: 24
Execute
- insert metric at cursor -
Graph
Console
Element
Value
pg_stat_database_numbackends{datid="1",datname="template1",instance="192.168.192.10:9187",job="postgres-exporter",server="patroni2:5432"} 0
pg_stat_database_numbackends{datid="1",datname="template1",instance="192.168.192.13:9187",job="postgres-exporter",server="patroni1:5432"} 0
pg_stat_database_numbackends{datid="1",datname="template1",instance="192.168.192.17:9187",job="postgres-exporter",server="patroni1:5432"} 0
pg_stat_database_numbackends{datid="1",datname="template1",instance="192.168.192.25:9187",job="postgres-exporter",server="patroni3:5432"} 0
pg_stat_database_numbackends{datid="1",datname="template1",instance="192.168.192.26:9187",job="postgres-exporter",server="patroni3:5432"} 0
pg_stat_database_numbackends{datid="1",datname="template1",instance="192.168.192.7:9187",job="postgres-exporter",server="patroni2:5432"} 0
pg_stat_database_numbackends{datid="12993",datname="template0",instance="192.168.192.10:9187",job="postgres-exporter",server="patroni2:5432"} 0
pg_stat_database_numbackends{datid="12993",datname="template0",instance="192.168.192.13:9187",job="postgres-exporter",server="patroni1:5432"} 0
pg_stat_database_numbackends{datid="12993",datname="template0",instance="192.168.192.17:9187",job="postgres-exporter",server="patroni1:5432"} 0
pg_stat_database_numbackends{datid="12993",datname="template0",instance="192.168.192.25:9187",job="postgres-exporter",server="patroni3:5432"} 0

PromQL


Prometheus
Alerts
Graph
Status
Help
Enable query history
sort_desc(pg_stat_activity_count{state="active"}) > 0
Load time: 12ms
Resolution: 14s
Total time series: 8
Execute
- insert metric at cursor -
Graph
Console
Element
Value
pg_stat_activity_count{datname="cms",instance="192.168.192.26:9187",job="postgres-exporter",server="patroni3:5432",state="active"} 6
pg_stat_activity_count{datname="cms",instance="192.168.192.25:9187",job="postgres-exporter",server="patroni3:5432",state="active"} 5
pg_stat_activity_count{datname="postgres",instance="192.168.192.17:9187",job="postgres-exporter",server="patroni1:5432",state="active"} 1
pg_stat_activity_count{datname="postgres",instance="192.168.192.7:9187",job="postgres-exporter",server="patroni2:5432",state="active"} 1
pg_stat_activity_count{datname="postgres",instance="192.168.192.26:9187",job="postgres-exporter",server="patroni3:5432",state="active"} 1
pg_stat_activity_count{datname="postgres",instance="192.168.192.25:9187",job="postgres-exporter",server="patroni3:5432",state="active"} 1
pg_stat_activity_count{datname="postgres",instance="192.168.192.10:9187",job="postgres-exporter",server="patroni2:5432",state="active"} 1
pg_stat_activity_count{datname="postgres",instance="192.168.192.13:9187",job="postgres-exporter",server="patroni1:5432",state="active"} 1
Remove Graph

PromQL


Prometheus
Alerts
Graph
Status
Help
Enable query history
sum by (sm_env, state) (pg_stat_activity_count)
Load time: 85ms
Resolution: 14s
Total time series: 12
Execute
- insert metric at cursor -
Graph
Console
Moment
Element
Value
{sm_env="staging",state="active"} 3
{sm_env="staging",state="disabled"} 0
{sm_env="staging",state="fastpath function call"} 0
{sm_env="staging",state="idle in transaction"} 0
{sm_env="production",state="active"} 44
{sm_env="production",state="fastpath function call"} 0
{sm_env="production",state="idle in transaction"} 7
{sm_env="production",state="idle in transaction (aborted)"} 0
{sm_env="production",state="disabled"} 0
{sm_env="production",state="idle"} 1158
{sm_env="staging",state="idle"} 350

Alert Manager Rules - Latency

```
- alert: PgSQLSlowLoginQuery
  expr: avg_over_time(pg_stat_statements_mean_time_seconds{datname="cms", queryid="3985044216"}[1m])
> 100
  for: 5m
  labels:
    severity: critical
    team: ops
  annotations:
    summary: "Slow login query (instance {{ $labels.instance }})"
    description: |
      Login queries are too slow (> 30s). Average is {{ $value }}.
      Check the PostgreSQL instance {{ $labels.instance }}
    runbook: pgsql@pgsqlslowloginquery
    title: PgSQLSlowLoginQuery
```

Alert Manager Rules - Latency

```
- alert: PgSQLReplicationLagIsTooBig
  expr: pg_replication_lag{instance!~"^(ba-patroni|analytics-patroni).*"} > 300 and pg_is_in_recovery == 1
  for: 15m
  labels:
    severity: critical
    team: ops
  annotations:
    description:
      Replication lag on PostgreSQL Slave {{ $labels.instance }} is
      {{ humanizeDuration $value }}. If lag is too big, it might be
      impossible for Slave to recover, it might not be considered as a new
      Patroni leader, or data loss could occur should such Slave be chosen as next
      leader.
    summary: PostgreSQL Slave Replication lag is too big.
    runbook: pgsql#pgsqlreplicationlagistoobig
    title: PgSQLReplicationLagIsTooBig
```

Alert Manager Rules - Traffic

```
- alert: PgSQLCommitRateTooLow
  expr: |
    rate(pg_stat_database_xact_commit{datname="oauth", sm_env="prod"}[5m]) < 200
  for: 5m
  labels:
    severity: warn
    team: ops
  annotations:
    description: |
      Commit Rate {{$labels.instance}} for database {{$labels.datname}}
      is {{$value}} which is suspiciously low.
    runbook: pgsql#pgsqlcommitrateislow
    title: PgSQLCommitRateTooLow
```

Alert Manager Rules - Saturation

```
- alert: PgSQLNumberOfConnectionsHigh
  expr: (100 * (sum(pg_stat_database_numbackends) by (instance, job) / pg_settings_max_connections)) > 90
  for: 10m
  labels:
    severity: critical
    team: ops
  annotations:
    description:
      Number of active/open connections to PostgreSQL on {{ $labels.instance }}
      is {{ $value }}. It's possible PostgreSQL won't be able to accept any
      new connections.
    summary: Number of active connections to Postgresql too high.
    runbook: pgsql#pgsqlnumberofconnectionshigh
    title: PgSQLNumberOfConnectionsHigh
```

Alert Manager Rules - Errors

```
- alert: PgSQLRollbackRateTooHigh
  expr: |
    rate(pg_stat_database_xact_rollback{datname="oauth"}[5m])
      / ON(instance, datname)
    rate(pg_stat_database_xact_commit{datname="oauth"}[5m])
      > 0.05
  for: 5m
  labels:
    severity: warn
    team: ops
  annotations:
    description: |
      Ratio of transactions being aborted compared to committed is
      {{$value | printf "%.2f" }} on {{$labels.instance}}
    runbook: pgsql@pgsqlrollbackrateishigh
    title: PgSQLRollbackRateTooHigh
```

Alert Manager Rules - Errors

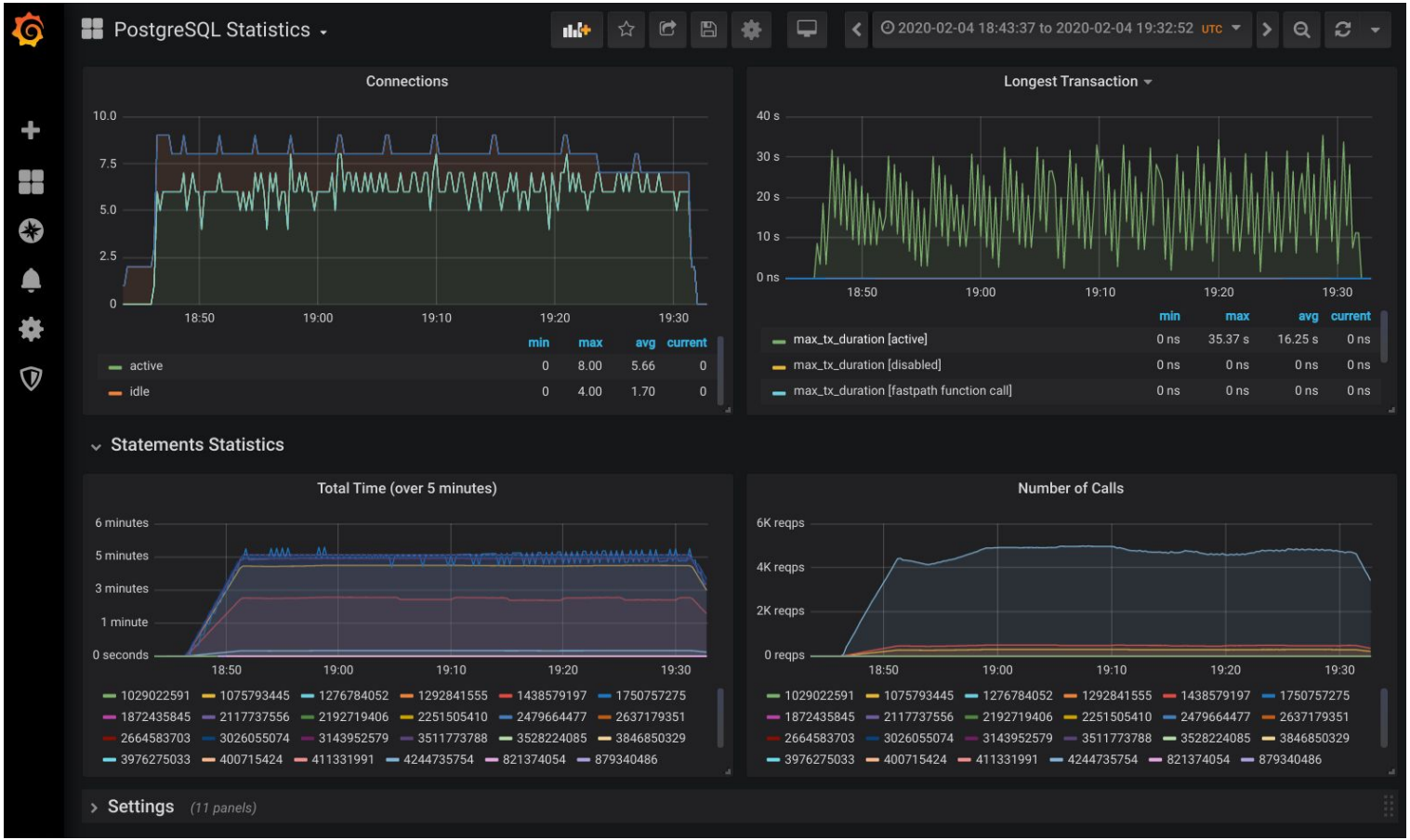
```
- alert: PgSQLDeadLocks
  expr: rate(pg_stat_database_deadlocks{datname!~"template.*|postgres"}[1m]) > 0
  for: 5m
  labels:
    severity: warn
    team: ops
  annotations:
    description: |
      Deadlocks has been detected on PostgreSQL {{ $labels.instance }}.
      Number of deadlocks: {{ $value }}
    summary: "Dead locks (instance {{ $labels.instance }})"
    runbook: pgsql#pgsqldeadlocksdetected
    title: PgSQLDeadLocks
```

