

Sign in to viblo.asia with Google

Khanh Nguyen Huy
khanhnh@s.hanu.edu.vnKhanh Nguyen Huy
vitgacume@gmail.com

1 more account

2.5K 4 15



Dao Thai Son @dao.thai.son

Theo dõi

★ 5.8K 👤 211 ✎ 56

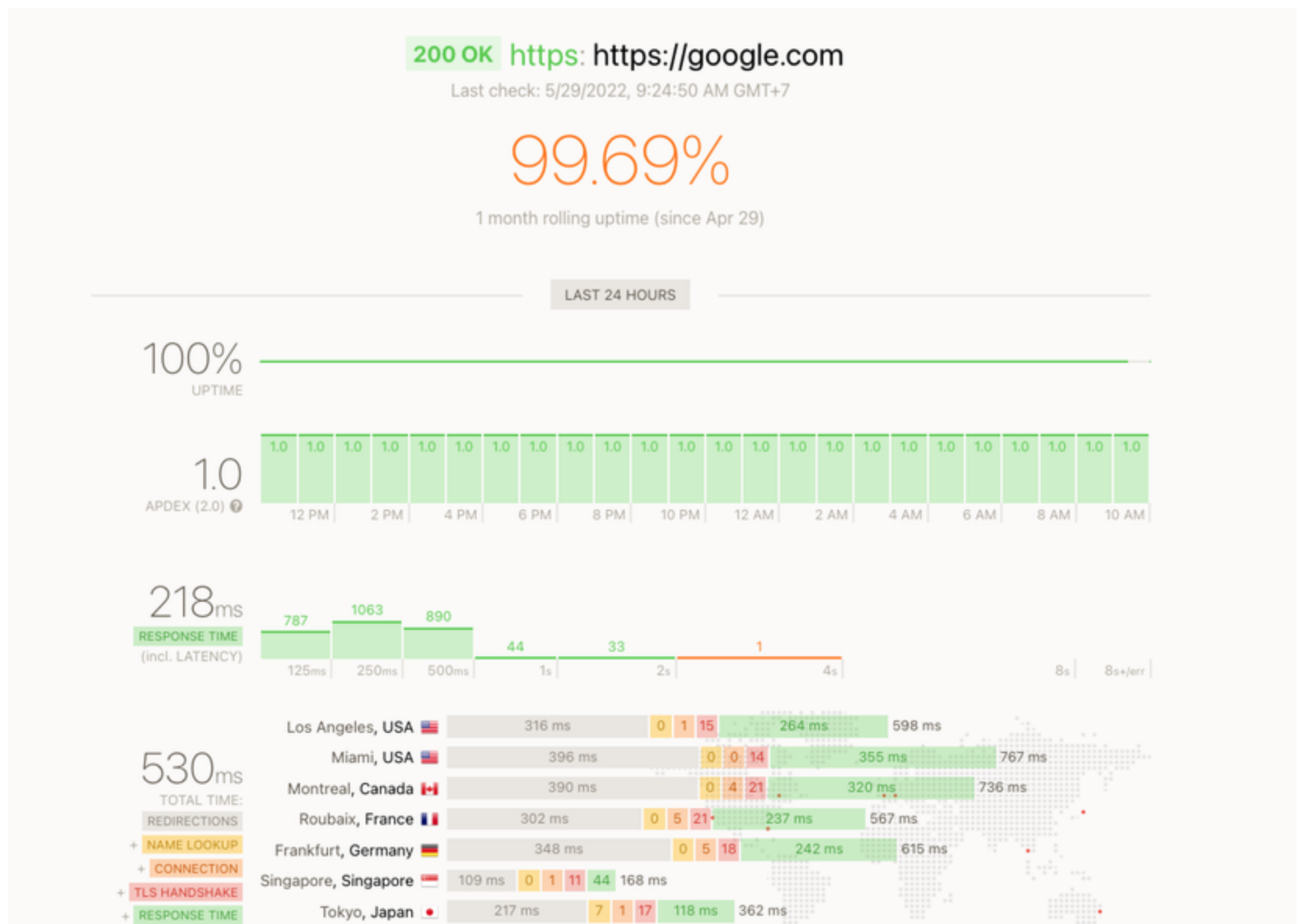
Sử dụng Prometheus, Blackbox Exporter, Alert Manager và Grafana để theo dõi trạng thái uptime website như updown.io

MayFest2022

Reconnection

...

