Monitoring

на примере Prometheus

Tinkoff.ru

Артем Шепелев

О чем пойдет речь?

1. Мониторинг
2. Существующие системы мониторинга
3. Архитектура Prometheus
4. Prometheus Metrics
5. Prometheus Targets
6. Prometheus Exporter
7. Prometheus Alerts
8. Prometheus Query Language
9. Prometheus Configuration
10. Prometheus Pushgateway

1

Мониторинг

Мониторинг

- Направлен на сбор данных о состоянии работы системы, приложения, отдельных компонентов и узлов.
- Преследует цели по представлению исторических данных, аналитике поведения систем, оперативному реагированию на аномальные показатели.
- Является одним из ключевых звеньев работы систем и приложений в компьютерных окружениях.

Примеры мониторинга

- Мониторинг состояния операционной системы: CPU, RAM, disk space, IO operations, File Descriptors.
- Мониторинг состояния сети: latency, packets/s, bytes/s.
- Мониторинг работы веб-сервера: requests/s, response time, upstream health, return codes.
- Мониторинг работы приложения: users online, collection count, backend response time.
- Мониторинг работы кэш-сервера: misses, hits, cache size.
- Мониторинг работы СУБД: transactions/s, queries run time, connection pool size.

2

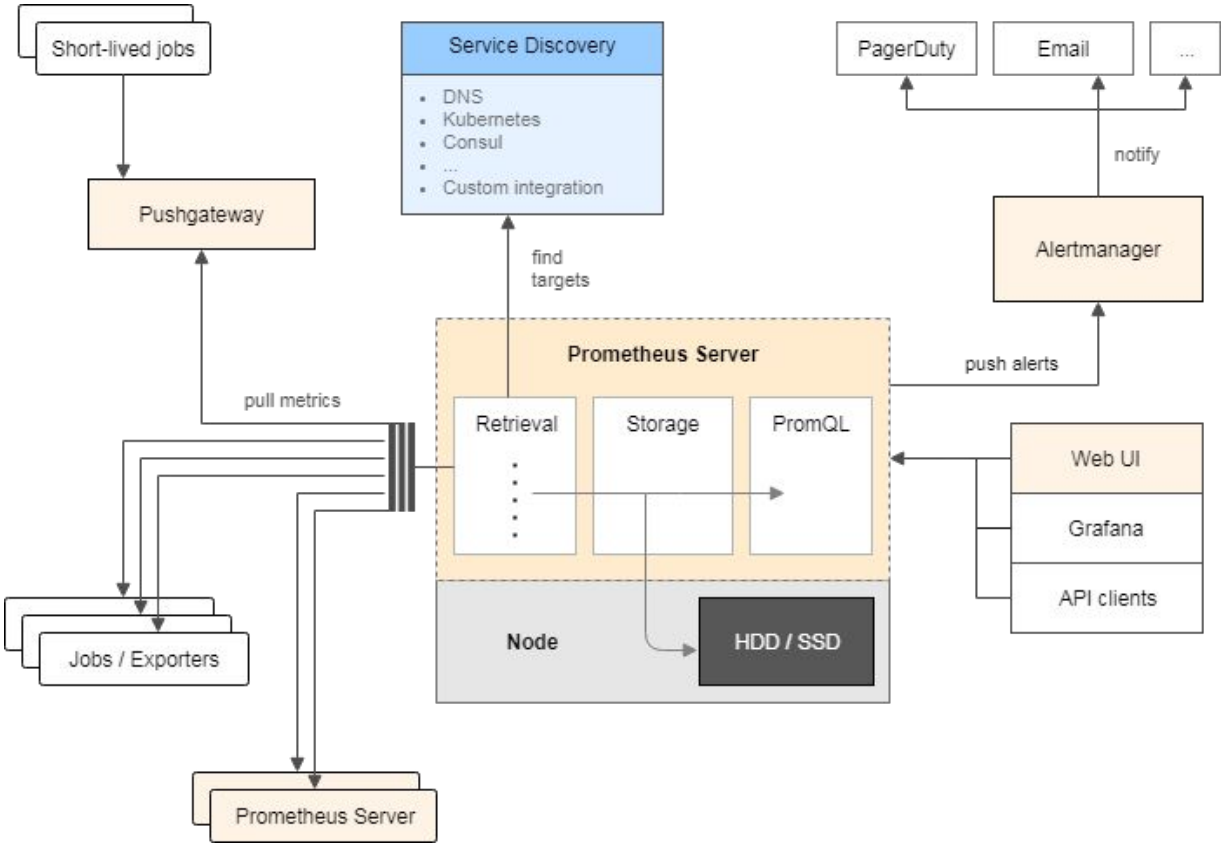
Существующие системы мониторинга

Существующие системы мониторинга

- Nagios
- Zabbix
- Graphite
- InfluxDB
- Sensu
- Prometheus

Архитектура Prometheus

Архитектура Prometheus

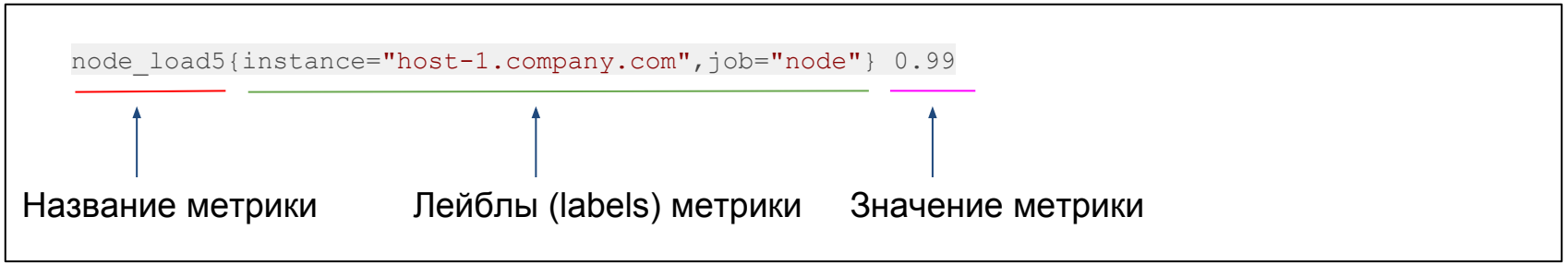


Ключевые особенности

- Time-series модель данных, предусматривающая наборы метрик с именем и набором key/value пар.
- Язык запросов, позволяющий представлять данные метрик.
- Немасштабируемая/неотказоустойчивая архитектура с одним сервером.
- Сбор данных по модели pull через HTTP вызовы.
- Сбор данных по модели push через отдельный шлюз.
- Добавление новых targets через статичную конфигурацию или динамически, через сторонние системы.

Prometheus Metrics

Prometheus Metrics



- instance - имя хоста, с которого была собрана метрика
- job - название задачи, в рамках, которой собирается данная метрика

