

Homework – DevSecOps

Javier Augustson

Buat satu proses ci/cd yang menerapkan proses devsecops didalamnya dan jelaskan step by step.

Jawab :

CI/CD dilakukan di github menggunakan github action. GitHub Actions adalah fitur dari GitHub yang memungkinkan untuk **mengotomatisasi** tugas seperti **CI/CD (Continuous Integration & Continuous Deployment)**, testing, deployment, dan lain-lain langsung di dalam repository GitHub.

Membuat CI/CD pada github

GitHub Actions bekerja berdasarkan **workflow** yang didefinisikan dalam file **YAML** di dalam folder `.github/workflows/`.

1. Buka atau siapkan repository GitHub yang diinginkan.

Buka terminal wsl yang sudah terhubung dengan akun github tersebut. Pastikan sudah memiliki repository kerja untuk ditambahkan github action. Clone repository tersebut ke wsl untuk mempermudah edit repository.

Dalam tugas ini, disiapkan repository baru bernama scan-gitleaks. Repo ini berisi satu Dockerfile menggunakan image dasar nginx. Clone repository ini ke wsl :

Git clone [git@github.com:javiersosial/scan-gitleaks.git](https://github.com:javiersosial/scan-gitleaks.git)

```
javiers@DESKTOP-EH0ID9D:~/github$ cd scan-gitleaks/  
javiers@DESKTOP-EH0ID9D:~/github/scan-gitleaks$ ls  
Dockerfile
```

2. Buat folder baru bernama `.github/workflows/`.

Folder bernama `.github/workflows` adalah direktori khusus tempat github action bekerja. Folder ini bersifat hidden karena menggunakan titik di depan nama folder. Untuk melihat folder ini di wsl digunakan perintah `ls -la`. Buat direktori ini di repository scan-gitleaks tersebut melalui wsl :

`mkdir -p .github/workflows`

```
javiers@DESKTOP-EH0ID9D:~/github/scan-gitleaks$ ls -la  
total 20  
drwxr-xr-x 4 javiers javiers 4096 Feb 18 18:11 .  
drwxr-xr-x 4 javiers javiers 4096 Feb 18 17:34 ..  
drwxr-xr-x 8 javiers javiers 4096 Feb 19 16:42 .git  
drwxr-xr-x 3 javiers javiers 4096 Feb 18 18:11 .github  
-rw-r--r-- 1 javiers javiers 469 Feb 18 17:34 Dockerfile
```

3. Buat file workflow di dalamnya, misalnya docker-build-push.yml. File workflow ini adalah file .yaml berisi perintah CI/CD yang akan dijalankan oleh Github Action.

Setelah membuat folder kerja github action, kita dapat membuat file workflow pada folder tersebut yang akan dijalankan github action. Pada tugas ini dibuat file workflow yang berisi tahapan CI/CD untuk membuat image Docker kemudian push ke repo di Docker Hub.

Sebelum membuat file workflow tersebut, Github memerlukan secrets berupa username dan password dari akun DockerHub yang dituju agar Github action bisa melakukan push ke repo Docker hub tersebut. Secrets tersebut disimpan di repository kerja yang sama tempat Github action dijalankan.

Tambahkan secrets di repository GitHub :

Buka Github Repo > Settings > Secrets and variables > Actions.

1

2

General

Access

Collaborators

Moderation options

Code and automation

Branches

Tags

Rules

Actions

Webhooks

Environments

Codespaces

Pages

Security

Code security

Deploy keys

Secrets and variables

Actions

Actions secrets and variables

Secrets and variables allow you to manage reusable configuration data. Secrets are **encrypted** and are used for sensitive data. [Learn more about encrypted secrets](#). Variables are shown as plain text and are used for **non-sensitive** data. [Learn more about variables](#).

Anyone with collaborator access to this repository can use these secrets and variables for actions. They are not passed to workflows that are triggered by a pull request from a fork.

Secrets Variables

Environment secrets

This environment has no secrets.

