

# Monitoring Machine Learning Models in Production

Филиппова Ольга

DMIA Production ML  весна 2021

# Three pillars of observability

- **Metrics** ← эта лекция
- Tracing
- Logs

# Мотивация

hooray! Модель в проде!



Как узнать что модель  
ведет себя именно так как  
мы от нее ожидаем?

<https://storage.googleapis.com/pub-tools-public-publication-data/pdf/0d556e45afc54afeb2eb6b51a9bc1827b9961ff4.pdf>

# Что особенного в мониторинге ML системы?

Мониторинг ML системы состоит из двух частей:

- Service health - совпадает с мониторингом любого ИТ сервиса
- Model health - проверка на то, что модель в проде делает **полезные** предсказания

# Software systems monitoring

- RED method (Rate, Errors, Duration)
- USE method (Utilization, Saturation и Errors)
- The Four Golden Signals by Google (Latency, Traffic, Errors, Saturation)
- ...

<https://nklya.medium.com/ключевые-метрики-в-мониторинге-b6f184cf1154>

# ML systems monitoring

- Data monitoring
- Model monitoring

# Data monitoring

- что-то изменили в схеме данных
- один из источников данных внезапно отвалился
- пришла только часть данных или неправильные данные

# Data monitoring

Что можно проверить?

- Совпадают ли типы данных?
- Не изменилась ли доля пропусков в данных?
- Значения признаков остались в “нормальном” диапазоне: Появились ли новые категории? Непрерывные признаки принимают значения, которых модель раньше не видела?
- Изменились ли статистики по признакам, распределение признаков?



# Model monitoring

- Shifts in the environment
- Changes in customers behavior
- Adversarial scenarios

# Model monitoring

Что можно проверить?

- Control ML quality metrics
- Compare model prediction distributions with statistical tests:
  - Basic statistics
  - Full-blown statistical tests

# Key Monitoring-Related Principles from the Papers

- Dependency changes result in [a] notification
- Data invariants hold in training and serving inputs, i.e. monitor Training/Serving Skew
- Training and serving features compute the same values
- Models are not too stale
- The model is numerically stable
- The model has not experienced dramatic or slow-leak regressions in training speed, serving latency, throughput, or RAM usage
- The model has not experienced a regression in prediction quality on served data