Prometheus Metrics

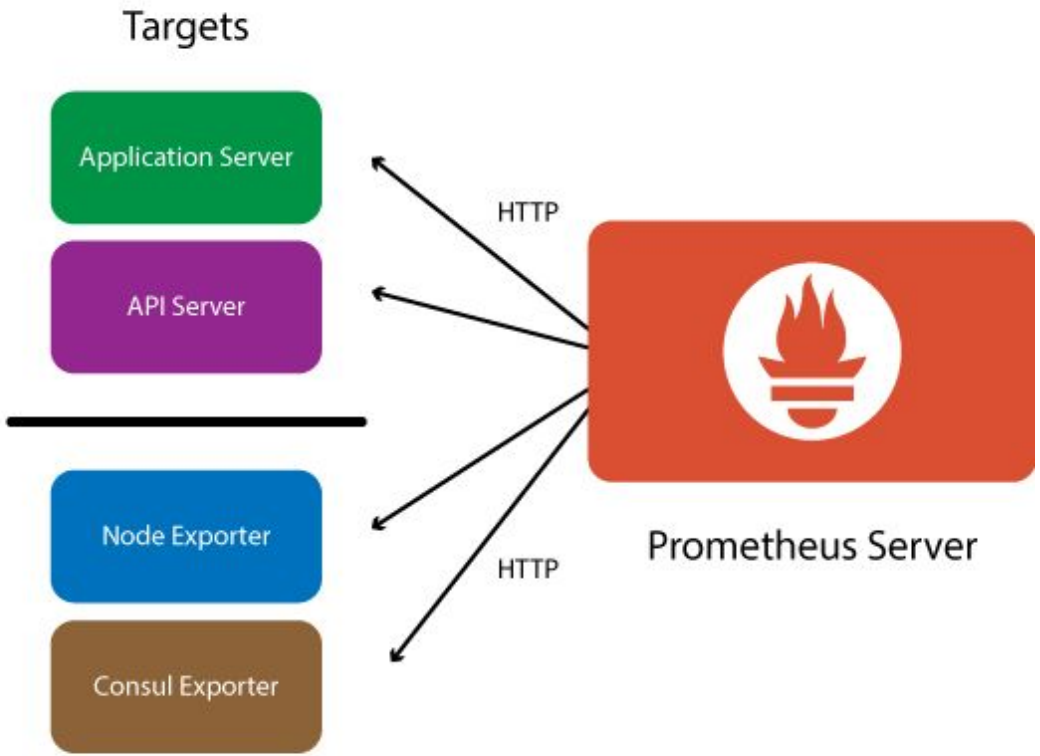
```
node_load5{env="prod",group="web",instance="msk-prod-web-app-1.company.com",job="node",service="node"}
```

В рамках конфигурации могут появляться дополнительные лейблы метрики:

- env - окружение, в котором собирается данная метрика
- group - проект/группа, к которой принадлежит данная метрика
- service - сервис (приложение), которое отдает данную метрику

Prometheus Targets

Prometheus Targets



Prometheus Targets

- Static targets
  - Заносятся вручную в конфигурацию Prometheus
- Dynamic targets
  - Используют интеграции с другими системами, для получения динамической информации о targets
    - Consul
    - Kubernetes
    - DNS
    - Azure
    - Google Compute Engine (GCE)
    - AWS EC2
    - ...

Prometheus Targets - Static

prometheus.yml:

Название job
Интервал pull-а метрик
Тип конфига job

Список хостов, с который собирать метрики

Дополнительные лейблы, накладываемые на метрики

```
  - job_name: webservers
    scrape_interval: 30s
    static_configs:
      - targets:
          - msk-prod-web-app-1.company.com:9100
          - msk-prod-web-app-2.company.com:9100
        labels:
          env: prod
          service: node
          group: web
```

