Bleemeo | Prometheus

WEBINAR

INTRODUCTION A PROMETHEUS

17 JUIN

11 HEURE

Lionel Porcheron
CEO & co-fondateur de Bleemeo
www.bleemeo.com

Bleemeo?

Observability & Monitoring as a service solution

Start monitoring your infrastructure in 30s

Prometheus, Graphite, Statsd, compatible

2 Open Source projects (https://github.com/bleemeo):

- **Glouton**, universal monitoring agent written in Go with Prometheus, Statsd, Graphite, Nagios compatibility
- **SquirrelDB**, a scalable Prometheus compatible storage backend based on Cassandra

Who am I?

Lionel Porcheron, CEO & co-founder Bleemeo

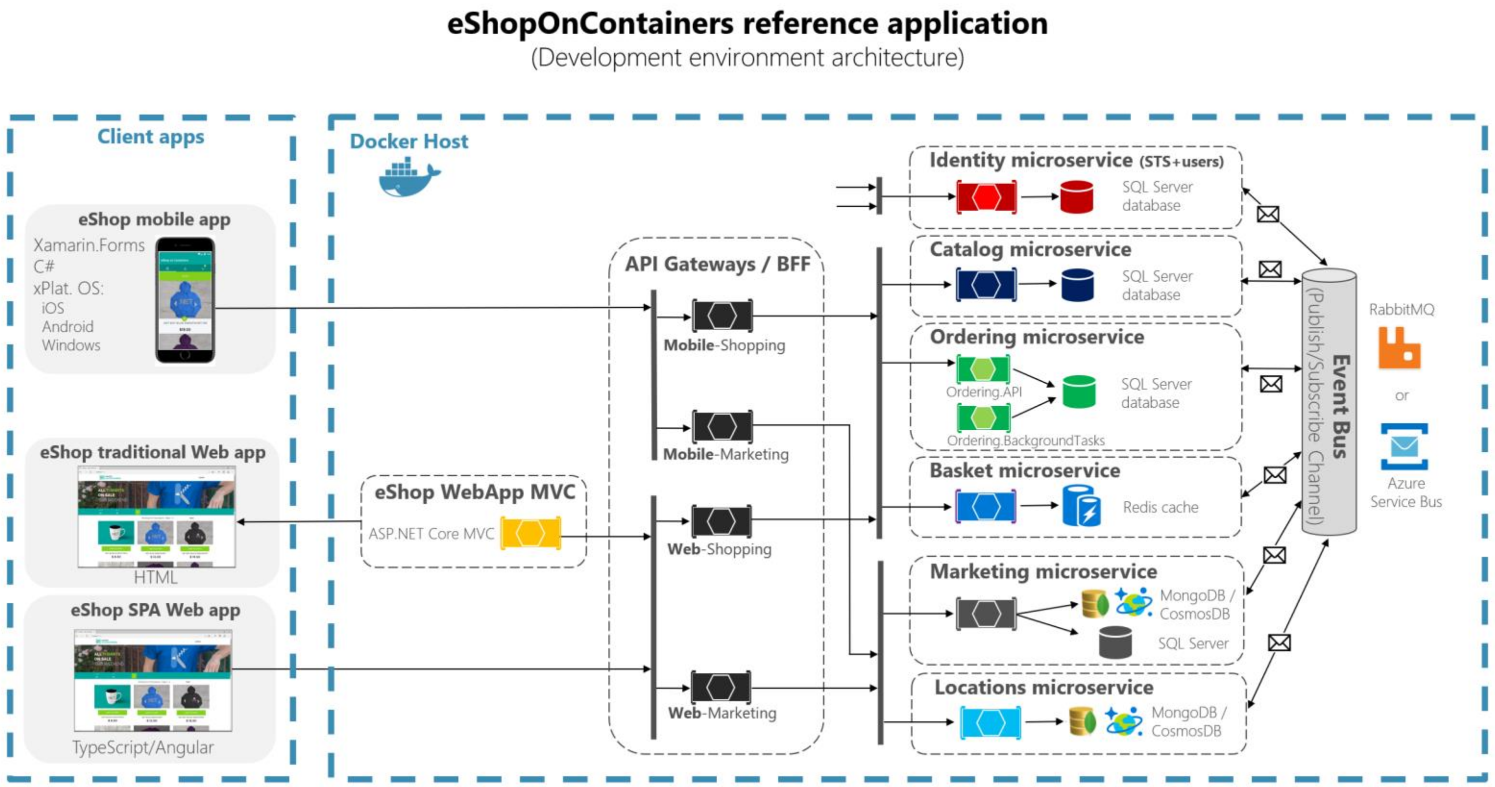
- DevOps for +15 years (started my monitoring journey with ~~nagios~~ netsaint)
- Toulouse DevOps Meetup Leader, Capitole du Libre Leader, PyconFR 2017 Organizer