# Если вы только задумались о мониторинге

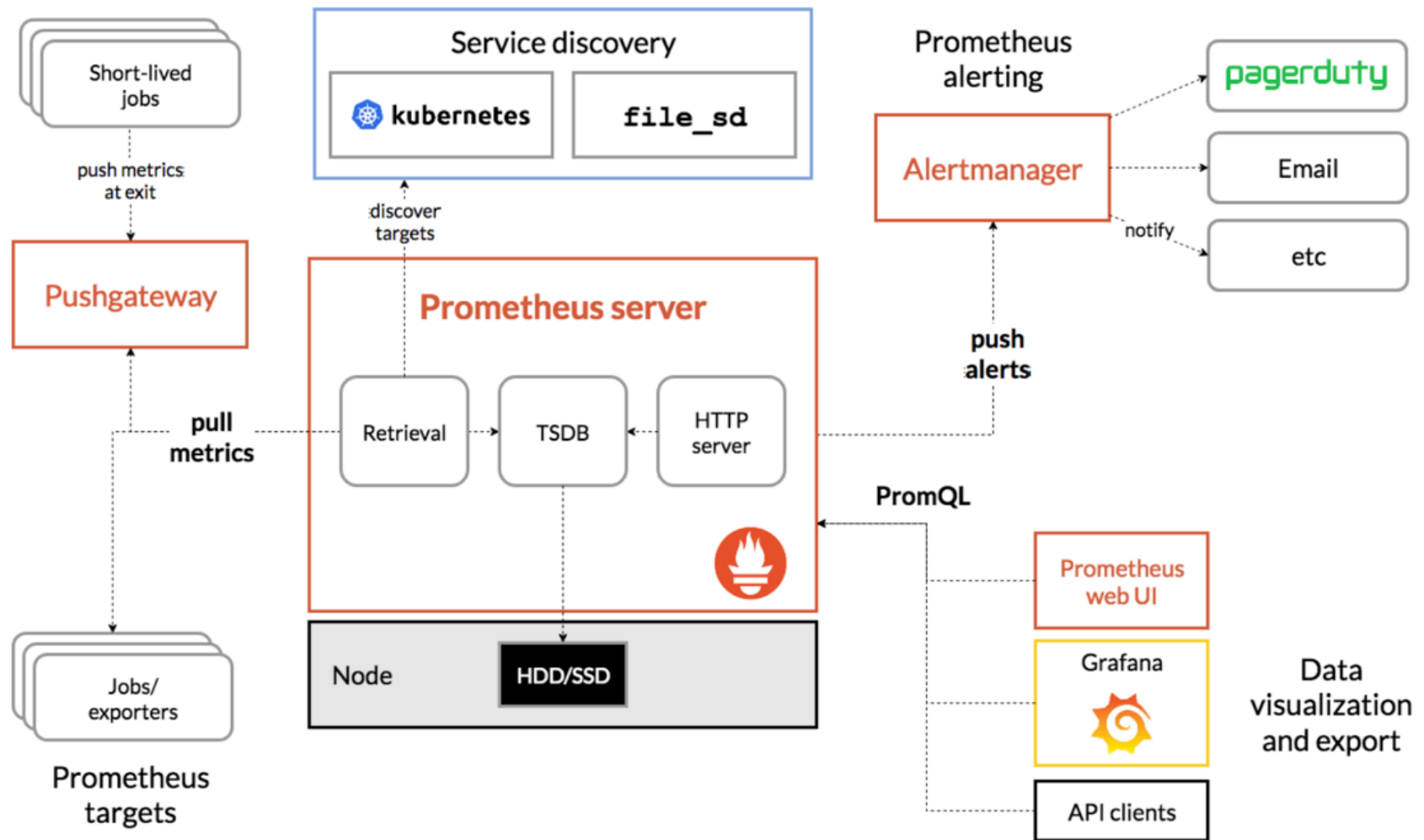
- даже базовый мониторинг лучше чем его отсутствие
- думайте о своей задаче
- следите за блогами стартапов