Uptime là gì? Tại sao cần theo dõi nó ?



Uptime (thường biểu thị theo phần trăm) cho biết tỉ lệ phần trăm hoạt động đầy đủ của website. Tỷ lệ phần trăm còn lại là mức độ thường xuyên mà nó không hoạt động. Nếu dịch vụ hosting web nói uptime là 93%, nghĩa là server của họ bị downtime trung bình 7% thời gian. Một trong những tính quan trọng nhất của trang web là tính khả dụng. Bạn có thể sở hữu nội dung tuyệt vời, hình ảnh sản phẩm đẹp mắt, và những tính năng đặc biệt đầy thu hút, nhưng khi người dùng cần sử dụng tới thì website lại không hoạt động, điều đó thật sự đem lại trải nghiệm xấu cho



↑ +18 ↓



người dùng, họ sẵn sàng trải nghiệm 1 website khác thay thế. Đó là lý do tại sao việc theo dõi hiệu suất trang web là cần thiết. Chỉ 1 phút ngừng hoạt động có thể khiến công ty của bạn mất hàng ngàn đô la. Hãy tưởng tượng đến những mất mát từ phút kéo dài sang hàng giờ hoặc thậm chí vài ngày.

Làm thế nào để theo dõi trạng thái Uptime của website ?

Có một số công cụ giám sát uptime website miễn phí sẽ cho bạn biết khi nào trang web ngừng hoạt động ví dụ như <https://updown.io>, <https://uptime.com>,... Tuy nhiên đa số chúng đều giới hạn số lượng website được sử dụng. Vì vậy, sẽ thật tuyệt khi bạn tự xây dựng cho mình 1 công cụ và tùy ý sử dụng chúng cho bao nhiêu website, bao nhiêu endpoint mà bạn mong muốn.

Hiện tại, mình đang sử dụng bộ công cụ gồm: **Prometheus, Blackbox Exporter, Alert Manager và Grafana** để theo dõi trạng thái của các website



Trong bài viết này, mình sẽ chia sẻ cách mình tự xây dựng công cụ giám sát uptime như thế nào, đây là cấu trúc thư mục của project:



↑ +18 ↓



```
.
├── config
│   ├── alertmanager.yml
│   ├── blackbox
│   │   └── blackbox.yml
│   ├── grafana
│   │   └── provisioning
│   │       ├── dashboards
│   │       │   ├── Prometheus\ Blackbox\ Exporter-1653754328923.json
│   │       │   └── dashboard.yml
│   │       └── datasources
│   │           └── datasource.yml
│   ├── nginx
│   │   └── default.conf
│   ├── prometheus.yml
│   └── rules.yml
├── data
│   ├── alertmanager
│   ├── grafana
│   └── prometheus
├── docker-compose.yml
└── .env
```

nào mình cùng bắt đầu thôi.

Cài đặt Docker

Docker là nền tảng cung cấp cho các công cụ, service để các lập trình viên và người quản trị hệ thống có thể phát triển, thực thi, chạy các ứng dụng với containers. Việc sử dụng các **Linux containers** để triển khai ứng dụng được gọi là **containerization**, sử dụng **container** giúp dễ dàng hơn trong việc triển khai ứng dụng.

Ngoài ra, mình còn sử dụng thêm **Docker Compose**, là công cụ giúp định nghĩa và khởi chạy **multi-container Docker applications**. Với **Compose**, chúng ta sử dụng **Compose** file để cấu hình application's services. Chỉ với một câu lệnh, mình có thể dễ dàng create và start toàn bộ các services phục vụ cho việc chạy ứng dụng.

Cách cài đặt 2 công cụ trên các bạn có thể tham khảo trực tiếp trên trang chủ của Docker:

- <https://docs.docker.com/engine/install/ubuntu/>
- <https://docs.docker.com/compose/install/>

Cài đặt Prometheus

Prometheus là gì ?