Old Monitoring Days

- Monitor your server as a blackbox
- Only monitor server & services (web server, database)
- Only availability, not metrics
- Nagios & derivates

Microservices and modern era

- Increase architecture complexity
- Increase number of technical components to monitor
- Moderns infrastructure base on containers are dynamic
- Some components may come from third parties



Graphite... and Prometheus

- Graphite change the way we were doing monitoring: metrics became central
- Graphite appeared in 2008
- Prometheus became de-facto standard for monitoring
- Prometheus was "initiated" in 2012 at Soundcloud and is now a (graduated) CNCF project
- Ecosystem based on Prometheus: exporters, Grafana, software themselves (Kubernetes, Traefik & many others)

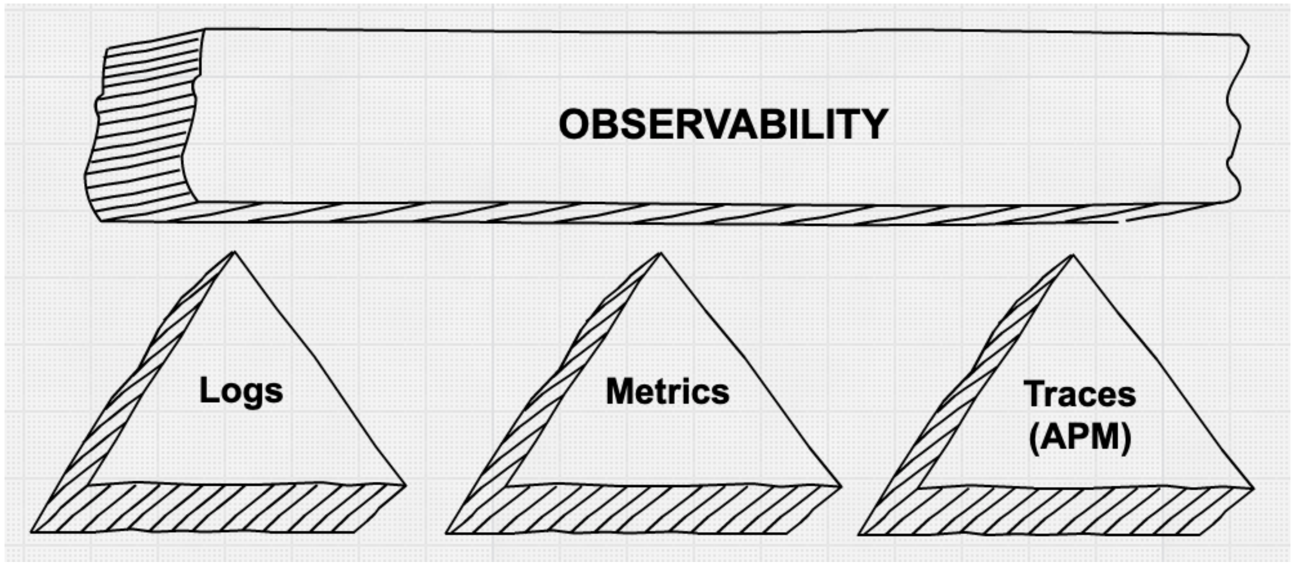
graphite

Prometheus

s/monitoring/observability/

- No more monitoring as blackbox: we now know what is inside
- Exports tons of metrics for future usage
- Code need to be instrumented to provide business metrics
- New Buzzzword 😎

Three pillars of observability



Observability Key Metrics

The RED Method

- (Request) **R**ate - the number of requests, per second, you services are serving.
- (Request) **E**rrors - the number of failed requests per second.
- (Request) **D**uration - distributions of the amount of time each request takes.