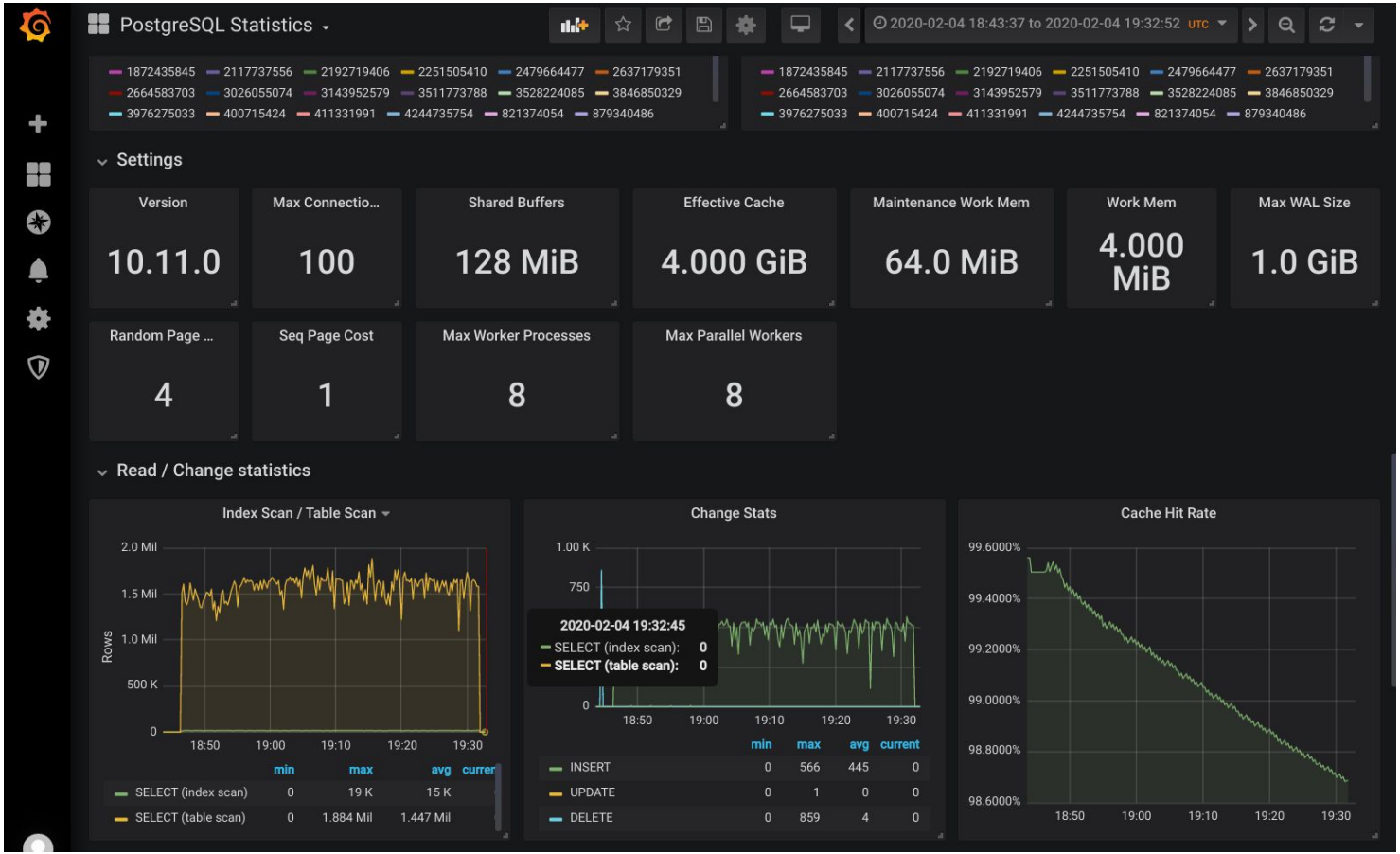
Alert Manager Rules - Errors

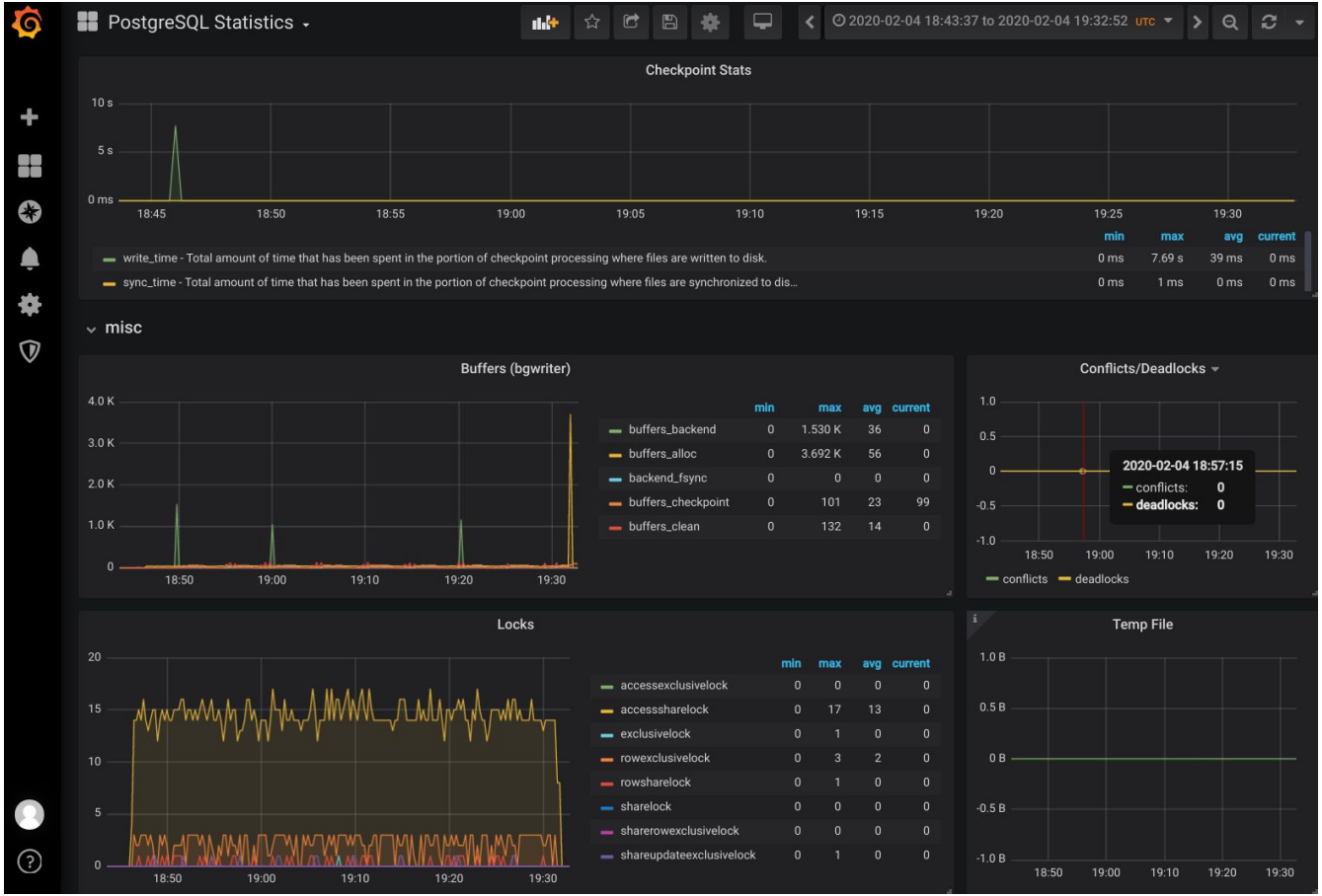
```
- alert: PgSQLTableNotVaccumed
  expr: time() - pg_stat_user_tables_last_autovacuum{datname="oauth", sm_env="prod"} > 60 * 60 * 24
  for: 5m
  labels:
    severity: warn
    team: ops
  annotations:
    summary: "Table not vaccumed (instance {{ $labels.instance }})"
    description: |
      Table has not been vaccum for 24 hours {{ $labels.relname }}
      (vacuumed before {{ humanizeDuration $value }}). There may be not enough vacuum workers.
    runbook: pgsql#pgsqltablenotvacuumed
    title: PgSQLTableNotVaccumed
```

Grafana


PostgreSQL Statistics
2020-02-04 18:43:37 to 2020-02-04 19:32:52 UTC
Instance 192.168.192.17:9187
Database cms
Replication statistics
Instance type
Master
Replication lag (bytes)
2.5 K
2.0 K
1.5 K
1.0 K
500
0
18:45
18:50
18:55
19:00
19:05
19:10
19:25
19:30
2020-02-04 19:11:00
patroni2-streaming: 0
patroni2-streaming:
patroni3-streaming: 0
patroni3-streaming:
patroni2-streaming
patroni2-streaming
patroni3-streaming
patroni3-streaming
Database Activity
Database Activity
10
8
6
4
2
0
18:50
19:00
19:10
19:20
19:30
active
disabled
fastpath function call
idle
idle in transaction
idle in transaction (aborted)
Transactions
8 K
6 K
4 K
2 K
0
18:50
19:00
19:10
19:20
19:30
commits
rollbacks