 Grafana

+

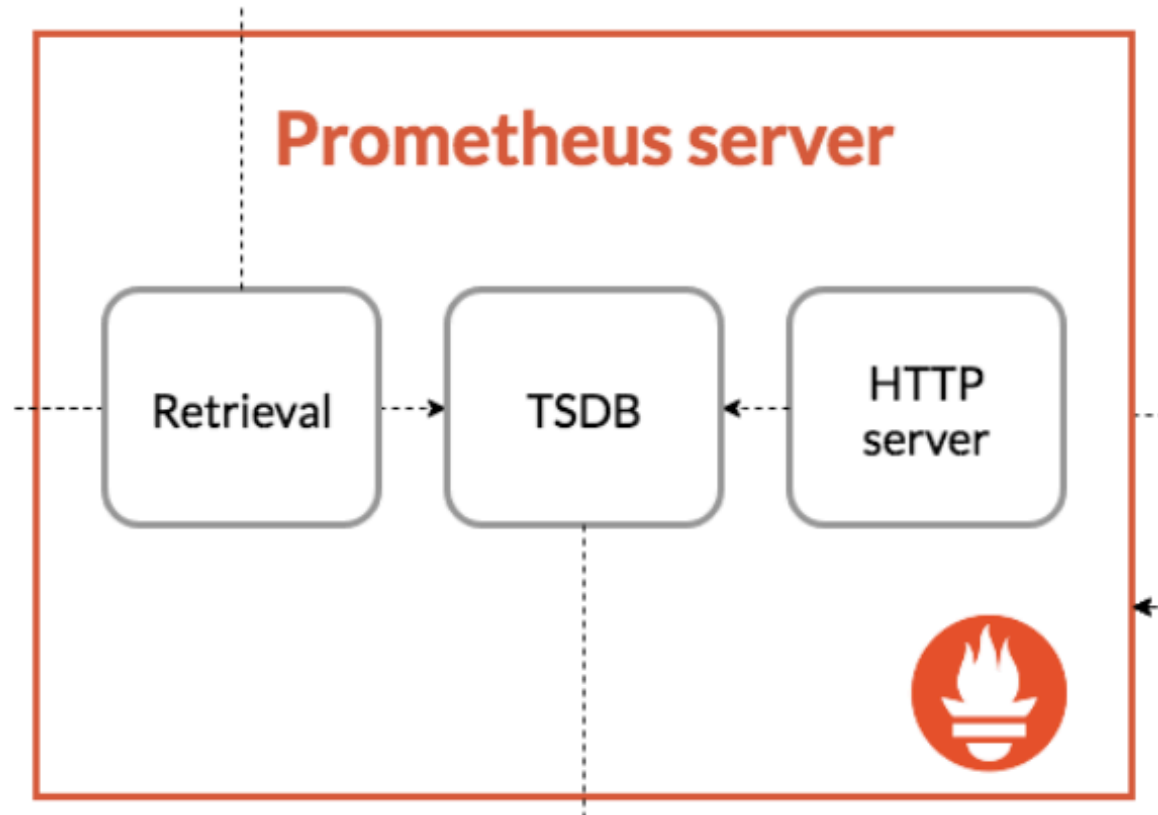
 Prometheus

# Prometheus



<https://prometheus.io/docs/introduction/overview/>

# Prometheus



# В чем особенность Prometheus?

## **push mechanism**

примеры: Amazon watch, New relic, etc

Нюансы:

- на каждый объект, который нужно отслеживать (target) необходимо установить ПО которое будет отправлять push запросы
- Если необходимо отслеживать много микросервисов и каждый из них отправляет данные в систему мониторинга, может возникнуть высоконагруженный трафик и система мониторинга становится узким местом. (инфраструктура перегружена постоянными push запросами)
- если мы перестали получать метрики от сервиса, это не может однозначно трактоваться как остановка сервиса (может что-то с сетью, или потерялся пакет, итд)