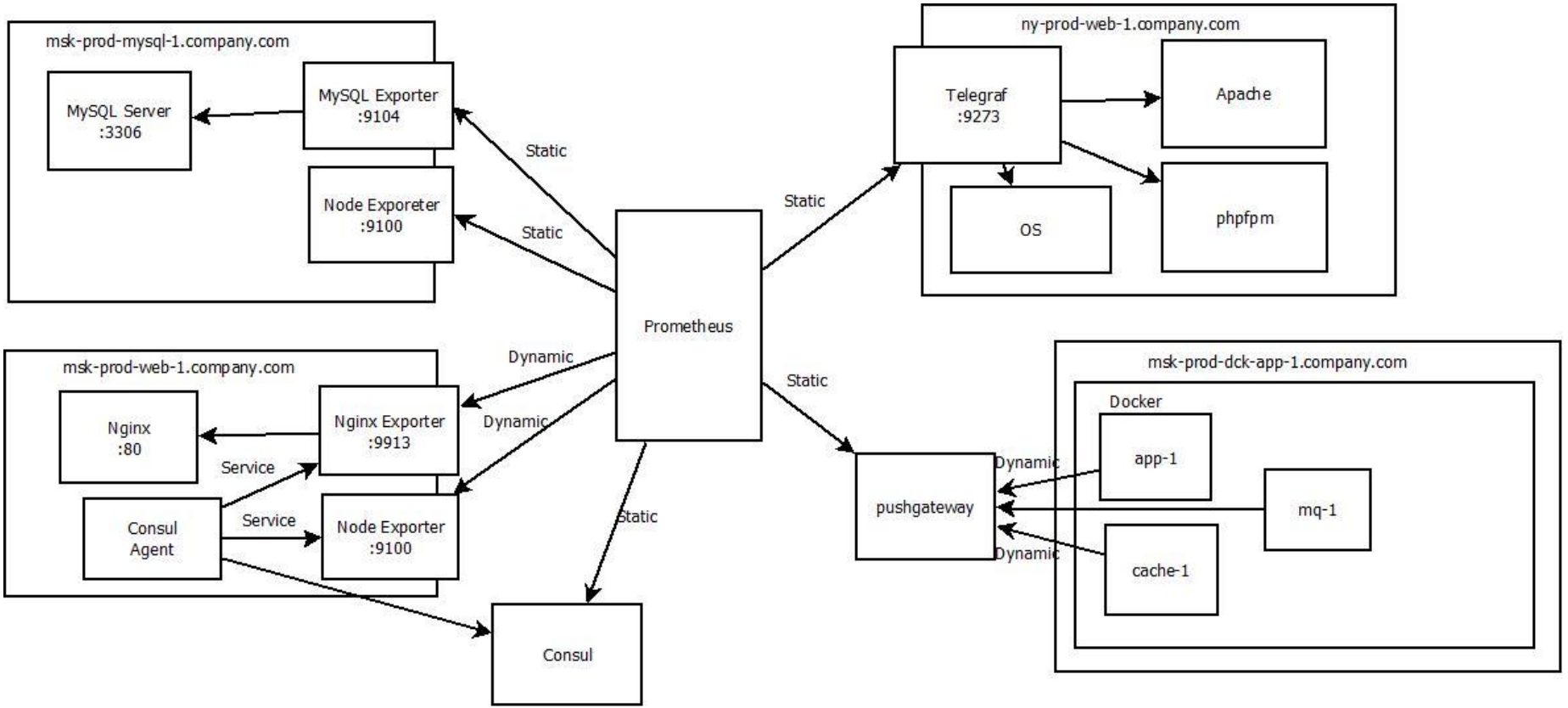
Prometheus Targets - Dynamic

- Позволяет динамически получать target-ы взаимодействуя со сторонними системами.
- Появление или удаление сервиса/хоста приводит к началу или остановке сбора метрик.
- Собирает дополнительную мета-информацию с targets для представления её в labels.

6

Prometheus Exporters

Prometheus Exporters



Prometheus Exporters

- Собирает необходимые данные с сервисов (OS, MySQL, Mongo, Nginx, JMX, Bind, Aerospike, phpfpm, и т.д.)
- Сериализуют данные в виде метрик, которые отдаются в текстовом виде через HTTP запросы.

Prometheus Exporters

```
admin@prometheus:~ # curl http://msk-prod-web-app-1.company.com:9100/metrics
# HELP node_boot_time Node boot time, in unixtime.
# TYPE node_boot_time gauge
node_boot_time 1.526650758e+09
# HELP node_cpu Seconds the cpus spent in each mode.
# TYPE node_cpu counter
node_cpu{cpu="cpu0",mode="idle"} 5.17059402e+06
node_cpu{cpu="cpu0",mode="iowait"} 170773.19
# HELP node_disk_bytes_read The total number of bytes read successfully.
# TYPE node_disk_bytes_read counter
node_disk_bytes_ read{device="sda"} 7.6266846208e+10
# HELP node_disk_bytes_written The total number of bytes written successfully.
# TYPE node_disk_bytes_written counter
node_disk_bytes_written{device= "sda"} 8.0542205952e+11
# HELP node_disk_io_time_ms Total Milliseconds spent doing I/Os.
# TYPE node_disk_io_time_ms counter
node_disk_io_time_ms{device= "sda"} 5.35792636e+08
# HELP node_filefd_allocated File descriptor statistics: allocated.
# TYPE node_filefd_allocated gauge
node_filefd_allocated 1568
# HELP node_filefd_maximum File descriptor statistics: maximum.
# TYPE node_filefd_maximum gauge
node_filefd_maximum 813514
# HELP node_filesystem_avail Filesystem space available to non-root users in bytes.
# TYPE node_filesystem_avail gauge
node_filesystem_avail{device= "/dev/sda1",fstype="ext4",mountpoint="/"} 2.186958848e+10
```