Prometheus là một dịch vụ theo dõi và cảnh báo về hệ thống. Đây là một dịch vụ mã nguồn mở (Open source) hoàn toàn miễn phí. SoundCloud đã khởi đầu xây dựng **Prometheus** từ năm 2012. **Prometheus** đã được rất nhiều hệ thống tin tưởng áp dụng. Dự án có một cộng đồng người đóng góp, phát triển rất tích cực.

Giờ đây **Prometheus** đã được tách khỏi SoundCloud và là một dự án mã nguồn mở độc lập. Năm 2016, **Prometheus** tham gia vào tổ chức CNCF (Cloud Native Computing Foundation) với vị trí được ưu tiên phát triển thứ hai sau **K8s** (Kubernetes).



↑ +18 ↓



Tính năng quan trọng nhất của **Prometheus** là thu thập thông số, dữ liệu từ các mục tiêu (dịch vụ) được nhắm đến theo khoảng thời gian nhất định đã được cài đặt trước. Ngoài ra còn các API khác thể hiện được kết quả, đánh giá bằng biểu thức quy tắc và đưa ra cảnh báo. **Prometheus** còn cung cấp một ngôn ngữ truy vấn rất mạnh **PromQL**, cực kì hữu ích khi giao tiếp với các dịch vụ monitor khác.

Cài đặt

Các công cụ mình sử dụng đều đã được đóng gói thành các **Docker Image** và publish trên **Docker Hub** thế nên đơn giản chúng ta chỉ cần sử dụng chúng vào trong file `docker-compose.yml` của mình, bổ sung thêm config cho từng công cụ.

Đầu tiên với **Prometheus**, chúng ta có file config đặt ở `config/prometheus.yml` như sau:

```
config/prometheus.yml

rule_files:
  - "/etc/prometheus/rules.yml"

alerting:
  alertmanagers:
    - static_configs:
      - targets:
        - alertmanager:9093
scrape_configs:
- job_name: 'blackbox'
  scrape_interval: 10s
  metrics_path: /probe
  static_configs:
    - targets:
      - https://google.com
      - https://viblo.asia
      - https://viblo.asia/api/web-init-404
params:
  module: [http_2xx]
relabel_configs:
  - source_labels: [ __address__ ]
    target_label: __param_target
  - source_labels: [ __param_target ]
    target_label: instance
  - target_label: __address__
    replacement: "blackbox_exporter:9115"
```

Thuộc tính	Ghi chú
rule_file	Là mảng chứa thông tin là đường dẫn tới các điều kiện gửi cảnh báo của Alert Manager
alerting	Định nghĩa cấu hình của instance Alert Manager
scrape_configs	Blackbox Exporter sẽ trả ra các metrics liên quan tới website, trong section này, chúng ta cần cấu hình cách Prometheus lấy chúng về
scrape_configs.*.static_configs	Chứa các url mà chúng ta cần theo dõi
scrape_configs.*.metrics_path	Endpoint mà Blackbox Exporter trả về metrics

Thuộc tính	Ghi chú
scrape_configs.*.relabel_configs	Định nghĩa cho Prometheus viết lại các nhãn trước khi nó kéo metrics từ Blackbox về

Với `rule_files` là thông tin đường dẫn tới file cấu hình gửi cảnh báo của **Alert Manager**:

config/rules.yml

```
groups:
  - name: AllInstances
    rules:
      - alert: InstanceDown
        # Condition for alerting
        expr: up == 0
        for: 5s
        # Annotation - additional informational labels to store more information
        annotations:
          title: 'Instance {{ $labels.instance }} down'
          description: '{{ $labels.instance }} of job {{ $labels.job }} has been down for more than 5 seconds.'
        # Labels - additional labels to be attached to the alert
        labels:
          severity: 'critical'
```

Tiếp theo, chúng ta cần mount các file cấu hình trên vào service Prometheus trong Docker Compose nữa:

docker-compose.yml

```
version: '2.2'

networks:
  frontend:
    name: frontend

services:
  prometheus:
    image: prom/prometheus
    restart: always
    command:
      - '--config.file=/etc/prometheus/prometheus.yml'
      - '--web.enable-lifecycle'
    labels:
      - traefik.enable=true
      - traefik.http.routers.${COMPOSE_PROJECT_NAME}-prometheus.rule=Host(`prometheus`)
      - traefik.http.services.${COMPOSE_PROJECT_NAME}-prometheus.loadbalancer.server.port=9090
    networks:
      - default
    volumes:
      - ./data/prometheus:/prometheus
      - ./config/prometheus.yml:/etc/prometheus/prometheus.yml
      - ./config/rules.yml:/etc/prometheus/rules.yml
    ports:
      - 9090:9090
```