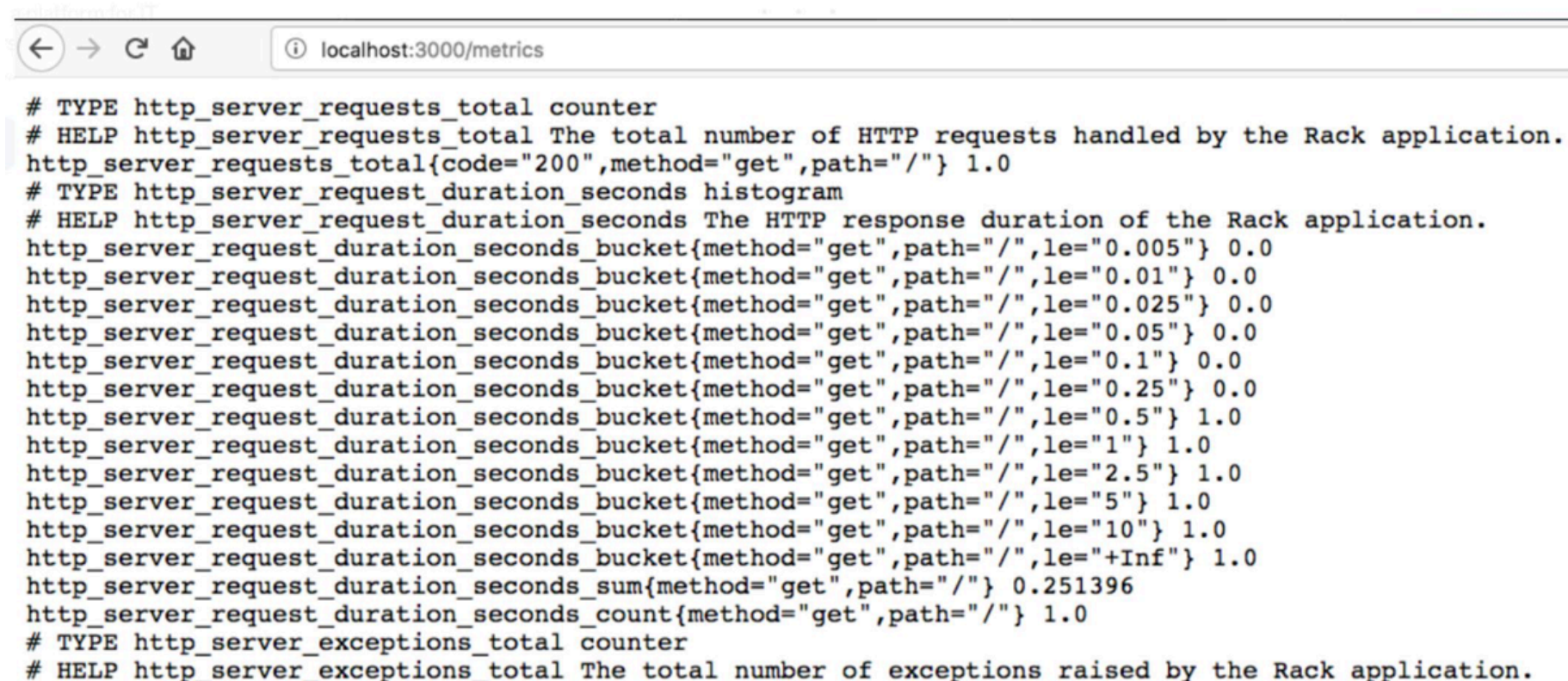
# В чем особенность Prometheus?

## **pull mechanism**

Нюансы:

- нет неопределенности, которая возникает в push mechanism системах, если запрос не пришел
- мы можем регулировать нагрузку на сеть
- микросервисы для работы с Prometheus должны иметь endpoint, кроме того для многих решений есть готовые exporters (об этом дальше)
- если нам критична точность и мы хотим учитывать каждое значение метрики, pull mechanism не лучший выбор