Manage environment secrets

Repository secrets

4

New repository secret

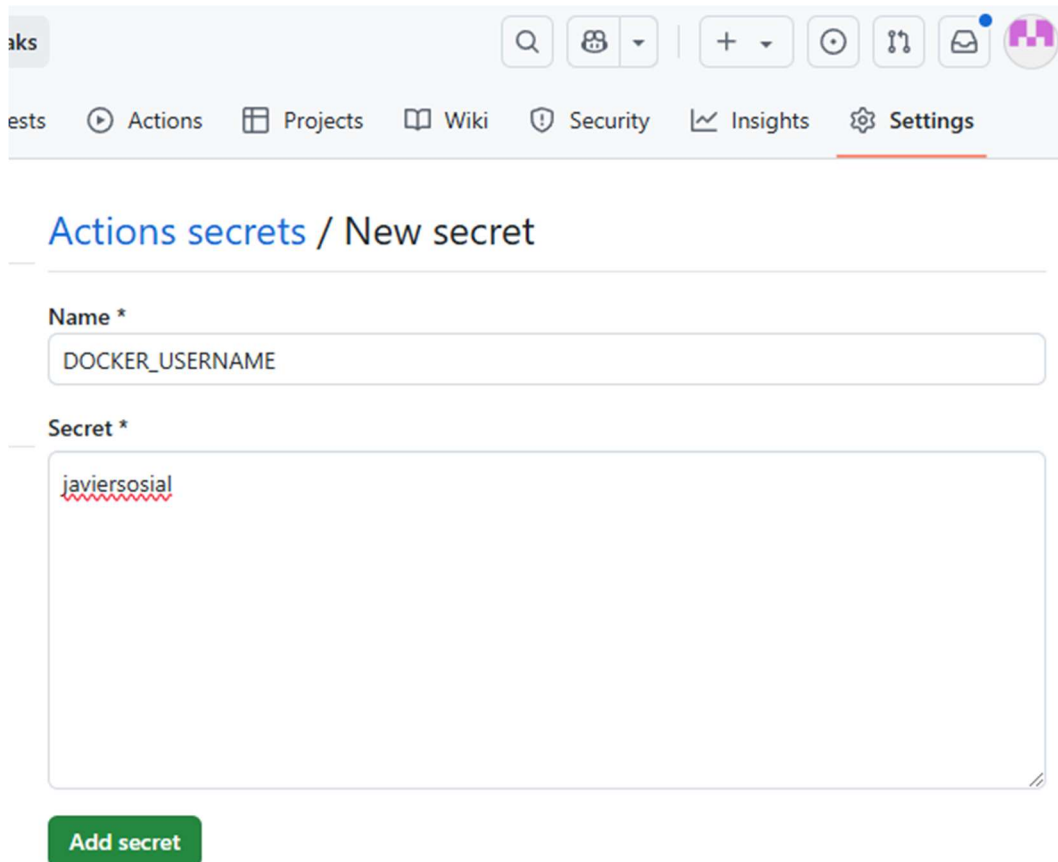
Name	Last updated
DOCKER_PASSWORD	1 hour ago
DOCKER_USERNAME	1 hour ago

3

Klik New repository secret dan tambahkan :

DOCKER_USERNAME = Username Docker Hub

DOCKER_PASSWORD = Password Docker Hub



The screenshot shows the GitHub Actions interface. At the top, there's a navigation bar with icons for search, repository, add, workflow, settings, and profile. Below this is a secondary navigation bar with links for 'ests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. The main heading is 'Actions secrets / New secret'. There are two input fields: 'Name *' with the value 'DOCKER_USERNAME' and 'Secret *' with the value 'javiersosial'. A green 'Add secret' button is at the bottom.