Cài đặt Blackbox Exporter

Blackbox Exporter là một chương trình exporter cho phép gọi tới các endpoint thông qua các giao thức *HTTP*, *HTTPS*, *DNS*, *TCP* và *ICMP*. Điều này cũng có nghĩa bạn có thể sử dụng nó để monitor các endpoint bất kì, có sử dụng

Với **Blackbox Exporter**, chúng ta cần cấu hình các module thực thi việc thăm dò tới các target được định nghĩa ở **Prometheus**. Do cơ chế **Service Discovery**, nên các target này được đặt ở **Prometheus** và ở **Blackbox Exporter** chúng ta chỉ cần cấu hình các module thôi

config/blackbox.yml

```
modules:
  http_2xx:
    http:
      fail_if_not_ssl: false
      ip_protocol_fallback: false
      method: GET
      no_follow_redirects: false
      preferred_ip_protocol: ip4
      valid_http_versions:
        - HTTP/1.1
        - HTTP/2.0
    prober: http
    timeout: 15s
```

Thuộc tính	Ghi chú
modules	Định nghĩa danh sách modules mà Blackbox Exporter sử dụng
http_2xx	Là tên của module
modules.<tên modules>.http	Cấu hình cho giao thức module này sử dụng, trong đó với giao thức HTTP ta có thể cấu hình method, điều kiện fail khi target không có SSL, cách resolve target,...
modules.<tên modules>.prober	Prober thực thi việc thăm dò
modules.<tên modules>.timeout	Thời gian timeout của mỗi lần thăm dò

Tiếp theo chúng ta bổ sung đoạn dưới đây vào file `docker-compose.yml` mà chúng ta đang xây dựng:

docker-compose.yml

```
....
blackbox_exporter:
  image: prom/blackbox-exporter
  restart: always
  tty: true
  labels:
    - traefik.enable=true
    - traefik.http.routers.${COMPOSE_PROJECT_NAME}-backbox.rule=Host(`blackbox_exporter`)
    - traefik.http.services.${COMPOSE_PROJECT_NAME}-backbox.loadbalancer.server.port=9115
  ports:
    - 9115:9115
  dns: 8.8.8.8
  command: --config.file=/etc/blackbox/blackbox.yml
  volumes:
    - ./config/blackbox:/etc/blackbox/
  networks:
    - default
```



Cài đặt Alert Manager

Alert manager sẽ xử lý các cảnh báo gửi từ ứng dụng **Prometheus**. Thông qua các rules mà ta định nghĩa ở file cấu hình **Prometheus**. **Alert Manager** sẽ hoạt động theo:

- **Prometheus** sẽ thu thập dữ liệu monitor là metrics từ các endpoint và lưu trong database **Prometheus**.
- Bạn tạo các rule alert trong **Prometheus**, ví dụ như cấu hình của mình ở trên thì: Bất cứ 1 instance nào có trạng thái `up == 0` trong 5s, endpoint down trong 5s.
- Và **Prometheus** đẩy các alert này về **Alert manager**, nó sẽ xử lý việc trùng lặp alert, group các alert và gửi alert tới kênh thông báo mà bạn muốn: *mail, telegram, slack,...* Nội dung thông báo bạn có thể cấu hình như dưới đây trong file `config/alertmanager.yml`

```
config/alertmanager.yml

global:
  resolve_timeout: 1m
  slack_api_url: 'https://hooks.slack.com/services/xxx'

route:
  receiver: 'slack-notifications'

receivers:
- name: 'slack-notifications'
  slack_configs:
    - channel: 'test'
      send_resolved: true
      icon_url: https://avatars3.githubusercontent.com/u/3380462
      title: |-
        [{{ .Status | toUpper }}]{{ if eq .Status "firing" }}:{{ .Alerts.Firing | len }}{{ end }}] [{{ .CommonLabels
        {{- if gt (len .CommonLabels) (len .GroupLabels) -}}
          {{" "}}(
          {{- with .CommonLabels.Remove .GroupLabels.Names }}
            {{- range $index, $label := .SortedPairs -}}
              {{ if $index }}, {{ end }}
              {{- $label.Name }}="{{ $label.Value -}}"
            {{- end }}
          {{- end -}}
        )
        {{- end }}
      text: >-
        {{ range .Alerts -}}
        *Alert:* [{{ .Annotations.title }}]{{ if .Labels.severity }} - `[{{ .Labels.severity }}`]{{ end }}

        *Description:* [{{ .Annotations.description }}]

        *Details:*
          {{ range .Labels.SortedPairs }} • *{{ .Name }}*: `{{ .Value }}`
          {{ end }}
        {{ end }}
```