# Что может мониторить Prometheus?



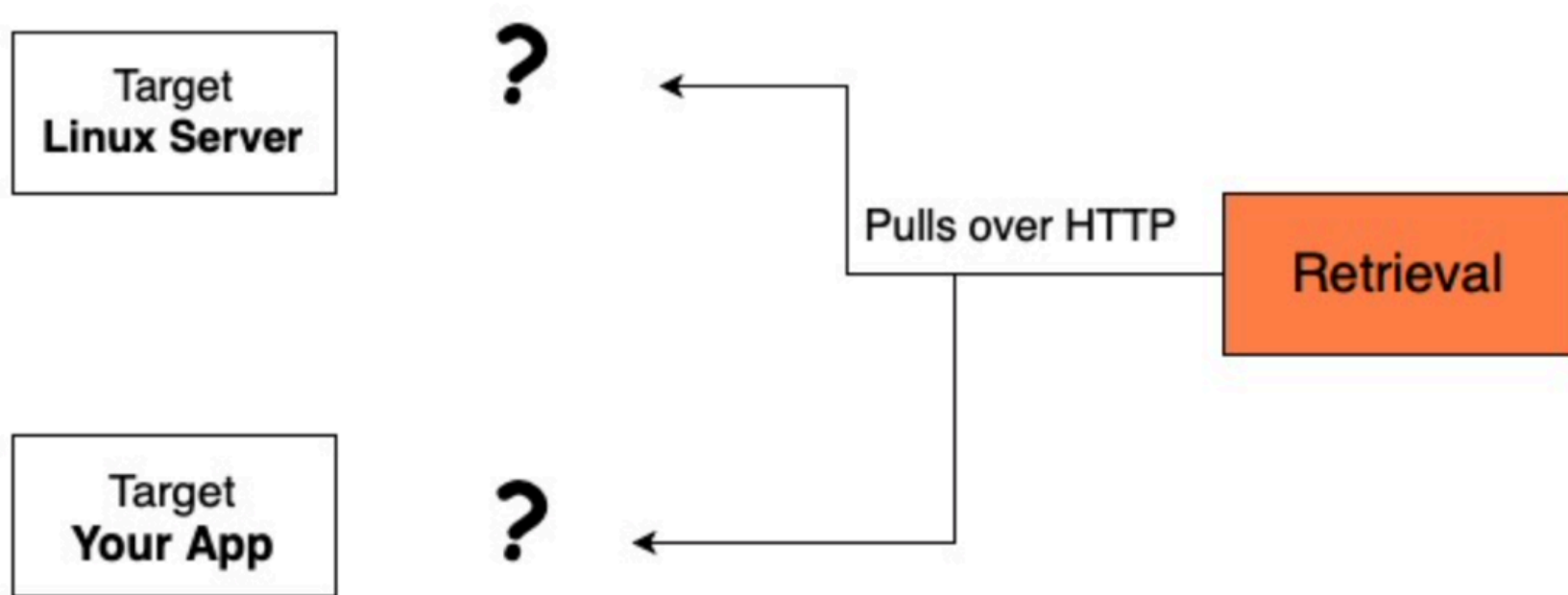
```
# TYPE http_server_requests_total counter
# HELP http_server_requests_total The total number of HTTP requests handled by the Rack application.
http_server_requests_total{code="200",method="get",path="/"} 1.0
# TYPE http_server_request_duration_seconds histogram
# HELP http_server_request_duration_seconds The HTTP response duration of the Rack application.
http_server_request_duration_seconds_bucket{method="get",path="/",le="0.005"} 0.0
http_server_request_duration_seconds_bucket{method="get",path="/",le="0.01"} 0.0
http_server_request_duration_seconds_bucket{method="get",path="/",le="0.025"} 0.0
http_server_request_duration_seconds_bucket{method="get",path="/",le="0.05"} 0.0
http_server_request_duration_seconds_bucket{method="get",path="/",le="0.1"} 0.0
http_server_request_duration_seconds_bucket{method="get",path="/",le="0.25"} 0.0
http_server_request_duration_seconds_bucket{method="get",path="/",le="0.5"} 1.0
http_server_request_duration_seconds_bucket{method="get",path="/",le="1"} 1.0
http_server_request_duration_seconds_bucket{method="get",path="/",le="2.5"} 1.0
http_server_request_duration_seconds_bucket{method="get",path="/",le="5"} 1.0
http_server_request_duration_seconds_bucket{method="get",path="/",le="10"} 1.0
http_server_request_duration_seconds_bucket{method="get",path="/",le="+Inf"} 1.0
http_server_request_duration_seconds_sum{method="get",path="/"} 0.251396
http_server_request_duration_seconds_count{method="get",path="/"} 1.0
# TYPE http_server_exceptions_total counter
# HELP http_server_exceptions_total The total number of exceptions raised by the Rack application.
```