PostgreSQL Statistics
2020-02-04 18:43:37 to 2020-02-04 19:32:52 UTC
Connections
10.0
7.5
5.0
2.5
0
18:50
19:00
19:10
19:20
19:30
min max avg current
active 0 8.00 5.66 0
idle 0 4.00 1.70 0
Longest Transaction
40 s
30 s
20 s
10 s
0 ns
18:50
19:00
19:10
19:20
19:30
min max avg current
max_tx_duration [active] 0 ns 35.37 s 16.25 s 0 ns
max_tx_duration [disabled] 0 ns 0 ns 0 ns 0 ns
max_tx_duration [fastpath function call] 0 ns 0 ns 0 ns 0 ns
Statements Statistics
Total Time (over 5 minutes)
6 minutes
5 minutes
3 minutes
1 minute
0 seconds
18:50
19:00
19:10
19:20
19:30
1029022591 1075793445 1276784052 1292841555 1438579197 1750757275
1872435845 2117737556 2192719406 2251505410 2479664477 2637179351
2664583703 3026055074 3143952579 3511773788 3528224085 3846850329
3976275033 400715424 411331991 4244735754 821374054 879340486
Number of Calls
6K reqps
4K reqps
2K reqps
0 reqps
18:50
19:00
19:10
19:20
19:30
1029022591 1075793445 1276784052 1292841555 1438579197 1750757275
1872435845 2117737556 2192719406 2251505410 2479664477 2637179351
2664583703 3026055074 3143952579 3511773788 3528224085 3846850329
3976275033 400715424 411331991 4244735754 821374054 879340486
Settings (11 panels)


PostgreSQL Statistics
2020-02-04 18:43:37 to 2020-02-04 19:32:52 UTC
1872435845 2117737556 2192719406 2251505410 2479664477 2637179351
2664583703 3026055074 3143952579 3511773788 3528224085 3846850329
3976275033 400715424 411331991 4244735754 821374054 879340486
Settings
Version
10.11.0
Max Connectio...
100
Shared Buffers
128 MiB
Effective Cache
4.000 GiB
Maintenance Work Mem
64.0 MiB
Work Mem
4.000 MiB
Max WAL Size
1.0 GiB
Random Page ...
4
Seq Page Cost
1
Max Worker Processes
8
Max Parallel Workers
8
Read / Change statistics
Index Scan / Table Scan
2.0 Mil
1.5 Mil
1.0 Mil
500 K
0
Rows
18:50
19:00
19:10
19:20
19:30
min max avg curre
SELECT (index scan) 0 19 K 15 K
SELECT (table scan) 0 1.884 Mil 1.447 Mil
Change Stats
1.00 K
750
0
2020-02-04 19:32:45
SELECT (index scan): 0
SELECT (table scan): 0
min max avg current
INSERT 0 566 445 0
UPDATE 0 1 0 0
DELETE 0 859 4 0
Cache Hit Rate
99.6000%
99.4000%
99.2000%
99.0000%
98.8000%
98.6000%


PostgreSQL Statistics
2020-02-04 18:43:37 to 2020-02-04 19:32:52 UTC
Checkpoint Stats
10 s
5 s
0 ms
18:45
18:50
18:55
19:00
19:05
19:10
19:15
19:20
19:25
19:30
min max avg current
write_time - Total amount of time that has been spent in the portion of checkpoint processing where files are written to disk. 0 ms 7.69 s 39 ms 0 ms
sync_time - Total amount of time that has been spent in the portion of checkpoint processing where files are synchronized to dis... 0 ms 1 ms 0 ms 0 ms
misc
Buffers (bgwriter)
4.0 K
3.0 K
2.0 K
1.0 K
0
18:50
19:00
19:10
19:20
19:30
min max avg current
buffers_backend 0 1.530 K 36 0
buffers_alloc 0 3.692 K 56 0
backend_fsync 0 0 0 0
buffers_checkpoint 0 101 23 99
buffers_clean 0 132 14 0
Conflicts/Deadlocks
1.0
0.5
0
-0.5
-1.0
2020-02-04 18:57:15
conflicts: 0
deadlocks: 0
18:50
19:00
19:10
19:20
19:30
conflicts
deadlocks
Locks
20
15
10
5
0
18:50
19:00
19:10
19:20
19:30
min max avg current
accessexclusivelock 0 0 0 0
accesssharelock 0 17 13 0
exclusivelock 0 1 0 0
rowexclusivelock 0 3 2 0
rowsharelock 0 1 0 0
sharelock 0 0 0 0
sharerowexclusivelock 0 0 0 0
shareupdateexclusivelock 0 1 0 0
Temp File
1.0 B
0.5 B
0 B
-0.5 B
-1.0 B
18:50
19:00
19:10
19:20
19:30

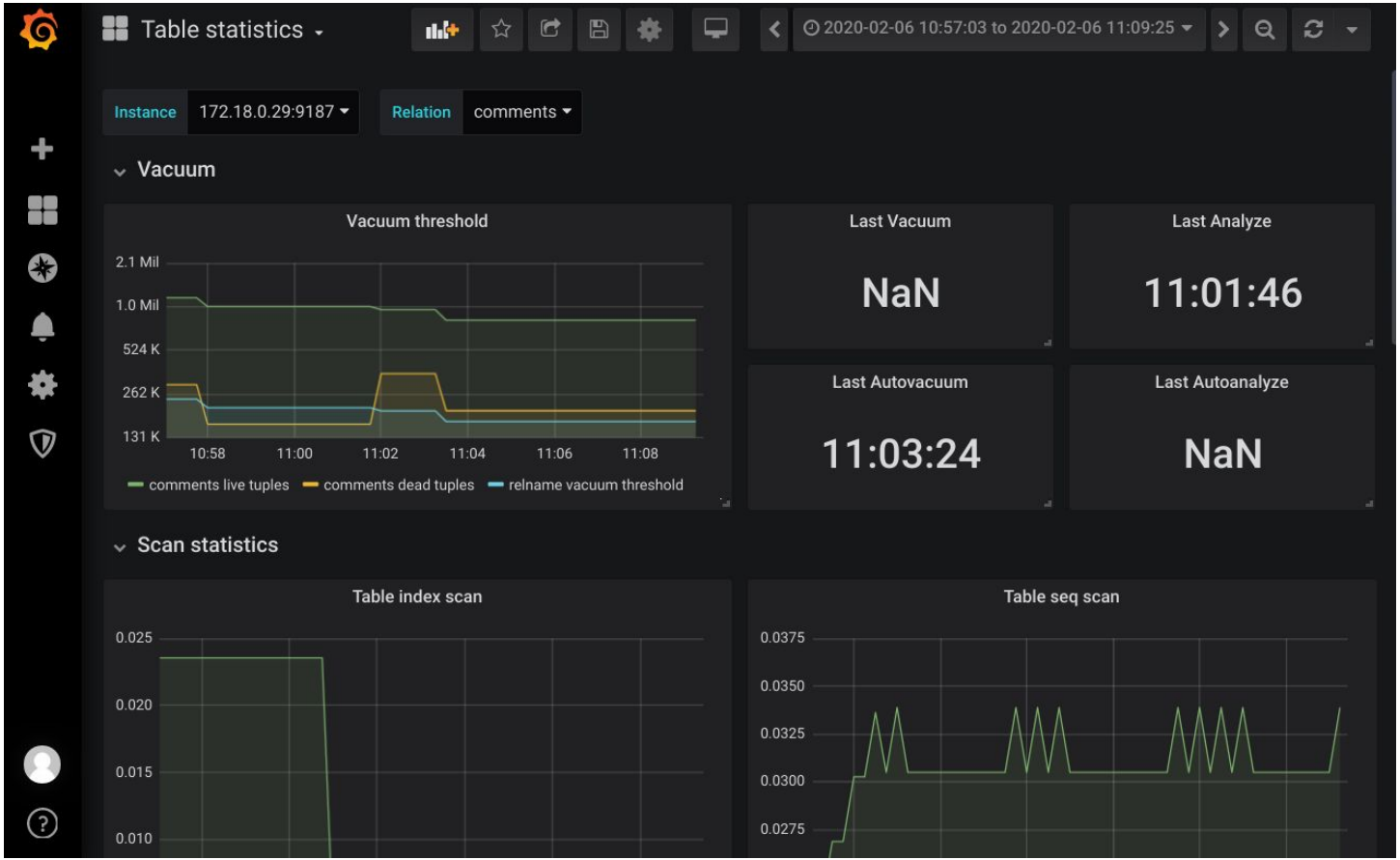


Table statistics
2020-02-06 10:57:03 to 2020-02-06 11:09:25
Instance
172.18.0.29:9187
Relation
comments
Vacuum
Vacuum threshold
2.1 Mil
1.0 Mil
524 K
262 K
131 K
10:58
11:00
11:02
11:04
11:06
11:08
comments live tuples
comments dead tuples
relname vacuum threshold
Last Vacuum
NaN
Last Analyze
11:01:46
Last Autovacuum
11:03:24
Last Autoanalyze
NaN
Scan statistics
Table index scan
0.025
0.020
0.015
0.010
Table seq scan
0.0375
0.0350
0.0325
0.0300
0.0275


Table statistics
2020-02-06 10:57:03 to 2020-02-06 11:09:25
Table changes
1.25 K
1.00 K
750
500
250
0
10:58
10:59
11:00
11:01
11:02
11:03
11:04
11:05
11:06
11:07
11:08
11:09
2020-02-06 11:04:30
comments inserts: 0
comments deletes: 678
comments updates: 0
comments hot updates: 0
comments inserts
comments deletes
comments updates
comments hot updates
Tables IO statistics
Heap
6 K
5 K
4 K
3 K
2 K
1 K
0
10:58
11:00
11:02
11:04
11:06
11:08
comments hits
comments read
Indexes
80
60
40
20
0
10:58
11:00
11:02
11:04
11:06
11:08
comments hits
comments read