↑ +18 ↓



```
docker-compose.yml

...
alertmanager:
  image: prom/alertmanager
  restart: always
  labels:
    - traefik.enable=true
    - traefik.http.routers.${COMPOSE_PROJECT_NAME}-alertmanager.rule=Host(`alertmanager`)
    - traefik.http.services.${COMPOSE_PROJECT_NAME}-alertmanager.loadbalancer.server.port=9093
  ports:
    - 9093:9093
  volumes:
    - ./data/alertmanager:/etc/alertmanager/
```

Với cách cấu hình như trên thì mỗi khi có downtime, 1 alert sẽ được gửi tới **Slack channel**



Cài đặt Grafana

Cuối cùng để visualize các metrics của các website, chúng ta sử dụng **Grafana**. **Grafana** là một giao diện/dashboard theo dõi hệ thống (opensource), hỗ trợ rất nhiều loại dashboard và các loại graph khác nhau để người quản trị dễ dàng theo dõi.

Grafana là một nền tảng open-source chuyên phục vụ mục đích theo dõi và đánh giá các số liệu thu được. Grafana có thể truy xuất dữ liệu từ **Graphite, Elasticsearch, OpenTSDB, Prometheus và InfluxDB**. **Grafana** là một công cụ mạnh mẽ để truy xuất và biểu diễn dữ liệu dưới dạng các đồ thị và biểu đồ.

Mình sẽ cấu hình cho **Grafana** lấy data từ *datasource* là **Prometheus**

```
config/grafana/datasources/datasource.yml

apiVersion: 1

datasources:
  - name: Prometheus
    type: prometheus
    access: proxy
    orgId: 1
    url: http://prometheus:9090
    basicAuth: false
    isDefault: true
    editable: true
```



↑ +18 ↓



Trong thư mục config của **Grafana**, ngoài cấu hình datasource còn có cấu hình dashboard và 1 file dashboard mình đã export sẵn ra, bạn có thể sử dụng các Dashboard có sẵn từ **Grafana** cũng được. Tiếp theo là đến cấu hình của service **Grafana** trong `docker-compose.yml`

```
docker-compose.yml

...
grafana:
  image: grafana/grafana-enterprise
  restart: always
  tty: true
  labels:
    - traefik.enable=true
    - traefik.http.routers.${COMPOSE_PROJECT_NAME}-grafana.rule=Host(`grafana`)
    - traefik.http.services.${COMPOSE_PROJECT_NAME}-grafana.loadbalancer.server.port=3000
  ports:
    - ${PORT:-80}:3000
  volumes:
    - ./data/grafana:/var/lib/grafana
    - ./config/grafana/provisioning:/etc/grafana/provisioning
  networks:
    - frontend
    - default
```