# Типы метрик

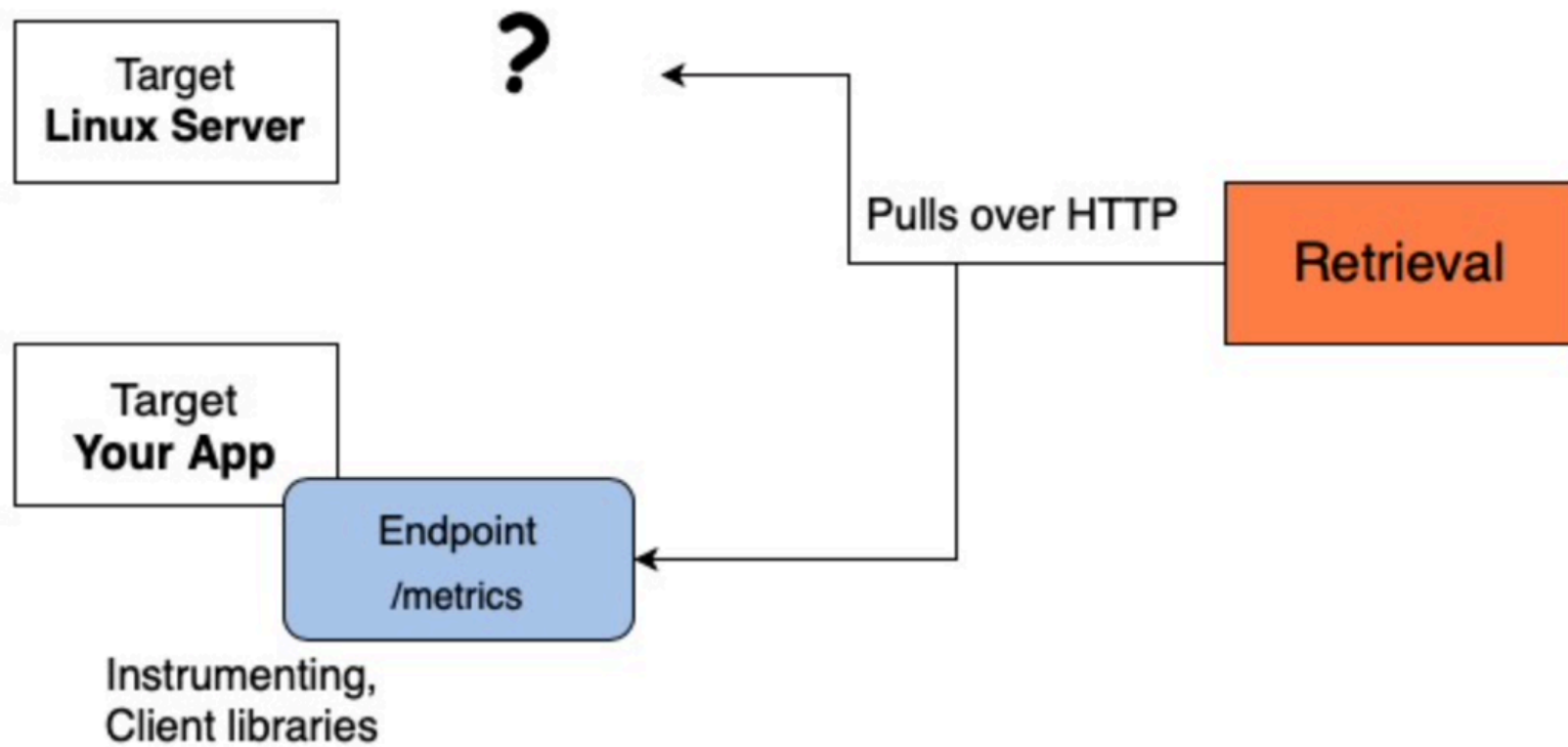
- counter - счетчики (the number of requests served, tasks completed, or errors)
- gauge - число которое может увеличиваться или уменьшаться (temperatures, current memory usage, but also "counts" that can go up and down, like the number of concurrent requests)
- Histogram - histograms, quantiles are calculated on the Prometheus server(request durations or response sizes)
- Summary - histograms, quantiles are calculated on the application server