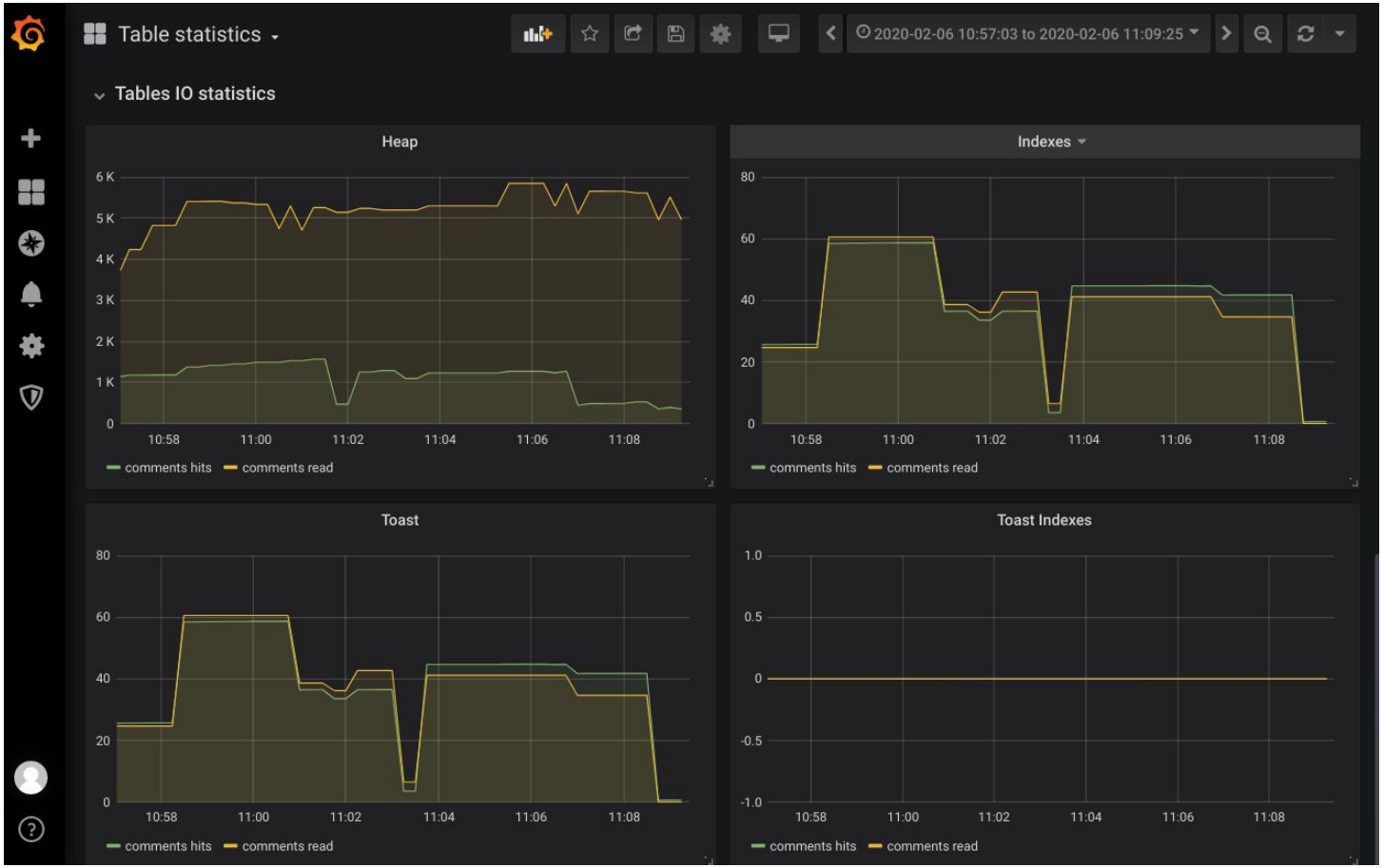
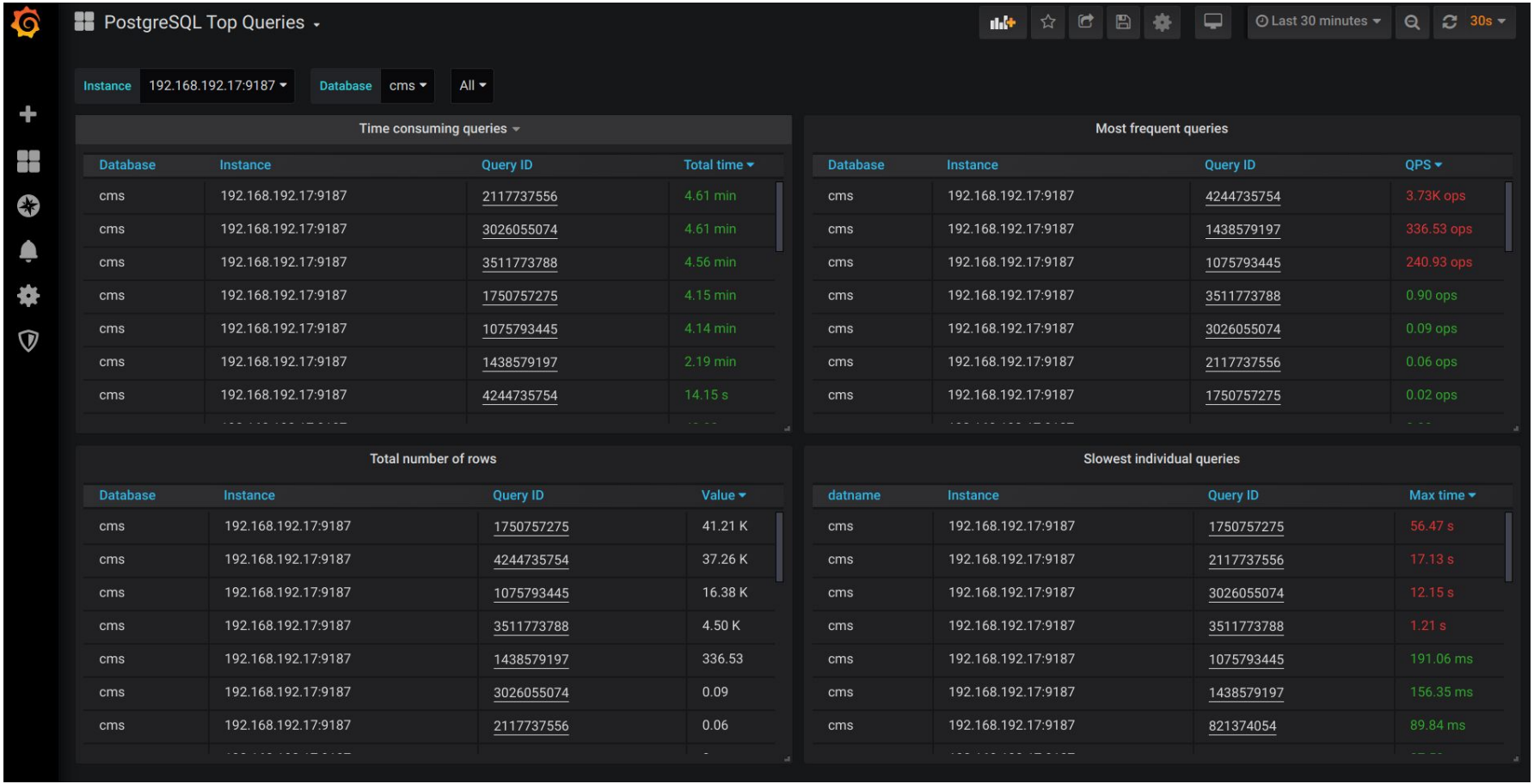
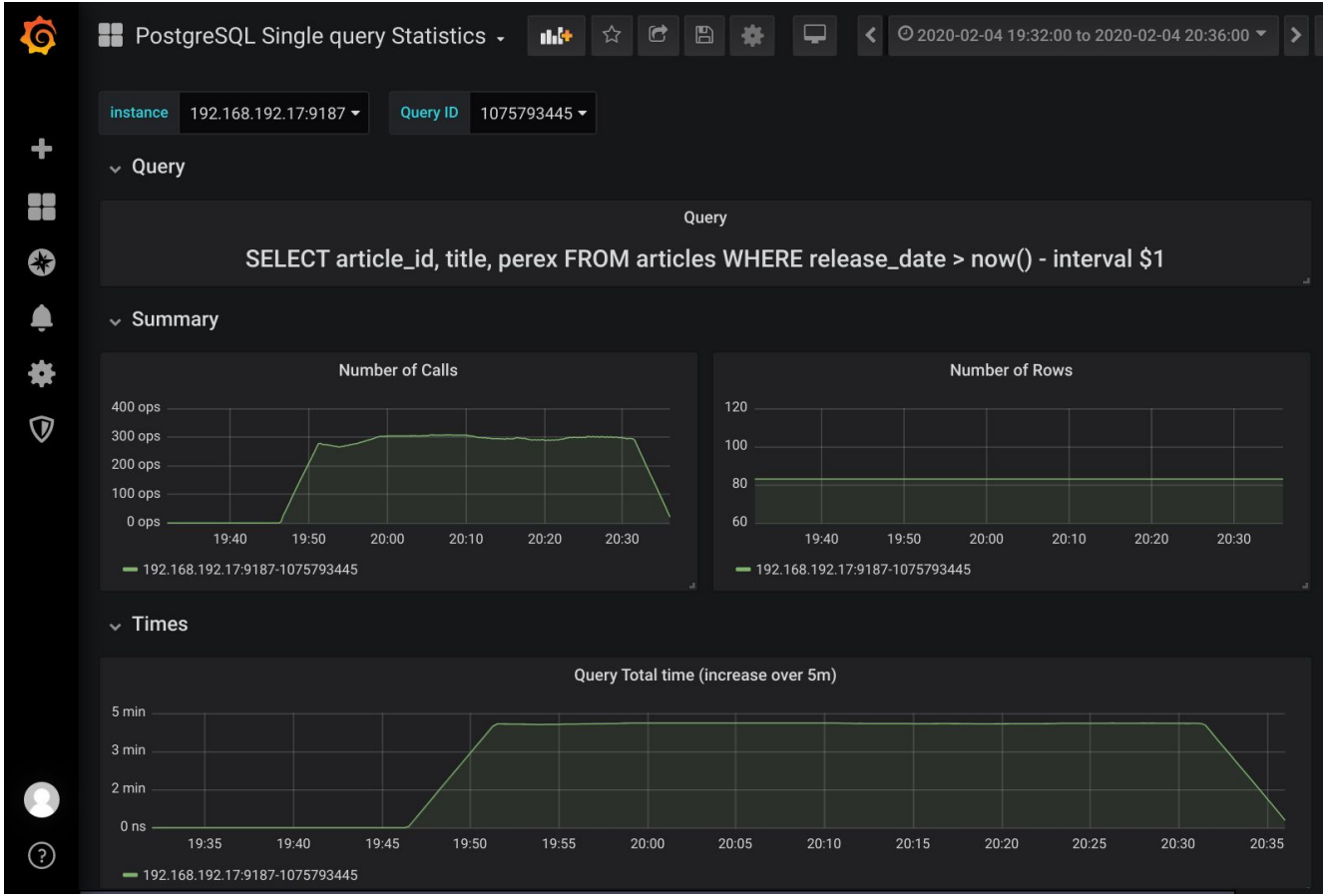
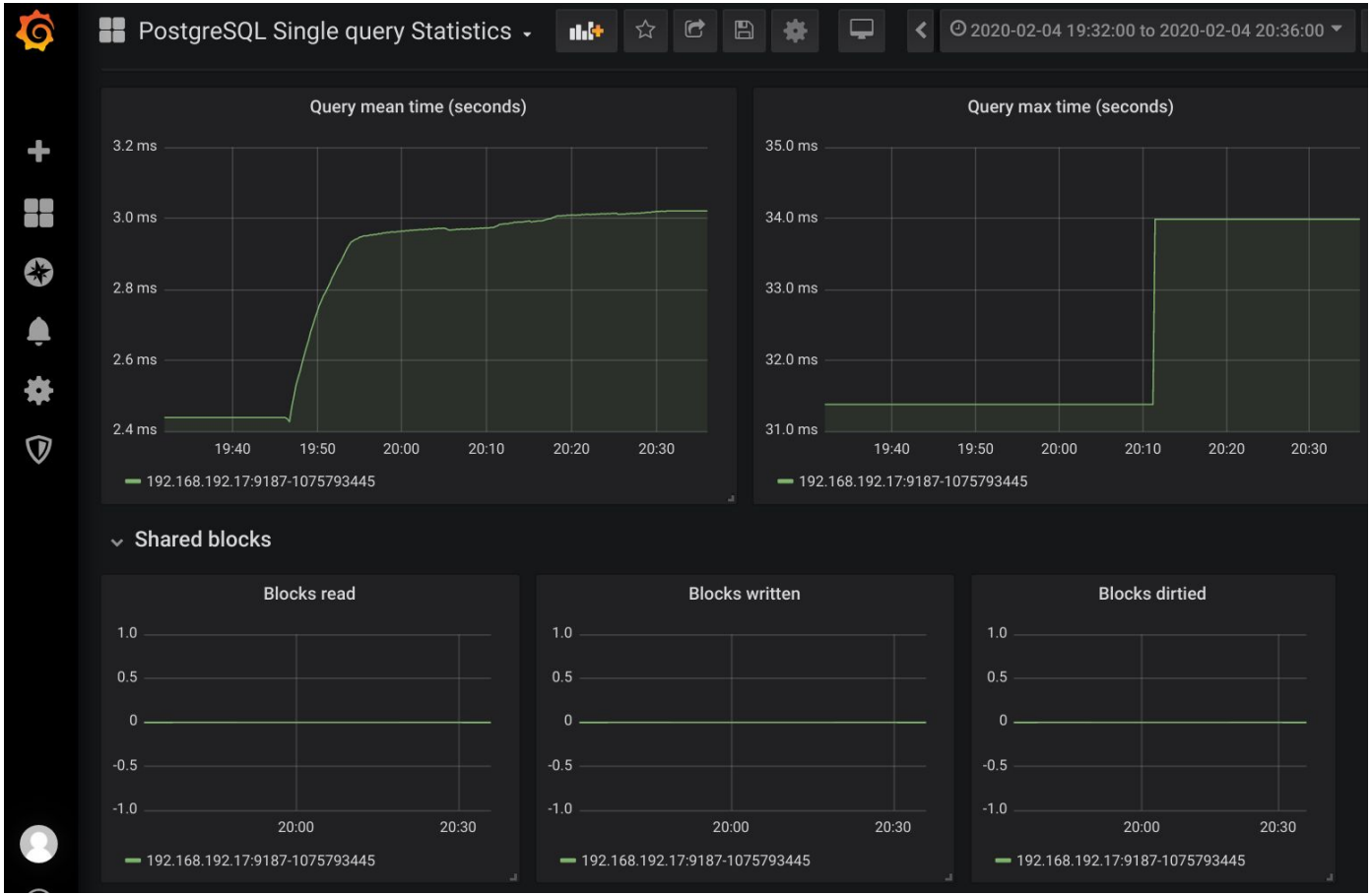
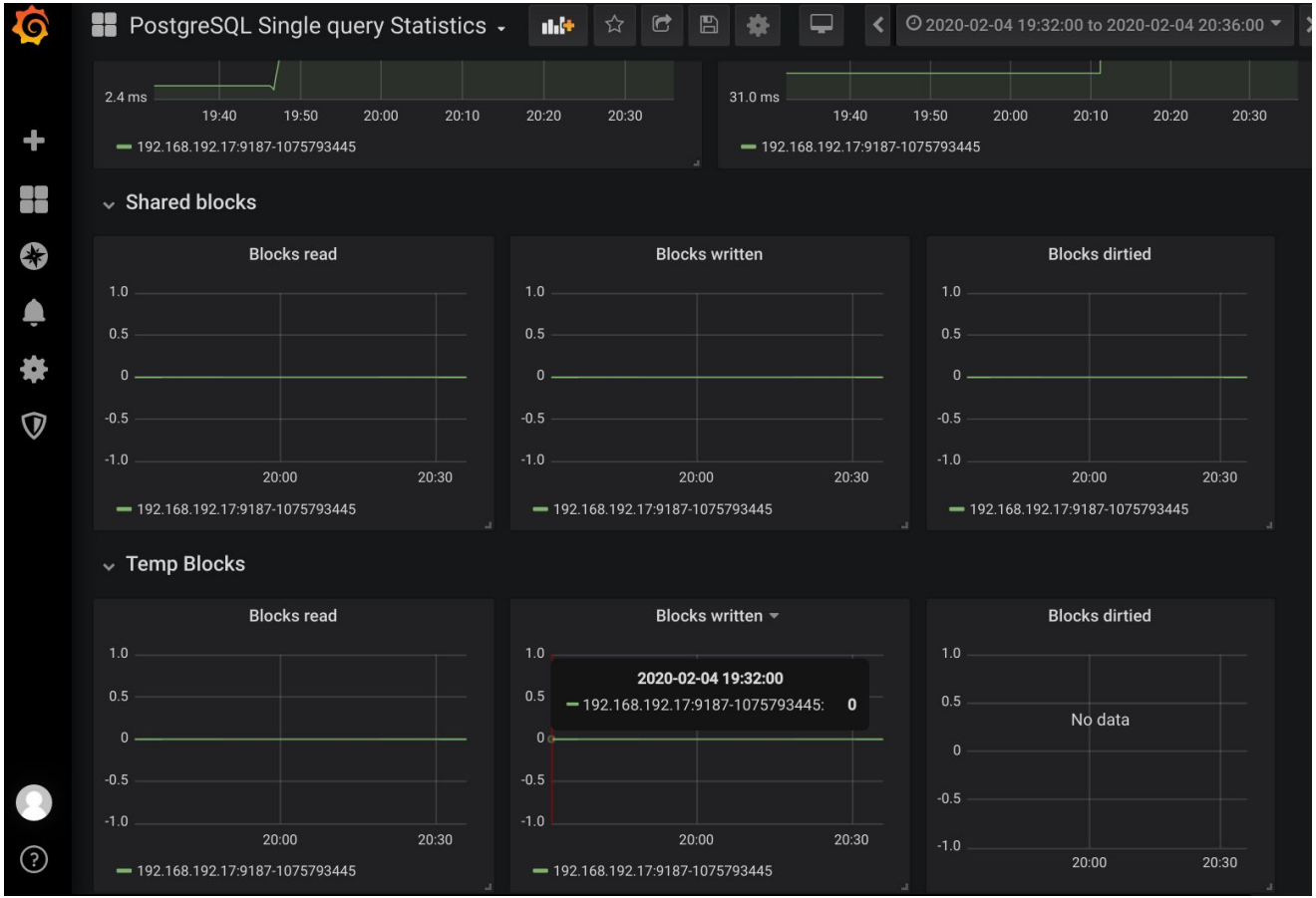