Test thử thôi

Sau các bước ở trên, chúng ta có 1 file `docker-compose.yml` hoàn chỉnh sẽ trông như này

```
version: '2.2'

networks:
  frontend:
    name: frontend

services:
  traefik:
    image: traefik:2.2
    restart: always
    command:
      - --api.dashboard=true
      - --entrypoints.http.address=:80
      - --providers.docker
      - --providers.docker.network=frontend
      - --providers.docker.watch=true
      - --providers.docker.exposedbydefault=false
    ports:
      - 8080:80
    networks:
      - frontend
    labels:
      - traefik.enable=true
      - traefik.http.routers.${COMPOSE_PROJECT_NAME}-traefik.rule=Host(`traefik.${DOMAIN}`)
      - traefik.http.routers.${COMPOSE_PROJECT_NAME}-traefik.service=api@internal
      - traefik.http.middlewares.${COMPOSE_PROJECT_NAME}-traefik-dashboard.redirectRegex.regex=/
      - traefik.http.middlewares.${COMPOSE_PROJECT_NAME}-traefik-dashboard.redirectRegex.replacement=/dashboard/
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock

  prometheus:
    image: prom/prometheus
    restart: always
    command:
      - '--config.file=/etc/prometheus/prometheus.yml'
      - '--web.enable-lifecycle'
    labels:
      - traefik.enable=true
      - traefik.http.routers.${COMPOSE_PROJECT_NAME}-prometheus.rule=Host(`prometheus`)
      - traefik.http.services.${COMPOSE_PROJECT_NAME}-prometheus.loadbalancer.server.port=9090
    networks:
      - default
    volumes:
      - ./data/prometheus:/prometheus
      - ./config/prometheus.yml:/etc/prometheus/prometheus.yml
      - ./config/rules.yml:/etc/prometheus/rules.yml
    ports:
      - 9090:9090

  alertmanager:
    image: prom/alertmanager
    restart: always
    labels:
      - traefik.enable=true
      - traefik.http.routers.${COMPOSE_PROJECT_NAME}-alertmanager.rule=Host(`alertmanager`)
      - traefik.http.services.${COMPOSE_PROJECT_NAME}-alertmanager.loadbalancer.server.port=9093
    ports:
      - 9093:9093
    volumes:
      - ./data/alertmanager:/etc/alertmanager/
      - ./config/alertmanager.yml:/etc/alertmanager/alertmanager.yml
    networks:
      - default
    depends_on:
      - prometheus
```