The USE Method

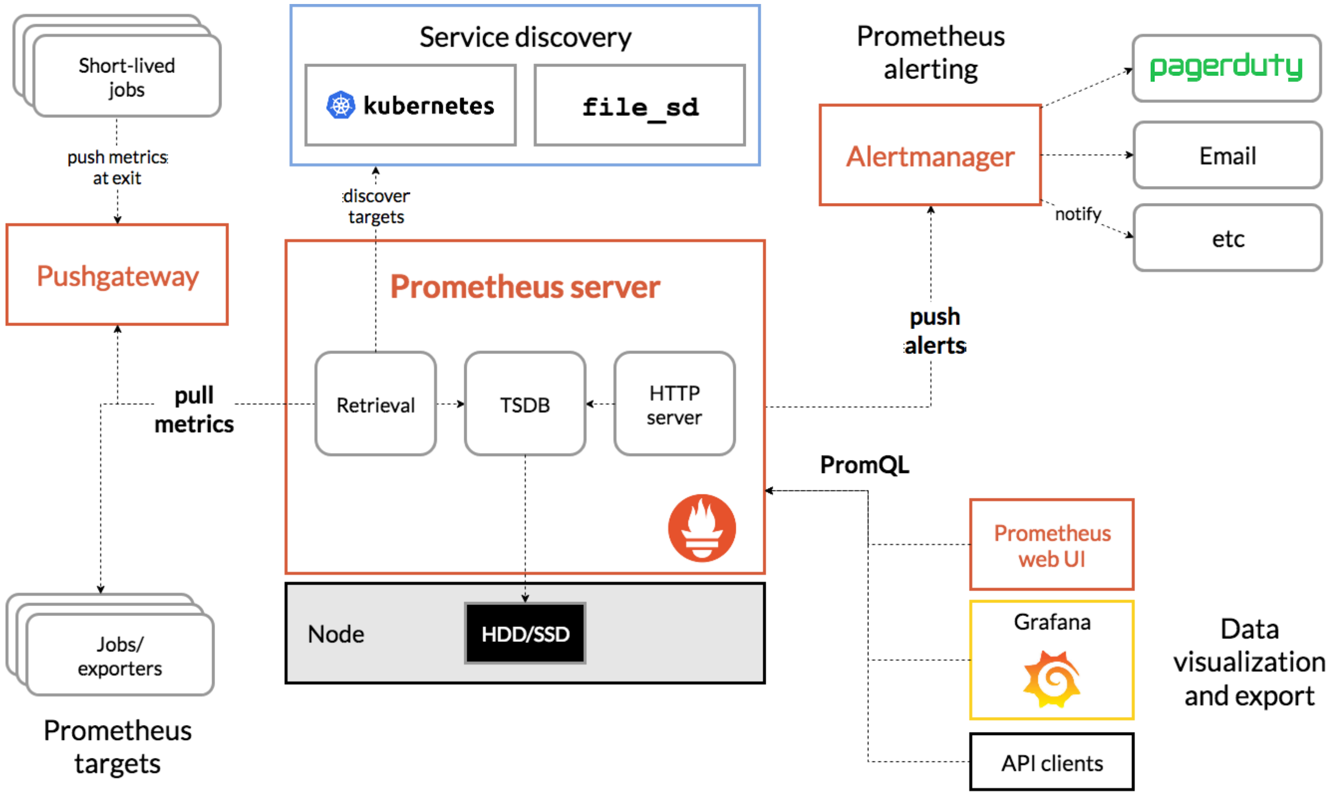
- (Ressource) **U**tilization: as a percent over a time interval. eg, "one disk is running at 90% utilization".
- (Ressource) **S**aturation: as a queue length. eg, "the CPUs have an average run queue length of four".
- (Ressource) **E**rrors: scalar counts. eg, "this network interface has had fifty late collisions".