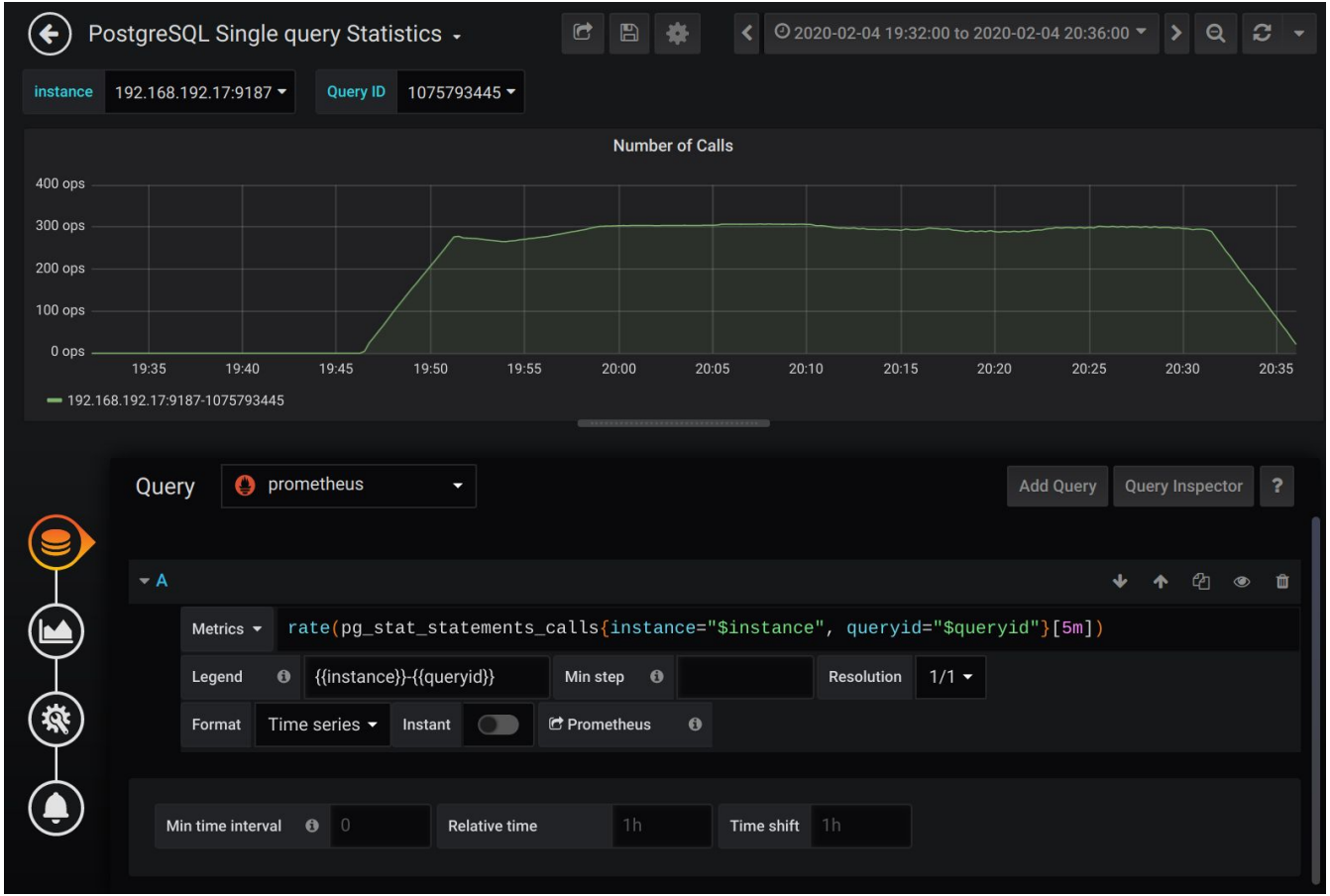
Table statistics
2020-02-06 10:57:03 to 2020-02-06 11:09:25
Tables IO statistics
Heap
6 K
5 K
4 K
3 K
2 K
1 K
0
10:58
11:00
11:02
11:04
11:06
11:08
comments hits
comments read
Indexes
80
60
40
20
0
Toast
Toast Indexes
1.0
0.5
-0.5
-1.0