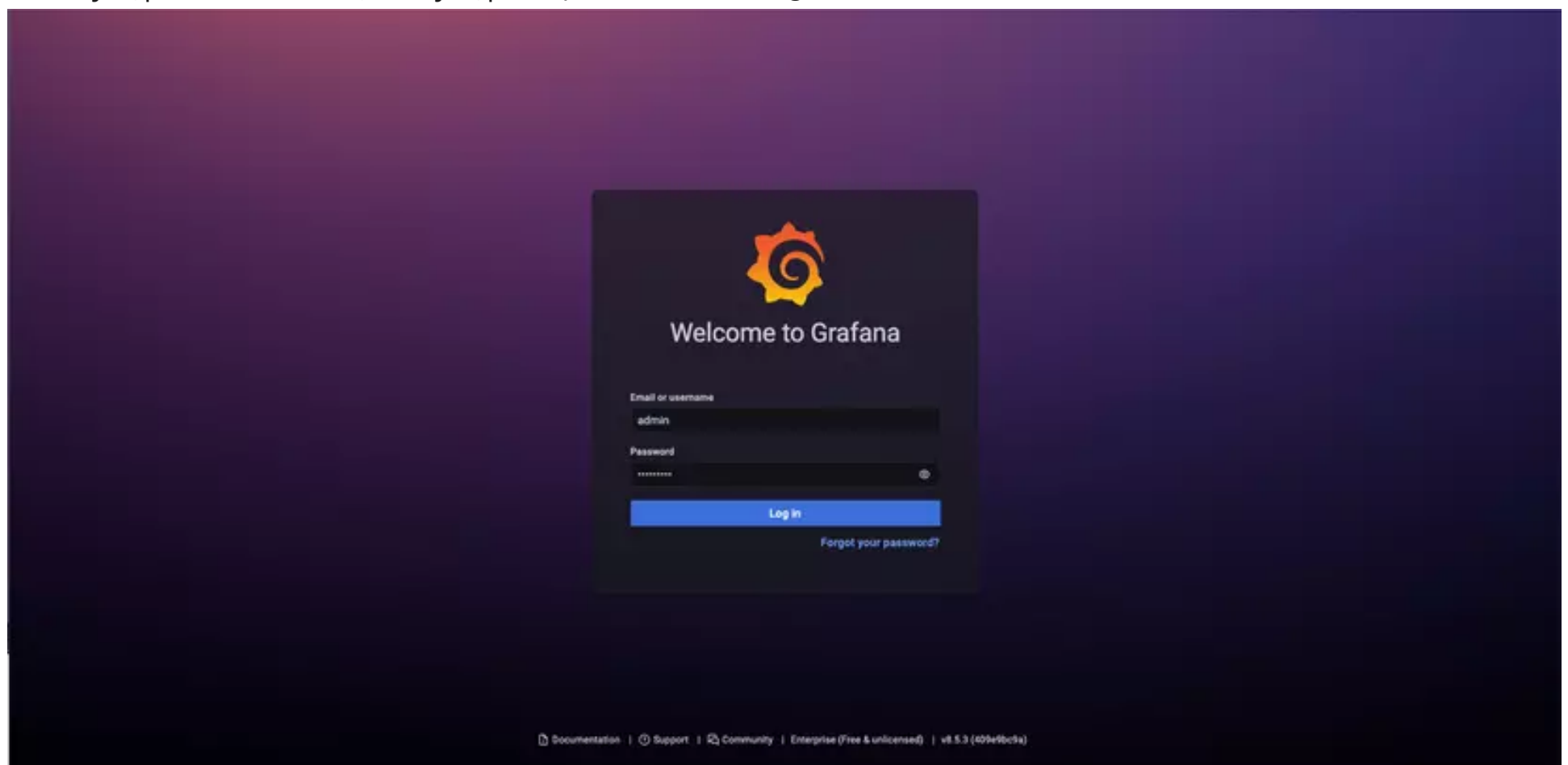
```
restart: always
tty: true
labels:
  - traefik.enable=true
  - traefik.http.routers.${COMPOSE_PROJECT_NAME}-blackbox.rule=Host(`blackbox_exporter`)
  - traefik.http.services.${COMPOSE_PROJECT_NAME}-blackbox.loadbalancer.server.port=9115
ports:
  - 9115:9115
dns: 8.8.8.8
command: --config.file=/etc/blackbox/blackbox.yml
volumes:
  - ./config/blackbox:/etc/blackbox/
networks:
  - default

grafana:
  image: grafana/grafana-enterprise
  restart: always
  tty: true
  labels:
    - traefik.enable=true
    - traefik.http.routers.${COMPOSE_PROJECT_NAME}-grafana.rule=Host(`grafana`)
    - traefik.http.services.${COMPOSE_PROJECT_NAME}-grafana.loadbalancer.server.port=3000
  ports:
    - ${PORT:-80}:3000
  volumes:
    - ./data/grafana:/var/lib/grafana
    - ./config/grafana/provisioning:/etc/grafana/provisioning
  networks:
    - frontend
    - default
```

Ta chỉ cần chạy command cuối cùng là

```
$ docker-compose up -d
```

Để truy cập Grafana các bạn truy cập <http://localhost/login>



tài khoản mặc định là admin/admin. Sau khi đăng nhập bước đầu tiên là bạn cần đổi mật khẩu mặc định sang 1 mật khẩu an toàn hơn.



↑ +18 ↓



Từ giao diện Grafana, bạn chọn Dashboard > Home > Prometheus Blackbox Exporter là sẽ thấy được các dashboard hiển thị thông tin liên quan tới website, cái mà Prometheus đã thu thập được



Mỗi instance được thu thập các thông tin trạng thái Up/Down, thời gian phản hồi của trang, trạng thái của SSL,... bạn có thể tùy ý thêm các Panel, ngoài ra bạn có thể share dashboard để các thành viên khác cũng có thể theo dõi trạng thái của website cùng.