<https://prometheus.io/docs/practices/histograms/>

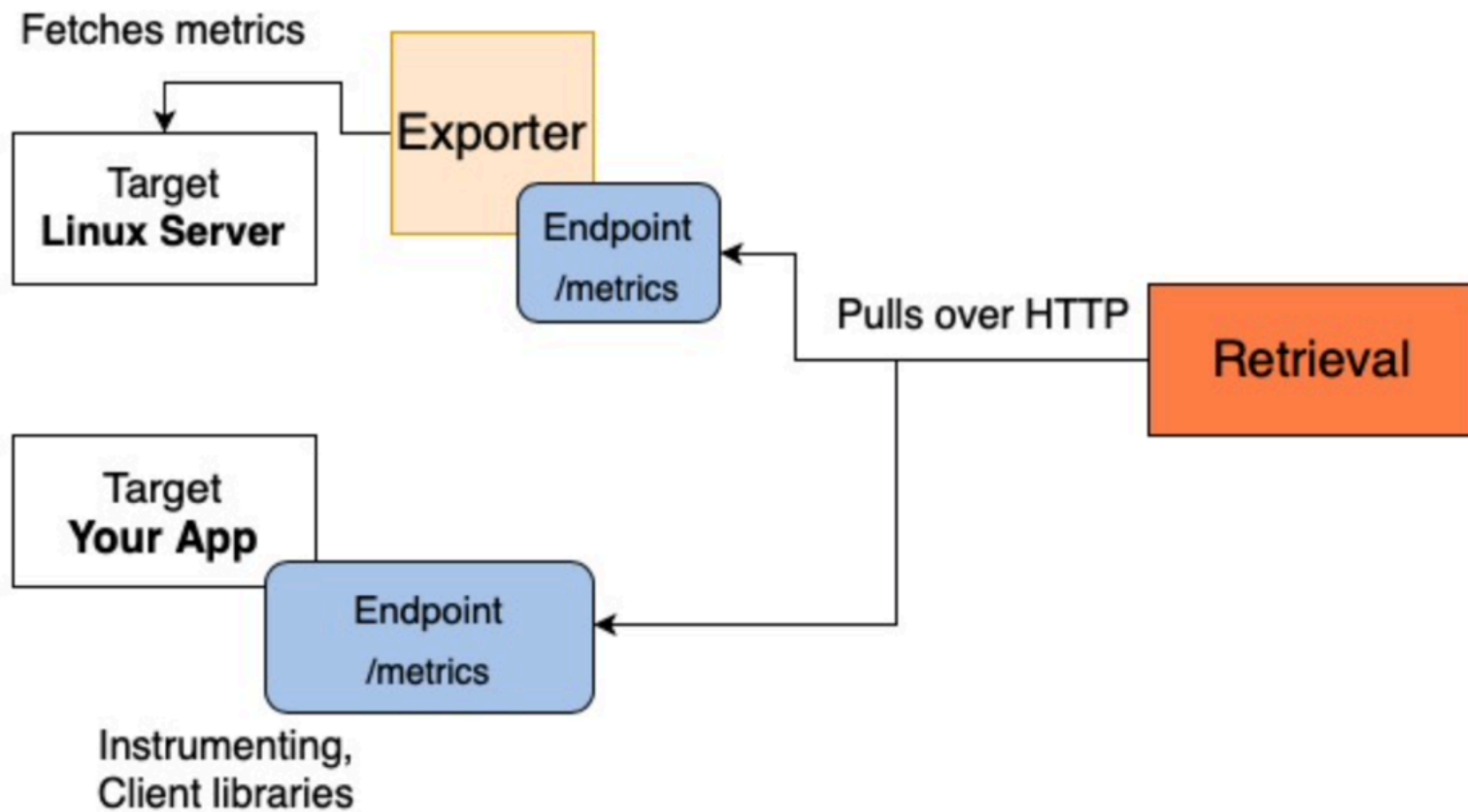
# Как все устроено?