PostgreSQL Top Queries
Last 30 minutes
30s
Instance 192.168.192.17:9187
Database cms
All
Time consuming queries
Database Instance Query ID Total time
cms 192.168.192.17:9187 2117737556 4.61 min
cms 192.168.192.17:9187 3026055074 4.61 min
cms 192.168.192.17:9187 3511773788 4.56 min
cms 192.168.192.17:9187 1750757275 4.15 min
cms 192.168.192.17:9187 1075793445 4.14 min
cms 192.168.192.17:9187 1438579197 2.19 min
cms 192.168.192.17:9187 4244735754 14.15 s
Most frequent queries
Database Instance Query ID QPS
cms 192.168.192.17:9187 4244735754 3.73K ops
cms 192.168.192.17:9187 1438579197 336.53 ops
cms 192.168.192.17:9187 1075793445 240.93 ops
cms 192.168.192.17:9187 3511773788 0.90 ops
cms 192.168.192.17:9187 3026055074 0.09 ops
cms 192.168.192.17:9187 2117737556 0.06 ops
cms 192.168.192.17:9187 1750757275 0.02 ops
Total number of rows
Database Instance Query ID Value
cms 192.168.192.17:9187 1750757275 41.21 K
cms 192.168.192.17:9187 4244735754 37.26 K
cms 192.168.192.17:9187 1075793445 16.38 K
cms 192.168.192.17:9187 3511773788 4.50 K
cms 192.168.192.17:9187 1438579197 336.53
cms 192.168.192.17:9187 3026055074 0.09
cms 192.168.192.17:9187 2117737556 0.06
Slowest individual queries
datname Instance Query ID Max time
cms 192.168.192.17:9187 1750757275 56.47 s
cms 192.168.192.17:9187 2117737556 17.13 s
cms 192.168.192.17:9187 3026055074 12.15 s
cms 192.168.192.17:9187 3511773788 1.21 s
cms 192.168.192.17:9187 1075793445 191.06 ms
cms 192.168.192.17:9187 1438579197 156.35 ms
cms 192.168.192.17:9187 821374054 89.84 ms


PostgreSQL Single query Statistics
2020-02-04 19:32:00 to 2020-02-04 20:36:00
instance
192.168.192.17:9187
Query ID
1075793445
Query
Query
SELECT article_id, title, perex FROM articles WHERE release_date > now() - interval $1
Summary
Number of Calls
400 ops
300 ops
200 ops
100 ops
0 ops
19:40
19:50
20:00
20:10
20:20
20:30
192.168.192.17:9187-1075793445
Number of Rows
120
100
80
60
19:40
19:50
20:00
20:10
20:20
20:30
192.168.192.17:9187-1075793445
Times
Query Total time (increase over 5m)
5 min
3 min
2 min
0 ns
19:35
19:40
19:45
19:50
19:55
20:00
20:05
20:10
20:15
20:20
20:25
20:30
20:35
192.168.192.17:9187-1075793445


PostgreSQL Single query Statistics
2020-02-04 19:32:00 to 2020-02-04 20:36:00
Query mean time (seconds)
3.2 ms
3.0 ms
2.8 ms
2.6 ms
2.4 ms
19:40
19:50
20:00
20:10
20:20
20:30
192.168.192.17:9187-1075793445
Query max time (seconds)
35.0 ms
34.0 ms
33.0 ms
32.0 ms
31.0 ms
19:40
19:50
20:00
20:10
20:20
20:30
192.168.192.17:9187-1075793445
Shared blocks
Blocks read
1.0
0.5
0
-0.5
-1.0
20:00
20:30
192.168.192.17:9187-1075793445
Blocks written
1.0
0.5
0
-0.5
-1.0
20:00
20:30
192.168.192.17:9187-1075793445
Blocks dirtied
1.0
0.5
0
-0.5
-1.0
20:00
20:30
192.168.192.17:9187-1075793445

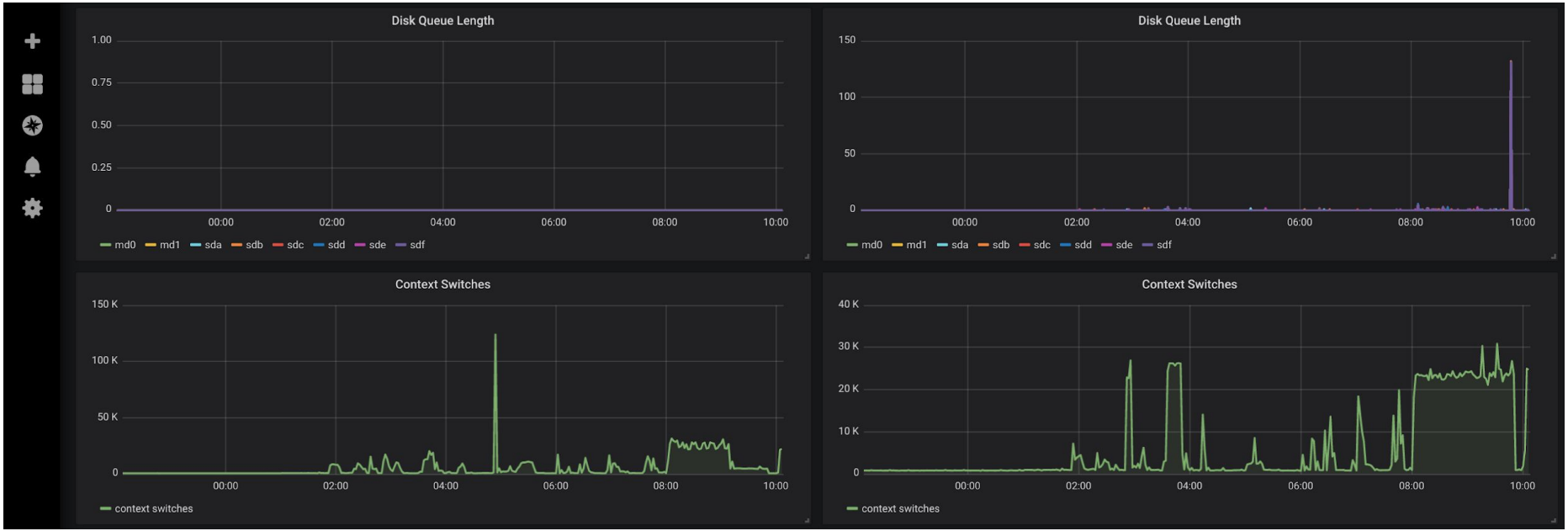

PostgreSQL Single query Statistics
2020-02-04 19:32:00 to 2020-02-04 20:36:00
2.4 ms
19:40
19:50
20:00
20:10
20:20
20:30
192.168.192.17:9187-1075793445
31.0 ms
19:40
19:50
20:00
20:10
20:20
20:30
192.168.192.17:9187-1075793445
Shared blocks
Blocks read
1.0
0.5
0
-0.5
-1.0
20:00
20:30
192.168.192.17:9187-1075793445
Blocks written
1.0
0.5
0
-0.5
-1.0
20:00
20:30
192.168.192.17:9187-1075793445
Blocks dirtied
1.0
0.5
0
-0.5
-1.0
20:00
20:30
192.168.192.17:9187-1075793445
Temp Blocks
Blocks read
1.0
0.5
0
-0.5
-1.0
20:00
20:30
192.168.192.17:9187-1075793445
Blocks written
1.0
0.5
0
-0.5
-1.0
2020-02-04 19:32:00
192.168.192.17:9187-1075793445: 0
20:00
20:30
192.168.192.17:9187-1075793445
Blocks dirtied
1.0
0.5
0
-0.5
-1.0
No data
20:00
20:30


PostgreSQL Single query Statistics
2020-02-04 19:32:00 to 2020-02-04 20:36:00
instance
192.168.192.17:9187
Query ID
1075793445
Number of Calls
400 ops
300 ops
200 ops
100 ops
0 ops
19:35
19:40
19:45
19:50
19:55
20:00
20:05
20:10
20:15
20:20
20:25
20:30
20:35
192.168.192.17:9187-1075793445
Query
prometheus
Add Query
Query Inspector
A
Metrics
rate(pg_stat_statements_calls{instance="$instance", queryid="$queryid"}[5m])
Legend
{{instance}}-{{queryid}}
Min step
Resolution
1/1
Format
Time series
Instant
Prometheus
Min time interval
0
Relative time
1h
Time shift
1h