Prometheus Overview

- A Time Series Database where data is identified by metric name and labels (key/value pairs)
- A powerfull PromQL query language
- No complex storage: designed to store multiple days (not weeks) of data
- Data are collected via a pull over HTTP

Prometheus Architecture

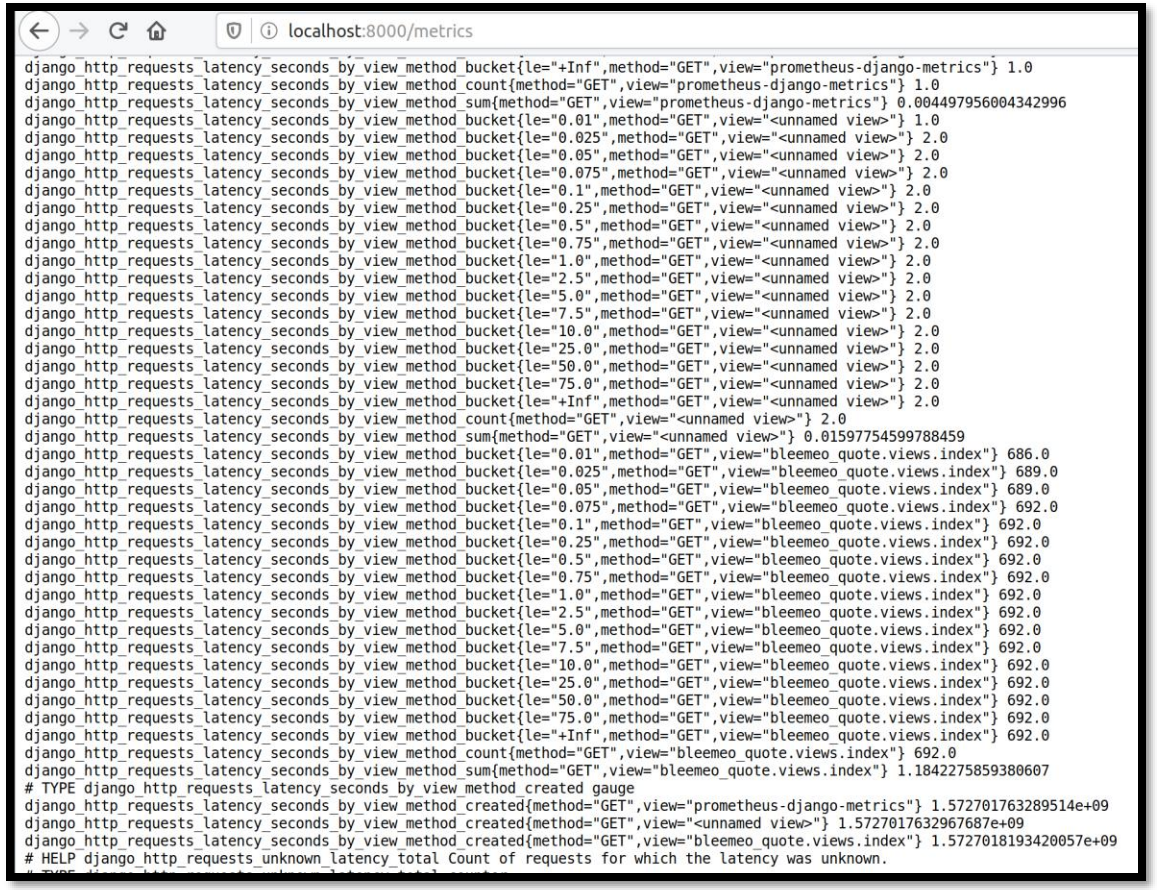
- Prometheus server pull metrics
- Can be integrated with service discovery
- Pushgateway allow to push metrics
- Alertmanager send alarms trigerred by Prometheus Server
- PromQL to query
- Long term storage are external projects