Tài liệu tham khảo

- <https://github.com/prometheus>
- <https://grafana.com/>
- [Project của mình](#)

Prometheus PHP Ruby

All rights reserved



Bài viết liên quan

[Sử dụng Charles Proxy để theo dõi network traffic trên...](#)

[Nguyễn Thanh Hưng](#)
5 phút đọc
10.5K 9 3 9

[Vòng đời và trạng thái của Thread](#)

[Văn Phúc](#)
9 phút đọc
1.8K 7 2 10

[Các mã nguồn mở tốt nhất để làm website bán hàng...](#)

[Phạm Minh Hường](#)
24 phút đọc
9.7K 4 0 1

[Build dashbo datasource l](#)

[Duong Trung Hie](#)
4 phút đọc
1.7K 4 1

[Sử dụng Charles Proxy để theo dõi network traffic trên Android](#)

[Nguyễn Thanh Hưng](#)
5 phút đọc
10.5K 9 3 9

[Vòng đời và trạng thái của Thread](#)

[Văn Phúc](#)
9 phút đọc
1.8K 7 2 10



+18



Phạm Minh Hường

24 phút đọc

👁 9.7K

🔖 4

💬 0

👍 1

Duong Trung Hieu

4 phút đọc

👁 1.7K

🔖 4

💬 1

👍 9

Bài viết khác từ Dao Thai Son

Cấu hình Traefik Ingress Controller cho cụm k3s

Dao Thai Son

5 phút đọc

👁 267

🔖 3

💬 2

👍 15

Chỉ với vài click dễ dàng có ngay công cụ theo dõi trạng...

Dao Thai Son

7 phút đọc

👁 971

🔖 6

💬 5

👍 11

Giới thiệu về Metabase - công cụ hỗ trợ phân tích cơ...

Dao Thai Son

3 phút đọc

👁 3.4K

🔖 3

💬 0

👍 9

Tìm hiểu về S (phần tiếp th...

Dao Thai Son

8 phút đọc

👁 3.4K

🔖 4

💬 1

👍 17

Cấu hình Traefik Ingress Controller cho cụm k3s

Dao Thai Son

5 phút đọc

👁 267

🔖 3

💬 2

👍 15

Chỉ với vài click dễ dàng có ngay công cụ theo dõi trạng thái uptime website với Uptime Kuma và Heroku

Dao Thai Son

7 phút đọc

👁 971

🔖 6

💬 5

👍 11

Giới thiệu về Metabase - công cụ hỗ trợ phân tích cơ sở dữ liệu

Dao Thai Son

3 phút đọc

👁 3.4K

🔖 3

💬 0

👍 9

Tìm hiểu về Service Mesh (phần tiếp theo).

Dao Thai Son

8 phút đọc

👁 3.4K


🔖 4

💬 3

👍 17

Bình luận

🗨 Đăng nhập để bình luận




Ted Minh .@teddy_kul

thg 6 12, 2022 2:48 CH

🔔

Bình luận này đã bị xóa



Trương Văn Huy .@huytvomi

thg 6 14, 2022 11:43 SA


Bài viết rất bổ ích. Tks thớt

👍 +1

👎

 | [Trả lời](#) [Chia sẻ](#)

⋮



Quach Thanh Tung .@chuoivanxanh

thg 6 27, 2022 9:43 SA

Hi,

Nếu được bạn bổ sung thêm giúp phần tình huống nếu thêm Exporter khác thì cần sửa đổi ra sao (ví dụ Node Exporter).

Cám ơn bạn nhiều

👍 0

👎 0

 | [Trả lời](#) [Chia sẻ](#)

⋮



👍 +18 👎





Jin @minhhungit
thg 4 26, 4:39 CH


Bài viết rất hữu ích khi demo stack grafana-prometheus! Nhưng nếu chỉ check uptime thì lại hơi rườm rà, trường hợp này thì chỉ cần chạy docker kuma là chiến tốt 😊

^ 0 v | [Trả lời](#) [Chia sẻ](#) ...

TÀI NGUYÊN

- [Bài viết](#)
- [Câu hỏi](#)
- [Videos](#)
- [Thảo luận](#)
- [Công cụ](#)
- [Trạng thái hệ thống](#)
- [Tổ chức](#)
- [Tags](#)
- [Tác giả](#)
- [Đề xuất hệ thống](#)
- [Machine Learning](#)

DỊCH VỤ

-  [Viblo Code](#)
-  [Viblo CV](#)
-  [Viblo CTF](#)
-  [Viblo Learning](#)
-  [Viblo Interview](#)

ỨNG DỤNG DI ĐỘNG



LIÊN KẾT



↑ +18 ↓