CPU Cores
16
CPU Cores
16
CPU
100%
75%
50%
25%
0%
%
00:00
02:00
04:00
06:00
08:00
10:00
system
user
nice
iowait
steal
idle
irq
softirq
CPU
100%
75%
50%
25%
0%
%
00:00
02:00
04:00
06:00
08:00
10:00
system
user
nice
iowait
steal
idle
irq
softirq
Memory
279 GiB
233 GiB
186 GiB
140 GiB
93 GiB
47 GiB
0 B
GB
00:00
02:00
04:00
06:00
08:00
10:00
Apps
Buffers
Cached
Free
Slab
SwapCached
PageTables
VmallocUsed
Committed
Mapped
Active
Inactive
Memory
279 GiB
233 GiB
186 GiB
140 GiB
93 GiB
47 GiB
0 B
GB
00:00
02:00
04:00
06:00
08:00
10:00
Apps
Buffers
Cached
Free
Slab
SwapCached
PageTables
VmallocUsed
Committed
Mapped
Active
Inactive


Load
4.0
3.0
2.0
1.0
0
00:00
02:00
04:00
06:00
08:00
10:00
load
Load
2.5
2.0
1.5
1.0
0.5
0
00:00
02:00
04:00
06:00
08:00
10:00
load
Disk Space Used
100%
75%
50%
25%
0%
00:00
02:00
04:00
06:00
08:00
10:00
/
/var/lib/postgresql
Disk Space Used
100%
75%
50%
25%
0%
00:00
02:00
04:00
06:00
08:00
10:00
/
/var/lib/postgresql


CDN / Node Exporter Server Metrics
Last 12 hours UTC
Disk Utilization per Device
100%
75%
50%
25%
0%
00:00
02:00
04:00
06:00
08:00
10:00
md0 md1 sda sdb sdc sdd sde sdf
Disk Utilization per Device
Disk IOs per Device
5 K
0
-5 K
-10 K
IO/second read (-) / write (+)
md0_read md1_read sda_read sdb_read sdc_read sdd_read sde_read sdf_read md0_write md1_write
sda_write sdb_write sdc_write sdd_write sde_write sdf_write
Disk IOs per Device
30 K
20 K
10 K
Disk Throughput per Device
954 MiB
0 B
-954 MiB
-1.9 GiB
477 MiB
381 MiB
286 MiB
191 MiB
95 MiB


Disk Queue Length
1.00
0.75
0.50
0.25
0
00:00
02:00
04:00
06:00
08:00
10:00
md0 md1 sda sdb sdc sdd sde sdf
Disk Queue Length
150
100
50
0
00:00
02:00
04:00
06:00
08:00
10:00
md0 md1 sda sdb sdc sdd sde sdf
Context Switches
150 K
100 K
50 K
0
00:00
02:00
04:00
06:00
08:00
10:00
context switches
Context Switches
40 K
30 K
20 K
10 K
0
00:00
02:00
04:00
06:00
08:00
10:00
context switches


Network Traffic
954 Mib
715 Mib
477 Mib
238 Mib
0 b
-238 Mib
bits in (-) / bits out (+)
2020-02-05 04:44:00
eth0_in: 42 Kib
lo_in: 67 Kib
eth0_out: 57 Kib
lo_out: 67 Kib
00:00
02:00
04:00
06:00
08:00
10:00
eth0_in
lo_in
eth0_out
lo_out
Network Traffic
238 Mib
0 b
-238 Mib
-477 Mib
-715 Mib
-954 Mib
bits in (-) / bits out (+)
Dropped packets / pause frames
1.0
0.5
0
-0.5
-1.0
packets per second
No data points
Dropped packets / pause frames
packets per second
No data points

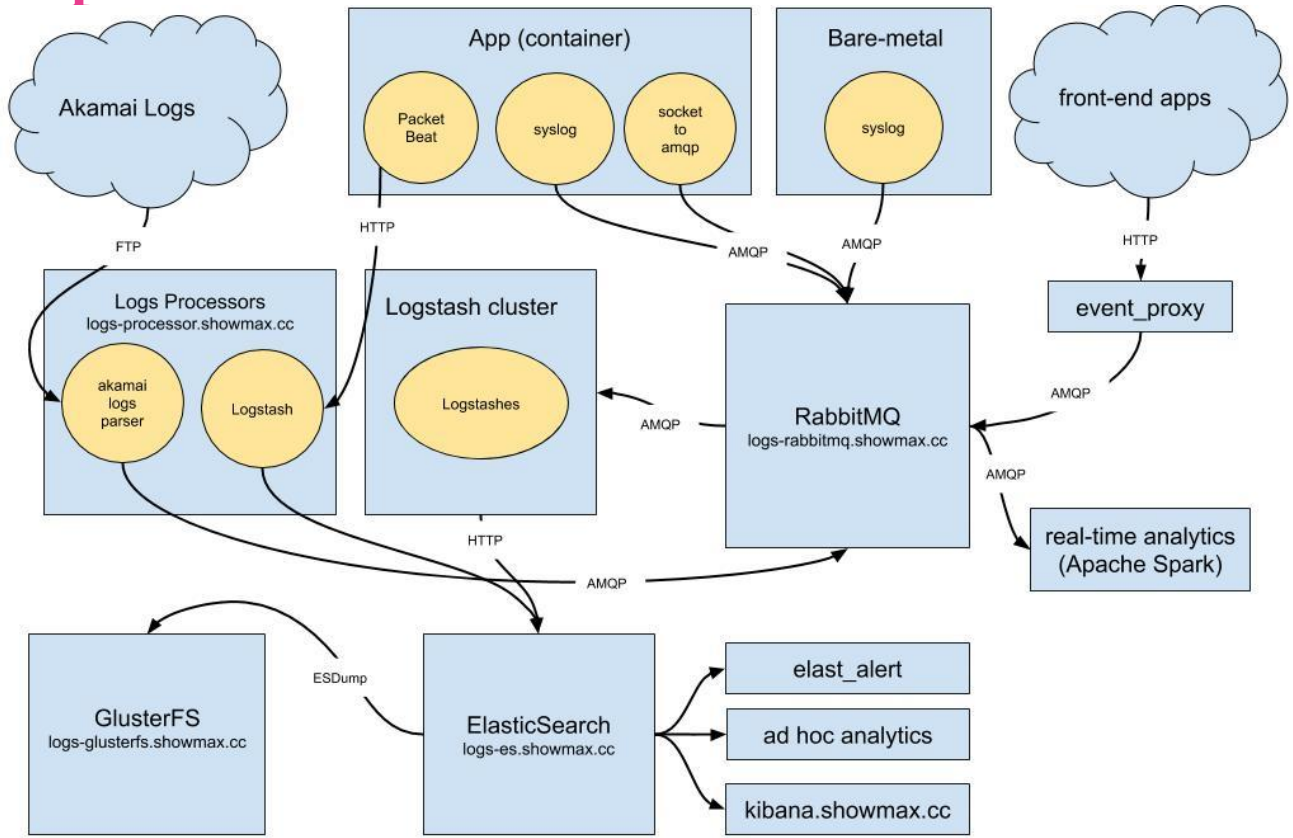
Prometheus caveats

- Beware of labels with high cardinality
  - Significant performance penalty
  - It is not possible to remove labels from DB

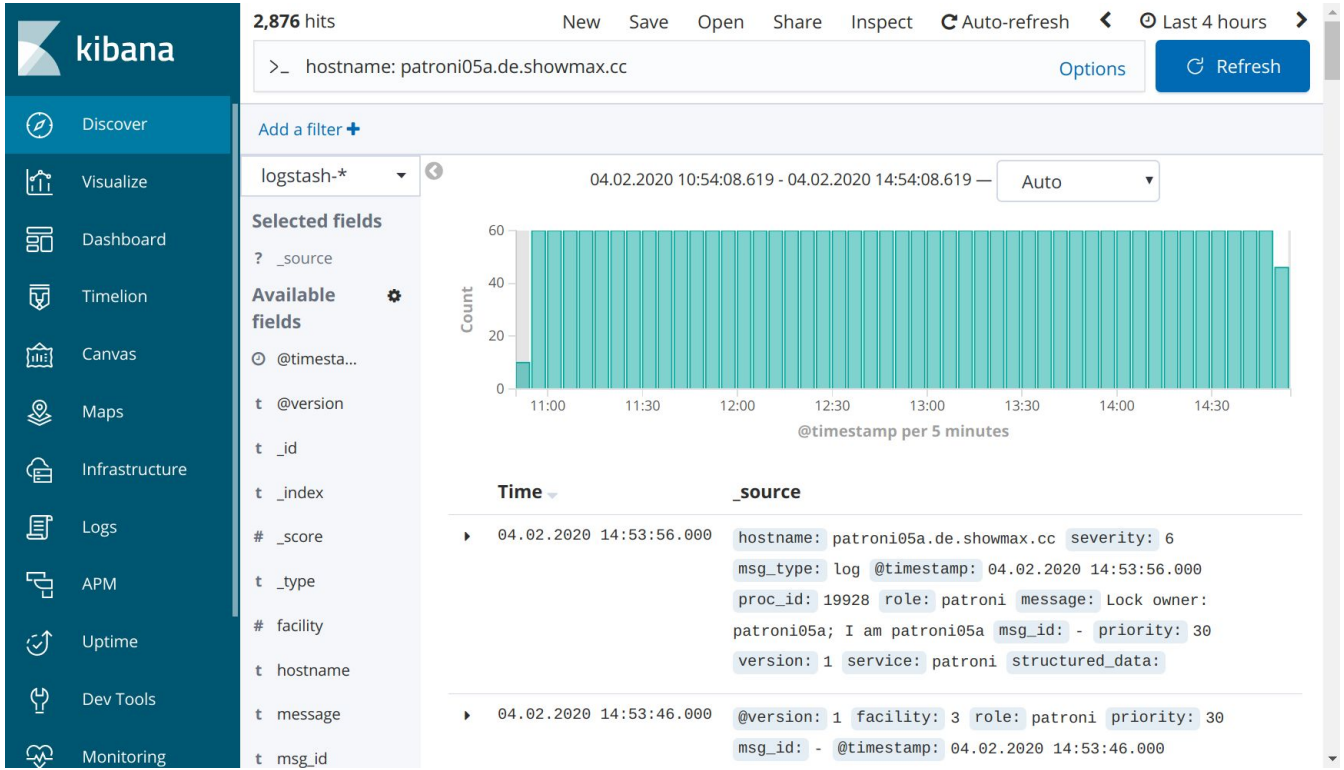
Tools - ELK Stack

- Elastic
  - Search and analytics engine
- RabbitMQ
  - OSS Message broker. Used for log messages delivery
- Logstash
  - Server-side data processing pipeline
- Kibana
  - Kibana lets users visualize data with charts and graphs in Elasticsearch.