Prometheus exporters

- Prometheus exporters export on web page metrics (basic plain text page with a metric per line)
- Prometheus poll regularly those endpoints
- Some projects embed a Prometheus endpoint (gitlab, traefik, …)
- Prometheus exporters exist for almost everything (244 projects listed today):

  https://prometheus.io/docs/instrumenting/exporters/

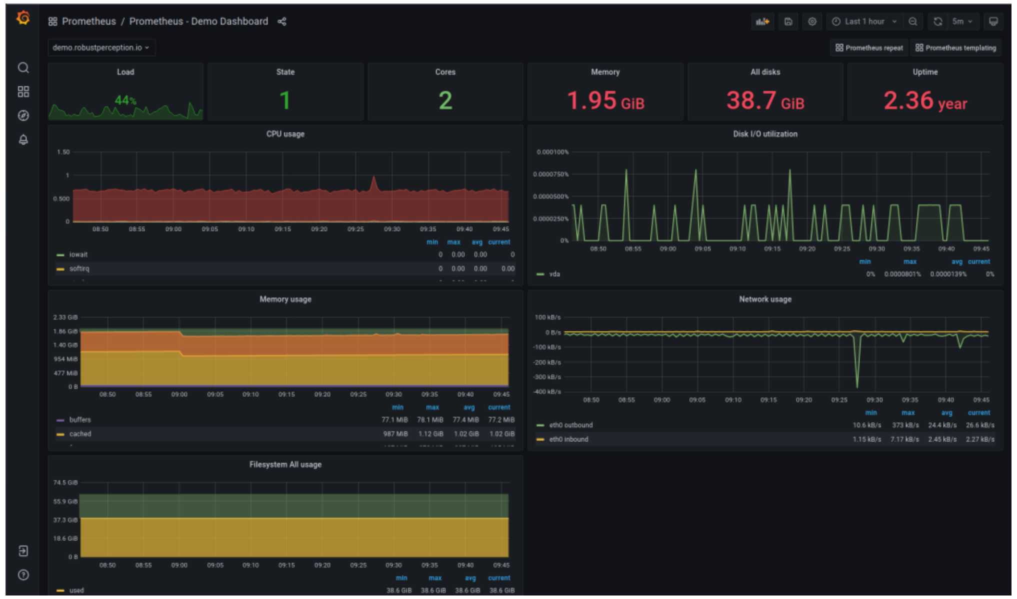


localhost:8000/metrics

```
django_http_requests_latency_seconds_by_view_method_bucket{le="+Inf",method="GET",view="prometheus-django-metrics"} 1.0
django_http_requests_latency_seconds_by_view_method_count{method="GET",view="prometheus-django-metrics"} 1.0
django_http_requests_latency_seconds_by_view_method_sum{method="GET",view="prometheus-django-metrics"} 0.004497956004342996
django_http_requests_latency_seconds_by_view_method_bucket{le="0.01",method="GET",view="<unnamed view>"} 1.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.025",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.05",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.075",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.1",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.25",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.5",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.75",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="1.0",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="2.5",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="5.0",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="7.5",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="10.0",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="25.0",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="50.0",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="75.0",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_bucket{le="+Inf",method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_count{method="GET",view="<unnamed view>"} 2.0
django_http_requests_latency_seconds_by_view_method_sum{method="GET",view="<unnamed view>"} 0.01597754599788459
django_http_requests_latency_seconds_by_view_method_bucket{le="0.01",method="GET",view="bleemeo_quote.views.index"} 686.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.025",method="GET",view="bleemeo_quote.views.index"} 689.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.05",method="GET",view="bleemeo_quote.views.index"} 689.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.075",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.1",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.25",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.5",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_bucket{le="0.75",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_bucket{le="1.0",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_bucket{le="2.5",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_bucket{le="5.0",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_bucket{le="7.5",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_bucket{le="10.0",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_bucket{le="25.0",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_bucket{le="50.0",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_bucket{le="75.0",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_bucket{le="+Inf",method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_count{method="GET",view="bleemeo_quote.views.index"} 692.0
django_http_requests_latency_seconds_by_view_method_sum{method="GET",view="bleemeo_quote.views.index"} 1.1842275859380607
# TYPE django_http_requests_latency_seconds_by_view_method_created gauge
django_http_requests_latency_seconds_by_view_method_created{method="GET",view="prometheus-django-metrics"} 1.572701763289514e+09
django_http_requests_latency_seconds_by_view_method_created{method="GET",view="<unnamed view>"} 1.5727017632967687e+09
django_http_requests_latency_seconds_by_view_method_created{method="GET",view="bleemeo_quote.views.index"} 1.5727018193420057e+09
# HELP django_http_requests_unknown_latency_total Count of requests for which the latency was unknown.
```