Prometheus Exporters

```
# HELP node_boot_time Node boot time, in unixtime.
# TYPE node_boot_time gauge
node_boot_time{instance= "msk-prod-web-app-1.company.com" ,job="node",service="node",env="prod",group="web"} 1.526650758e+09
# HELP node_cpu Seconds the cpus spent in each mode.
# TYPE node_cpu counter
node_cpu{instance= "msk-prod-web-app-1.company.com" ,job="node",service="node",env="prod",group="web",cpu="cpu0",mode="idle"} 5.17059402e+06
node_cpu{instance= "msk-prod-web-app-1.company.com" ,job="node",service="node",env="prod",group="web",cpu="cpu0",mode="iowait"} 170773.19
# HELP node_disk_bytes_read The total number of bytes read successfully.
# TYPE node_disk_bytes_read counter
node_disk_bytes_ read{instance="msk-prod-web-app-1.company.com" ,job="node",service="node",env="prod",group="web",device="sda"}
7.6266846208e+10
# HELP node_disk_bytes_written The total number of bytes written successfully.
# TYPE node_disk_bytes_written counter
node_disk_bytes_written{instance= "msk-prod-web-app-1.company.com" ,job="node",service="node",env="prod",group="web",device="sda"}
8.0542205952e+11
# HELP node_disk_io_time_ms Total Milliseconds spent doing I/Os.
# TYPE node_disk_io_time_ms counter
node_disk_io_time_ms{instance= "msk-prod-web-app-1.company.com" ,job="node",service="node",env="prod",group="web",device="sda"} 5.35792636e+08
# HELP node_filefd_allocated File descriptor statistics: allocated.
# TYPE node_filefd_allocated gauge
node_filefd_allocated{instance= "msk-prod-web-app-1.company.com" ,job="node",service="node",env="prod",group="web"} 1568
# HELP node_filefd_maximum File descriptor statistics: maximum.
# TYPE node_filefd_maximum gauge
node_filefd_maximum{instance= "msk-prod-web-app-1.company.com" ,job="node",service="node",env="prod",group="web"} 813514
# HELP node_filesystem_avail Filesystem space available to non-root users in bytes.
# TYPE node_filesystem_avail gauge
node_filesystem_avail{instance= "msk-prod-web-app-1.company.com" ,job="node",service="node",group="web",device="/dev/sda1",fstype="ext4",mount
point="/"} 2.186958848e+10
```

Prometheus Query Language

Prometheus Query Language

- PromQL - язык функциональных выражений, позволяющий выбирать и агрегировать данные в реальном времени.
- Результат может быть визуализирован в виде графа, выведен в табличном виде в веб-интерфейсе Prometheus или получен через HTTP API.
- Запросы бывают:
  - Запрос текущего значения (instant vector)
  - Запрос данных за последнее время (range vector)
- Поддерживает большое количество способов агрегирования данных

Prometheus Query - Операторы

- Операторы бывают:
  - Арифметические бинарные операции (+,-,*,/,%,^)
  - Бинарные операции сравнения (==,!=,>,<,>=,<=)
  - Логические бинарные операции (and, or, unless)
  - Векторное сопоставление (“JOIN” метрик по ключу значений лейблов)

Prometheus Query - Операторы - Примеры

```
rancher_host_agent_state{state=~"reconnecting|reconnected|disconnecting"} == 1

(java_OpenFileDescriptorCount /  java_MaxFileDescriptorCount) * 100 > 80

probe_duration_seconds > 1 and probe_success != 0

(node_filesystem_files_free / ON(mountpoint, instance, device) GROUP_LEFT()
node_filesystem_files) * 100 < 5
```