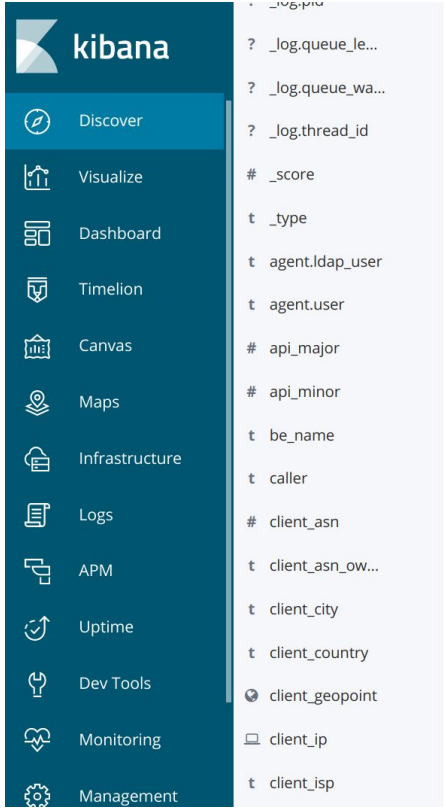
Logging Pipeline



Kibana


kibana
Discover
Visualize
Dashboard
Timelion
Canvas
Maps
Infrastructure
Logs
APM
Uptime
Dev Tools
Monitoring
2,876 hits
New
Save
Open
Share
Inspect
Auto-refresh
Last 4 hours
>_ hostname: patroni05a.de.showmax.cc
Options
Refresh
Add a filter +
logstash-*
Selected fields
? _source
Available fields
@timesta...
t @version
t _id
t _index
_score
t _type
facility
t hostname
t message
t msg_id
04.02.2020 10:54:08.619 - 04.02.2020 14:54:08.619 —
Auto
Count
60
40
20
0
11:00
11:30
12:00
12:30
13:00
13:30
14:00
14:30
@timestamp per 5 minutes
Time
_source
04.02.2020 14:53:56.000
hostname: patroni05a.de.showmax.cc severity: 6 msg_type: log @timestamp: 04.02.2020 14:53:56.000 proc_id: 19928 role: patroni message: Lock owner: patroni05a; I am patroni05a msg_id: - priority: 30 version: 1 service: patroni structured_data:
04.02.2020 14:53:46.000
@version: 1 facility: 3 role: patroni priority: 30 msg_id: - @timestamp: 04.02.2020 14:53:46.000

Kibana




```
available FROM etl.etl_log WHERE job_id = 'daily_run_setup' AND daily_run_id =
(SELECT MAX(daily_run_id) FROM etl.etl_log) AND job_status = 'started';
data_warehouse_worker.thread: #<Thread:0x0000564309d241a8@worker.rb:33 run>
```

Table | JSON | Call Tracer | View surrounding documents | View single document

t	@timestamp	05.02.2020 09:56:47.201
t	@version	1
t	_id	UCrIFHABJWHn0EJ0BrdY
t	_index	logstash-2020.02.05-000135
?	_log.outputter	amqp
?	_log.pid	399
?	_log.queue_length	0
?	_log.queue_wait_ms	0.113
?	_log.thread_id	47422963785940
#	_score	-
t	_type	fluentd
t	data_warehouse_worker.db	dwh
#	data_warehouse_worker.pid	18,767
t	data_warehouse_worker.query	SELECT 1 as available FROM etl.etl_log WHERE job_id = 'daily_run_setup' AND daily_run_id = (SELECT MAX(daily_run_id) FROM etl.etl_log) AND job_status = 'started';

Elastalert

- Simple framework to alert anomalies, spikes, or other patterns of interest from data in Elasticsearch
- Two types of components:
  - Rule types (frequency, spike, flatline, etc.)
  - Alert types (email, slack, OpsGenie, etc.)
- Alerts can include:
  - Link to Kibana dashboards
  - Aggregate counts for arbitrary fields
  - Combine alerts into periodic reports
  - Intercept and enhance match data

Elastalert

```
name: OPS Kernel OOM

# Rule description
description: >-
  Rule monitors Kernel Out of Memory issues / OOM killer triggers. If experiencing OOM
  the machine is short on free memory. Check the provided Kibana link in depth, find killed
  process(es) that required too much memory (thus it has to be killed) and fix the situation
  accordingly.

# (Required)
# Type of alert.
# the frequency rule type alerts when num_events events occur with timeframe time
type: frequency

# (Required)
# Index to search, wildcard supported
index: syslog-*

use_kibana4_dashboard: "https://kibana.showmax.cc/app/kibana#/discover/28305370-6b9e-11e7-a7ee-fd0eb2c06785"

# (Required, frequency specific)
# Alert when this many documents matching the query occur within a timeframe
num_events: 1

# (Required, frequency specific)
# num_events must occur within this amount of time to trigger an alert
timeframe:
  minutes: 15

# (Required)
# A list of elasticsearch filters used for find events
# These filters are joined with AND and nested in a filtered query
# For more info: http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/query-dsl.html
filter:
- term:
    service: "kernel"
- query:
    query_string:
      query: "message: \"Out of memory\""
```

OPS Kernel OOM

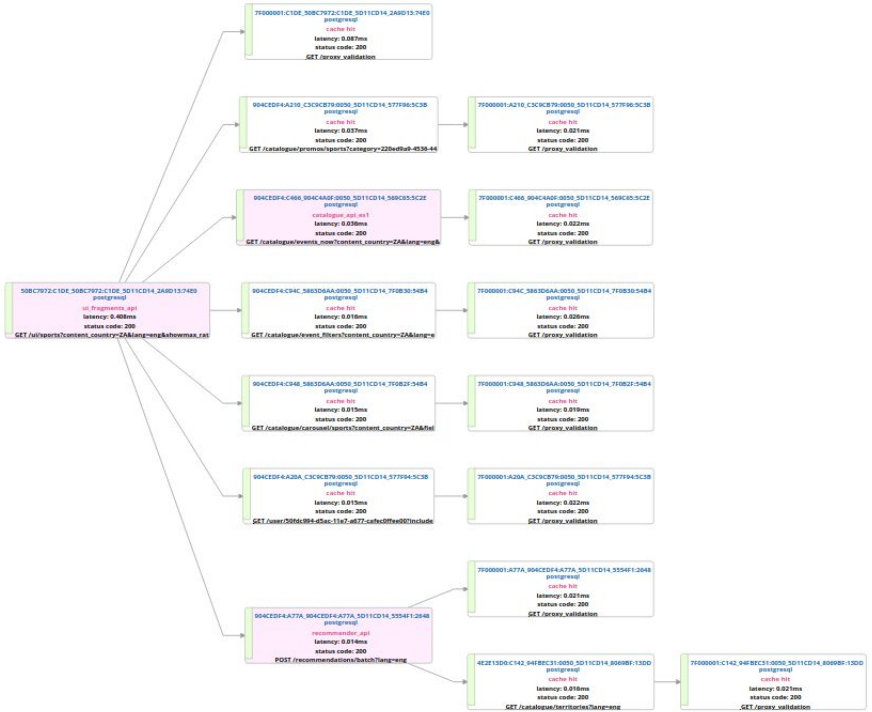
At least 1 events occurred between 2019-06-05 07:23 UTC and 2019-06-05 07:38 UTC

@timestamp: 2019-06-05T07:38:31Z
@version: 1
id: RCWTJmsBCc8JNv71Q6o
_index: syslog-2019.06.05
_type: syslog
facility: 0
hostname: nomad01we.de.showmax.io
kibana_link: https://kibana.showmax.cc/app/kibana#/discover/28305370-6b9e-11e7-a7ee-fd0eb2c06785?_g=%28refreshInterval%3A%28display%3AOff%2Csection%3A0%2Cvalue%3A0%29%2Ctime%3A%28from%3A%272019-06-05T07%3A23%3A31Z%27%2Cmode%3Aabsolute%2Cto%3A%272019-06-05T07%3A53%3A31Z%27%29%29
message: Memory cgroup out of memory: Kill process 18274 (ruby) score 837 or sacrifice child

msg_id: -
msg_type: syslog
num_hits: 1
num_matches: 1
priority: 3
proc_id: -
service: kernel

Tracing the API requests

- Trace request across services
  - Down to DB statements



Going down to the DB statements

- Comments to the rescue
  - SELECT "user_profiles".* FROM "user_profiles" WHERE "user_profiles"."user_id" = '6881a8eb-5e54-4073-a2ec-a62eb4e8e746' /* ***74CA2798:B364_904C6C7E:0050_5E1D733D_12189B7:0428 */***
- Instrument the ORM layers to include the tracing information
  - Active Record for Ruby
  - adapter.prepend(::ActiveRecord::Tags::ExecuteWithTags)
- ELK stack to trace the sql requests

Why: #1 - Watchdog for the excellence

- Regularly analyze the queries, identify the weak points

Database queries taking too long based on logs-app-postgresql-2020.01.02-*
See https://kibana.showmax.cc/app/kibana#/discover/c5869b80-fe1d-11e8-a107-856e4e008c55
backend@showmax.com
download
002 1,605: DELETE FROM "download_events" WHERE "download_events"."download_id" = '<param>'
001 1,289: SELECT MAX("downloads"."updated_at") FROM "downloads" WHERE "downloads"."user_id" = '<param>'
001 1,209: SELECT "downloads".* FROM "downloads" WHERE ("downloads"."state" != '<param>') AND "downloads"."master_user_id" = '<param>'

Why: #2 - Easy investigation of failures

- Kibana queries based on request ID are completely trackable
- Easy to analyze abnormal patterns, and track back to user actions

Why: #3 - Auditing

- Find who messed with given data?
  - Additionally to classical audit log, we are able to track the API operations down to SQL statements.

Postgres @ Showmax - monitoring

- Icinga
  - Active checks - db connection test
  - HAProxy backends state monitoring
- Prometheus
  - Clusters stats ingestion and alerting
    - Master switching
    - Replica lags
    - Current connection count

Next steps

- Monitoring based on trends
  - Create usual traffic envelope using recording rules
  - Alert on anomalies in the traffic
- Loki
  - “Prometheus for logs” from Grafana
  - Tightly integrated with performance data
- Thanos
  - HA, Long term storage, Downsampling
  - Single interface to all Prometheus instances

Come and join us!

We’re looking for new colleagues

tech.showmax.com

Thanks!

Questions?