Prometheus ecosystem

- We already covered exporters
- A lot of projects are now natively proposing a /metrics Prometheus endpoint
- Long term storage: Thanos, Cortex
- Lot of libraries in various languages to instrument your code
- Most famous project in ecosystem is Grafana



Prometheus Query Language

- PromQL (Prometheus Query Language) allow to query metrics
- Can query one or multiple metrics
- Can apply operators and functions on metrics

Return the per-second rate for all time series with the http_requests_total metric name, as measured over the last 5 minutes:

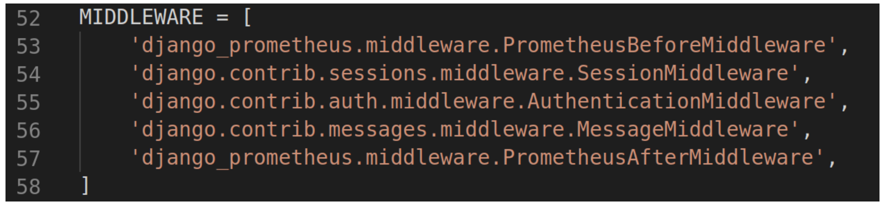
```
rate(http_requests_total[5m])
```

CPU time with labels to select your metric:

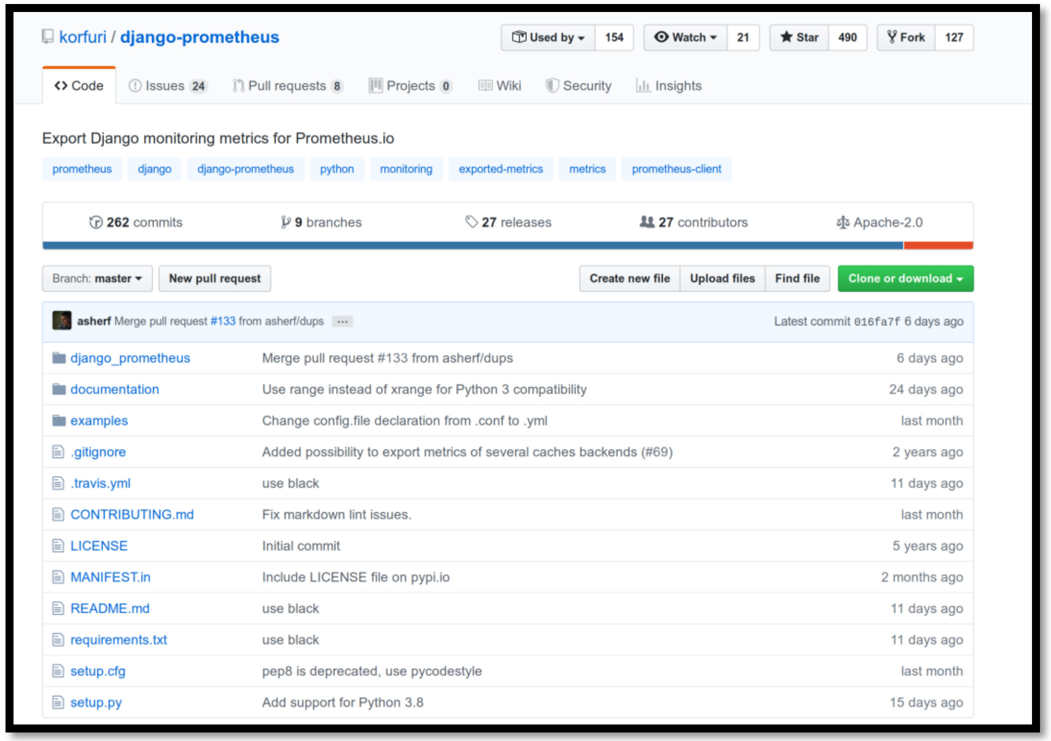
```
instance_cpu_time_ns{app="lion", proc="web", rev="34d0f99",
env="prod", job="cluster-manager"}
```