Prometheus Query - Функции

- Функции позволяет агрегировать вектор метрик по времени
- Операции над текущими значениями:
  - abs, ceil, floor, log2, round, sort, sqrt, и т.д.
- Операции над вектором значений за определенный промежуток:
  - delta - возвращает разницу между первым и последним значением метрики в промежутке
  - rate - возвращает среднее увеличение метрики за заданный промежуток в секунду. 50 -> 30 секунд -> 100 -> 30 секунд -> 150 = 1.6
  - increase - rate, которые умножается на количество секунд. Даст значение 50 в предыдущем примере.
  - irate - rate, с instant увеличением метрики за последние два показателя метрики
  - sum - считает сумму метрик за указанный промежуток
  - sum(irate(<metric_name>[5m])) - посчитает сумму увеличений значений метрики

Prometheus Query - Функции - Пример

node_cpu{instance="msk-prod-web-1.company.com",mode="idle"}

```
node_cpu{cluster="my_cluster",cpu="cpu0",env="prod",group="web",instance="msk-prod-web-1.compan
y.com",job="node",mode="idle",service="mq"}  11192588.62
node_cpu{cluster="my_cluster",cpu="cpu1",env="prod",group="web",instance="msk-prod-web-1.compan
y.com",job="node",mode="idle",service="mq"}  11165289.81
node_cpu{cluster="my_cluster",cpu="cpu2",env="prod",group="web",instance="msk-prod-web-1.compan
y.com",job="node",mode="idle",service="mq"}  11051004.13
node_cpu{cluster="my_cluster",cpu="cpu3",env="prod",group="web",instance="msk-prod-web-1.compan
y.com",job="node",mode="idle",service="mq"}  11173625.68
node_cpu{cluster="my_cluster",cpu="cpu4",env="prod",group="web",instance="msk-prod-web-1.compan
y.com",job="node",mode="idle",service="mq"}  11186385.82
node_cpu{cluster="my_cluster",cpu="cpu5",env="prod",group="web",instance="msk-prod-web-1.compan
y.com",job="node",mode="idle",service="mq"}  11187443.53
node_cpu{cluster="my_cluster",cpu="cpu6",env="prod",group="web",instance="msk-prod-web-1.compan
y.com",job="node",mode="idle",service="mq"}  11196599.55
node_cpu{cluster="my_cluster",cpu="cpu7",env="prod",group="web",instance="msk-prod-web-1.compan
y.com",job="node",mode="idle",service="mq"}  11194539.57
```

Prometheus Query - Функции - Пример

delta(node_cpu{instance="msk-prod-web-1.company.com",mode="idle"}[2h])

```
{cluster="my_cluster",cpu="cpu0",env="prod",group="web",instance="msk-prod-web-1.company.com",job="node",mode="idle",service="mq"} 7111.220083681858
{cluster="my_cluster",cpu="cpu1",env="prod",group="web",instance="msk-prod-web-1.company.com",job="node",mode="idle",service="mq"} 7075.380753138449
{cluster="my_cluster",cpu="cpu2",env="prod",group="web",instance="msk-prod-web-1.company.com",job="node",mode="idle",service="mq"} 6807.4744769868485
{cluster="my_cluster",cpu="cpu3",env="prod",group="web",instance="msk-prod-web-1.company.com",job="node",mode="idle",service="mq"} 7115.7087866114025
{cluster="my_cluster",cpu="cpu4",env="prod",group="web",instance="msk-prod-web-1.company.com",job="node",mode="idle",service="mq"} 7111.852719665945
{cluster="my_cluster",cpu="cpu5",env="prod",group="web",instance="msk-prod-web-1.company.com",job="node",mode="idle",service="mq"} 7094.4803347279585
{cluster="my_cluster",cpu="cpu6",env="prod",group="web",instance="msk-prod-web-1.company.com",job="node",mode="idle",service="mq"} 7101.39916318044
{cluster="my_cluster",cpu="cpu7",env="prod",group="web",instance="msk-prod-web-1.company.com",job="node",mode="idle",service="mq"} 7092.411715480573
```