# Как все устроено?

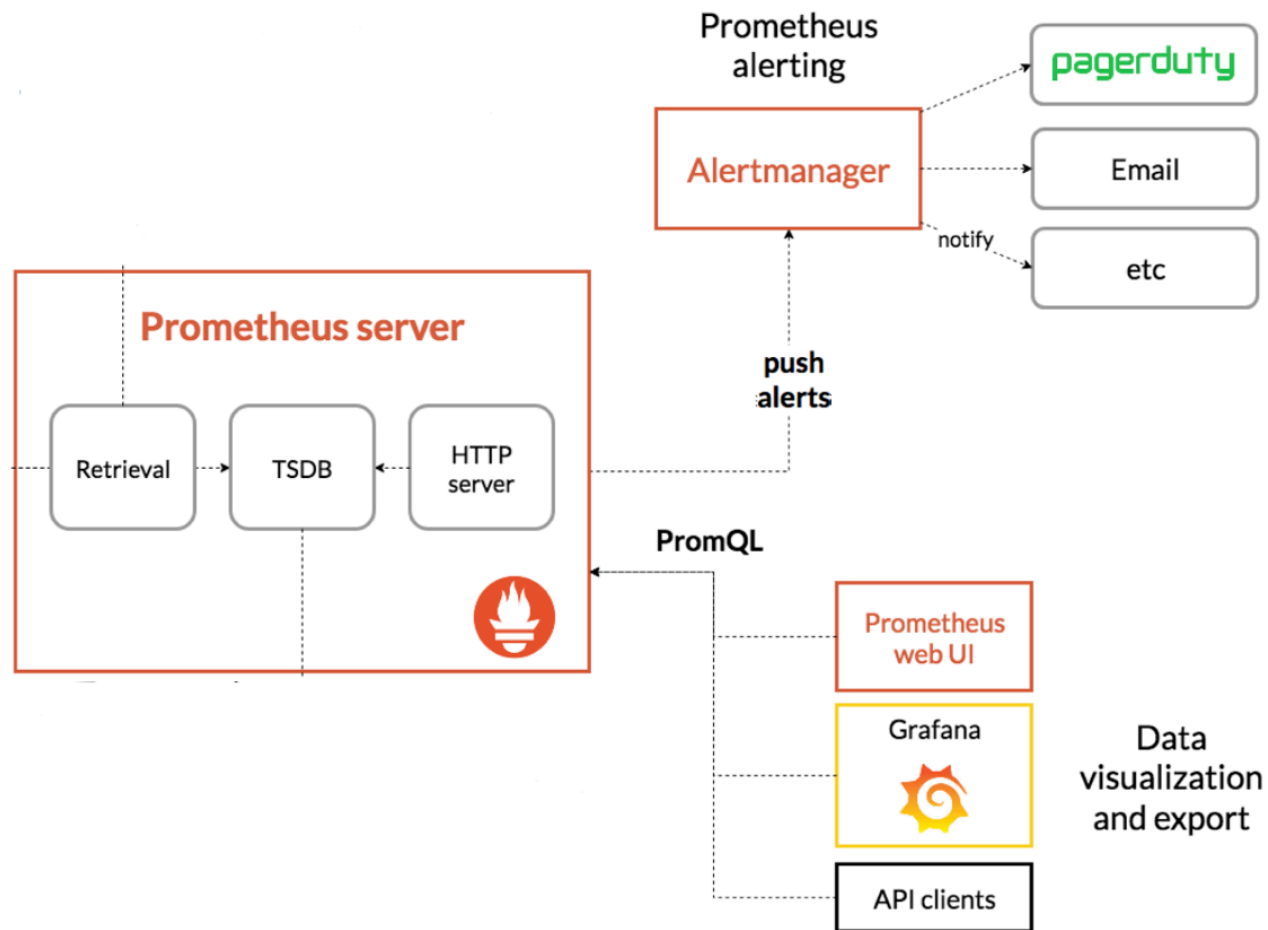


# Как все устроено?



**Как задать что и с какой частотой  
мониторить?**

# Что делать с метриками?





# Grafana



allows you to query, visualize, alert on and understand your metrics no matter where they are stored

# Grafana

- позволяет легко делать дашборды
- комбинировать на них данные из разных источников (можно на графиках по метрикам из Prometheus нанести события из логов, собранных в Loki)
- совместима с множеством источников
- позволяет настраивать alerts

## Полезные ссылки

[https://christophergs.com/machine learning/2020/03/14/how-to-monitor-machine-learning-models/](https://christophergs.com/machine-learning/2020/03/14/how-to-monitor-machine-learning-models/)

<https://evidentlyai.com/blog>

<https://www.youtube.com/watch?v=h4SI21AKiDg>