Instrument your code

- Example of Django web framework
- For Django application: django-prometheus
- Django Middleware for metrics



```python
MIDDLEWARE = [
    'django_prometheus.middleware.PrometheusBeforeMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django_prometheus.middleware.PrometheusAfterMiddleware',
]
```



korfuri / django-prometheus

Used by 154 · Watch 21 · Star 490 · Fork 127

Code · Issues 24 · Pull requests 8 · Projects 0 · Wiki · Security · Insights

Export Django monitoring metrics for Prometheus.io

prometheus · django · django-prometheus · python · monitoring · exported-metrics · metrics · prometheus-client

262 commits · 9 branches · 27 releases · 27 contributors · Apache-2.0

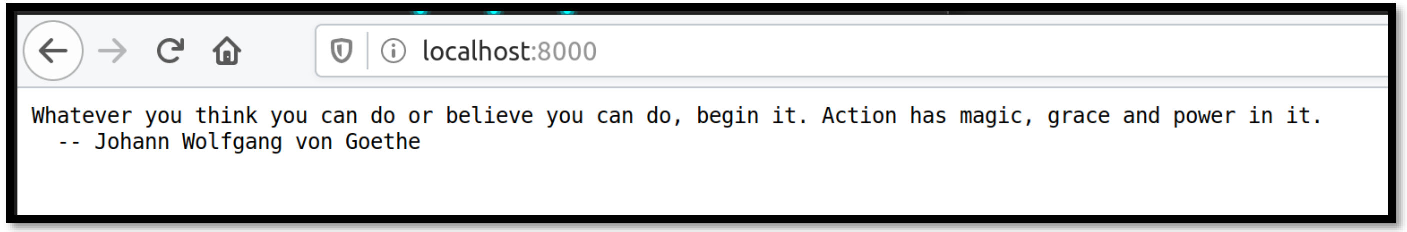
Branch: master · New pull request · Create new file · Upload files · Find file · Clone or download

asherf Merge pull request #133 from asherf/dups ... Latest commit 016fa7f 6 days ago

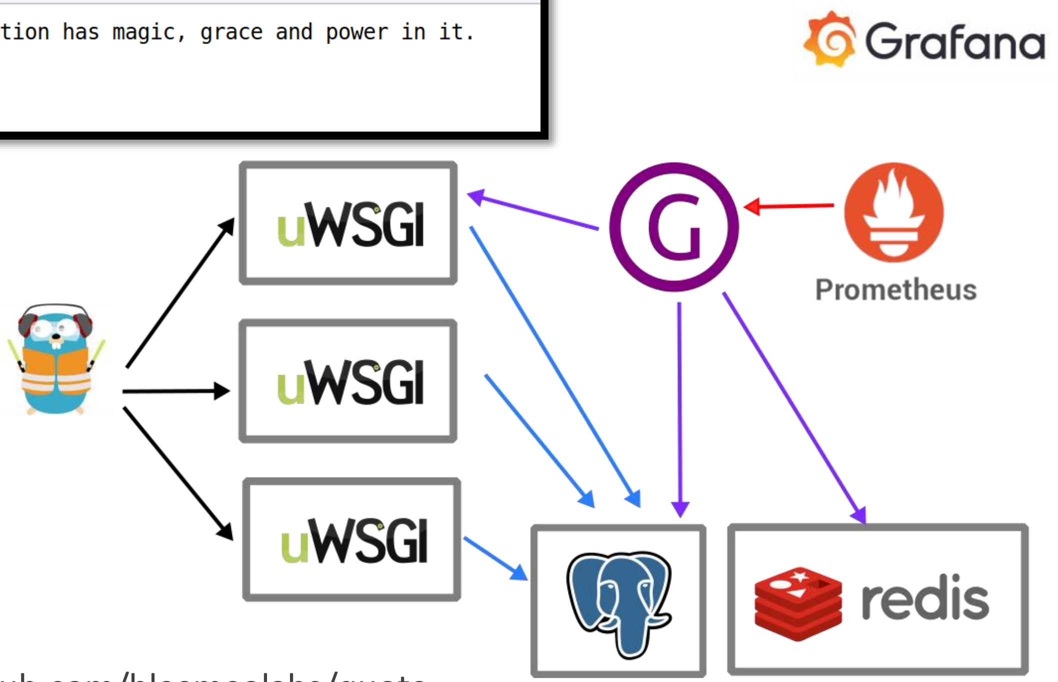
django_prometheus	Merge pull request #133 from asherf/dups	6 days ago
documentation	Use range instead of xrange for Python 3 compatibility	24 days ago
examples	Change config.file declaration from .conf to .yml	last month
.gitignore	Added possibility to export metrics of several caches backends (#69)	2 years ago
.travis.yml	use black	11 days ago
CONTRIBUTING.md	Fix markdown lint issues.	last month
LICENSE	Initial commit	5 years ago
MANIFEST.in	Include LICENSE file on pypi.io	2 months ago
README.md	use black	11 days ago
requirements.txt	use black	11 days ago
setup.cfg	pep8 is deprecated, use pycodestyle	last month
setup.py	Add support for Python 3.8	15 days ago