Prometheus Query - Функции - Пример

rate(node_cpu{instance="msk-prod-web-1.company.com",mode="idle"}[2h])

```
{cluster="my_cluster",cpu="cpu0",env="prod",group="web",instance="msk-prod-web-1.company.com",jo
b="node",mode="idle",service="mq"} 0.987753138075241
{cluster="my_cluster",cpu="cpu1",env="prod",group="web",instance="msk-prod-web-1.company.com",jo
b="node",mode="idle",service="mq"} 0.9826889818687526
{cluster="my_cluster",cpu="cpu2",env="prod",group="web",instance="msk-prod-web-1.company.com",jo
b="node",mode="idle",service="mq"} 0.9457196652717794
{cluster="my_cluster",cpu="cpu3",env="prod",group="web",instance="msk-prod-web-1.company.com",jo
b="node",mode="idle",service="mq"} 0.9881520223153061
{cluster="my_cluster",cpu="cpu4",env="prod",group="web",instance="msk-prod-web-1.company.com",jo
b="node",mode="idle",service="mq"} 0.9877824267782946
{cluster="my_cluster",cpu="cpu5",env="prod",group="web",instance="msk-prod-web-1.company.com",jo
b="node",mode="idle",service="mq"} 0.9851868898185642
{cluster="my_cluster",cpu="cpu6",env="prod",group="web",instance="msk-prod-web-1.company.com",jo
b="node",mode="idle",service="mq"} 0.986266387726805
{cluster="my_cluster",cpu="cpu7",env="prod",group="web",instance="msk-prod-web-1.company.com",jo
b="node",mode="idle",service="mq"} 0.9850292887029496
```

Prometheus Query - Функции - Пример

```
sum(rate(node_cpu{instance="msk-prod-web-1.company.com",mode="idle"}[2h]))
```

{} **7.848598326359467**

Prometheus Alerts

Prometheus Alerts

- Позволяют описывать правила, определяющие необходимость создания оповещения об аномальном поведении сервиса.
- Позволяют использовать языка PromQL для агрегирования данных. Данный язык покрывает большое количество функций над метриками.
- Каждая правило алерта, позволяет добавлять к оповещению дополнительные данные в формате key/value.
- Дополнительные данные делятся на лейблы (labels) и аннотации (annotations).
- Благодаря языку описания значений (jinja-template) есть возможность генерировать полезные сообщения, содержащие в себе дополнительную информацию, которая берется из лейблов метрики.

Prometheus Alerts - Пример

Prometheus Alerts - Пример

```
    - alert: FreeInodes
      expr: (node_filesystem_files_free / ON(mountpoint, instance, device) GROUP_LEFT() node_filesystem_files) * 100 < 5
      for: 1s
      labels:
        severity: critical
        env: "{{ $labels.env }}"
        group: "{{ $labels.group }}"
        service: "{{ $labels.service }}"
      annotations:
        summary: Host {{ $labels.instance }} reports low inodes count on {{ $labels.mountpoint }} (current usage - {{ printf "%.1f" $value }}%)
        grafana_url: http://grafana.company.com/d/000000133/node-full?orgId=1&var-datasource=Prometheus&var-node={{ $labels.instance }}
```

Prometheus Alerts - Пример

```
    - alert: NginxHigh5xxRate
      expr: increase(nginx_vts_server_requests_total{code="5xx"}[5m]) > 0
      for: 5m
      labels:
        severity: warning
        env: "{{ $labels.env }}"
        group: "{{ $labels.group }}"
        service: "{{ $labels.service }}"
      annotations:
        summary: Количество 5xx ошибок на  {{ $labels.instance }}  host - {{ $labels.host }}  высокое - {{ printf "%.1f" $value }}
ошибок
        brief_summary: Host -  {{ $labels.host }}, value - {{ printf "%.1f" $value }}  requests
        grafana_url: http://grafana.company.com/d/zxqx9Ddik/nginx-vts-stats?orgId=1&var-Instance=   {{ $labels.instance }}
```

9

Prometheus Configuration

Prometheus Configuration

- Скачиваем prometheus
  https://github.com/prometheus/prometheus/releases/download/v2.3.2/prometheus-2.3.2.linux-amd64.tar.gz

Prometheus Configuration

prometheus.yml:

Глобальный интервал сбора метрик

Секция конфигурации alertmanager

Секция конфигурации targets

Job, который собирает метрики ноды (метрики ОС)