Example

Small Django application showing quote of the day


localhost:8000

```
Whatever you think you can do or believe you can do, begin it. Action has magic, grace and power in it.
  -- Johann Wolfgang von Goethe
```



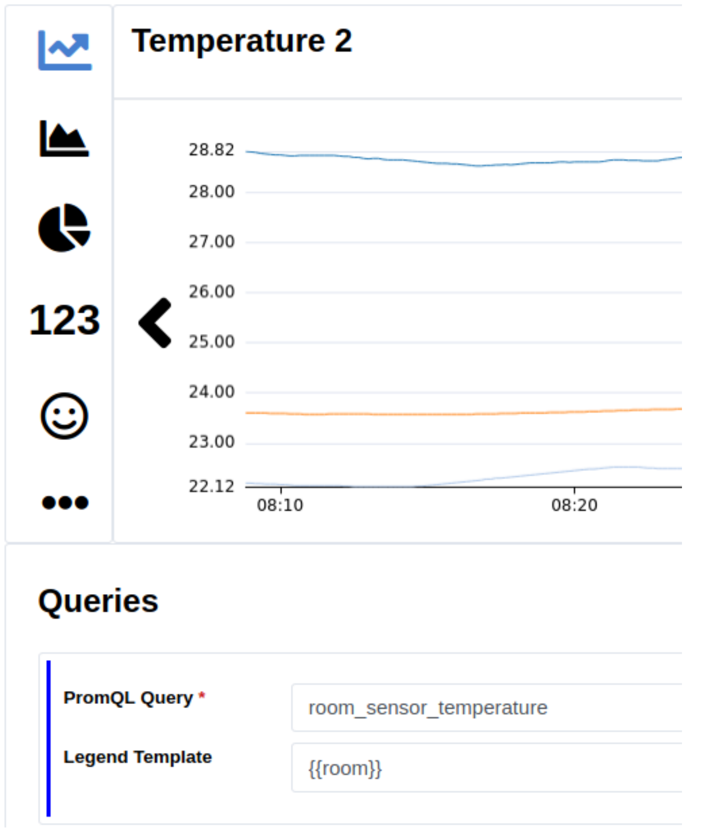
Code is available on Bleemeo Labs github: https://github.com/bleemeolabs/quote

Prometheus Drawbacks

- Metric centric, triggering alarms can be a bit more complex
- Can consume a lot of resources, especially when deployed in your Kubernetes (kubernetes lack of ressources, ops team start looking at grafana dashboards and... 💥)
- Can be complex to scale
- Not designed for high availability (without complex workaround)

Prometheus with Bleemeo

- Can replace Prometheus server and Grafana
- Data are scrapped from our Open Source agent "Glouton"
- All network streams are authenticated (different for each server) and encrypted (TLS 1.3)
- Metrics can be queried and dashboards configured using PromQL queries
- Service autodiscovery is built-in in the agent
- Coming soon: alerts can be configured using PromQL queries



PromQL Query * room_sensor_temperature

Legend Template {{room}}

Conclusion

- Became de facto standard for new monitoring projects
- Very easy to bootstrap a Prometheus + Grafana project
- Can be resource consuming
- Can be complex to scale
- You should consider it for instrumenting your code

Questions?

☞ Try Bleemeo for Free: https://bleemeo.com/trial