Job, который собирает метрики Prometheus

```yaml
---
# yamllint disable rule:line-length
global:
  scrape_interval:  30s
alerting:
  alertmanagers:
    - timeout:  1s
      static_configs:
        - targets:
            - alertmanager.company.com
scrape_configs:
  - job_name: node
    static_configs:
      - targets:
          - localhost: 9100
        labels:
          group: devops
          service: node
          env: dev
  - job_name: node
    static_configs:
      - targets:
          - localhost: 9090
        labels:
          group: devops
          service: prometheus
          env: dev
```

Prometheus Configuration

```
user00@your.host.name:~$ ./prometheus
level=info ts=2018-07-22T17:28:07.354862032Z caller=main.go:222 msg="Starting Prometheus" version="(version=2.3.2,
branch=HEAD, revision=71af5e29e815795e9dd14742ee7725682fa14b7b)"
level=info ts=2018-07-22T17:28:07.355011052Z caller=main.go:223 build_context="(go=go1.10.3, user=root@5258e0bd9cc1,
date=20180712-14:02:52)"
level=info ts=2018-07-22T17:28:07.355135488Z caller=main.go:224 host_details="(Linux 4.9.0-3-amd64 #1 SMP Debian
4.9.30-2+deb9u5 (2017-09-19) x86_64 ash-debian (none))"
level=info ts=2018-07-22T17:28:07.355214932Z caller=main.go:225 fd_limits="(soft=1024, hard=1048576)"
level=info ts=2018-07-22T17:28:07.356383046Z caller=main.go:533 msg="Starting TSDB ..."
level=info ts=2018-07-22T17:28:07.388320942Z caller=web.go:415 component=web msg="Start listening for connections"
address=0.0.0.0:9090
level=info ts=2018-07-22T17:28:07.4179943Z caller=main.go:543 msg="TSDB started"
level=info ts=2018-07-22T17:28:07.418093148Z caller=main.go:603 msg="Loading configuration file" filename=prometheus.yml
level=info ts=2018-07-22T17:28:07.423610535Z caller=main.go:629 msg="Completed loading of configuration file"
filename=prometheus.yml
level=info ts=2018-07-22T17:28:07.42376939Z caller=main.go:502 msg="Server is ready to receive web requests."
```

Prometheus Configuration

- Загрузим стандартный экспортер данных ОС: https://github.com/prometheus/node_exporter/releases/download/v0.16.0/node_exporter-0.16.0.linux-amd64.tar.gz

```
user00@your.host.name:~$ ./node_exporter &
[1] 15878
user00@your.host.name:~$ INFO[0000] Starting node_exporter (version=0.15.2, branch=HEAD,
revision=98bc64930d34878b84a0f87dfe6e1a6da61e532d)    source="node_exporter.go:43"
INFO[0000] Build context (go=go1.9.2, user=root@d5c4792c921f, date=20171205-14:50:53)
source="node_exporter.go:44"
INFO[0000] No directory specified, see --collector.textfile.directory    source="textfile.go:57"
INFO[0000] Enabled collectors:                                    source="node_exporter.go:50"
INFO[0000]  - arp                                                 source="node_exporter.go:52"
INFO[0000]  - loadavg                                             source="node_exporter.go:52"
INFO[0000]  - stat                                                source="node_exporter.go:52"
INFO[0000]  - zfs                                                 source="node_exporter.go:52"
INFO[0000]  - netstat                                             source="node_exporter.go:52"
INFO[0000]  - entropy                                             source="node_exporter.go:52"
INFO[0000]  - bcache                                              source="node_exporter.go:52"
INFO[0000]  - diskstats                                           source="node_exporter.go:52"
INFO[0000]  - wifi                                                source="node_exporter.go:52"
INFO[0000]  - uname                                               source="node_exporter.go:52"
INFO[0000]  - xfs                                                 source="node_exporter.go:52"
```

Домашнее задание

https://gitlab.com/tfs_s18_admin/homework/blob/master/materials/class08/README.md

Полезные ссылки

- Официальная документация: https://prometheus.io/docs/introduction/overview/
- Репозиторий: https://github.com/prometheus/prometheus
- Node_exporter: https://github.com/prometheus/node_exporter

Спасибо за внимание!

Артем Шепелев
a.shepelev@tinkoff.ru
Telegram: @ashepelev