aks

ests Actions Projects Wiki Security Insights Settings

Actions secrets / New secret

Name *

DOCKER_USERNAME

Secret *

javiersosial

Add secret

Setelah secrets DOCKER_USERNAME dan DOCKER_PASSWORD telah disimpan, kita lanjutkan dengan membuat file workflow .yaml di direktori .github/workflows pada repository tersebut. File ini merupakan file Workflow untuk pembuatan image Docker dan push ke Docker Hub.

```

🔗 docker-build-push.yml
1  name: Docker Build and Push
2
3  on:
4    push:
5      branches: [main] # Ganti dengan branch yang sesuai
6
7  jobs:
8    build-and-push:
9      runs-on: ubuntu-latest
10     steps:
11       - name: Checkout kode
12         uses: actions/checkout@v3
13
14       - name: Login ke Docker Hub
15         uses: docker/login-action@v2
16         with:
17           username: ${ secrets.DOCKER_USERNAME }
18           password: ${ secrets.DOCKER_PASSWORD }
19
20       - name: Build image Docker
21         run: docker build -t javiersosial/hello-world-gitaction:1.0 . # Ganti de
22
23       - name: Push image Docker ke Docker Hub
24         run: docker push javiersosial/hello-world-gitaction:1.0

```

Penjelasan konfigurasi file workflow

- **name:** Nama workflow yang akan ditampilkan di GitHub Actions.
- **on:** Memicu workflow ketika ada *push* ke branch main. Anda bisa menggantinya dengan branch lain yang sesuai.
- **jobs:** Kumpulan tugas yang akan dijalankan.
- **runs-on:** Menentukan sistem operasi yang akan digunakan untuk menjalankan tugas.
- **steps:** Daftar langkah-langkah yang akan dieksekusi dalam tugas.
 - `actions/checkout@v3`: Mengambil kode dari repositori GitHub.
 - `docker/login-action@v2`: Login ke Docker Hub menggunakan kredensial yang disimpan di *secrets* GitHub.
 - `docker build`: Membangun image Docker. Ganti `<nama-image>:<tag>` dengan nama dan tag image yang sesuai.
 - `docker push`: Mendorong image Docker ke Docker Hub.

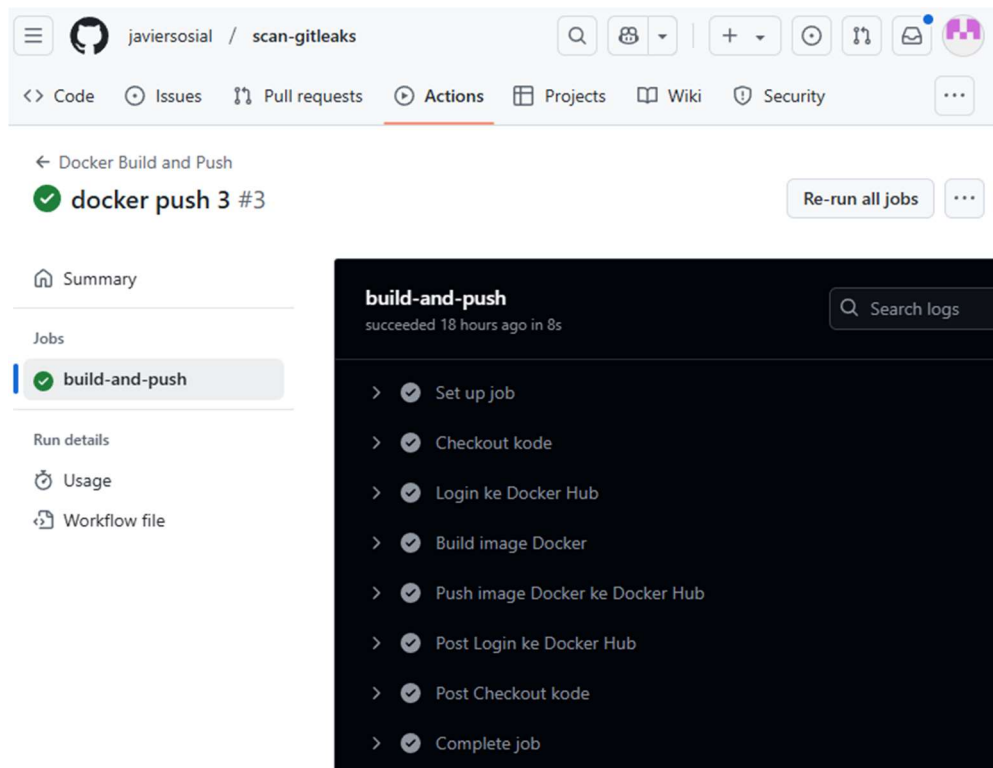
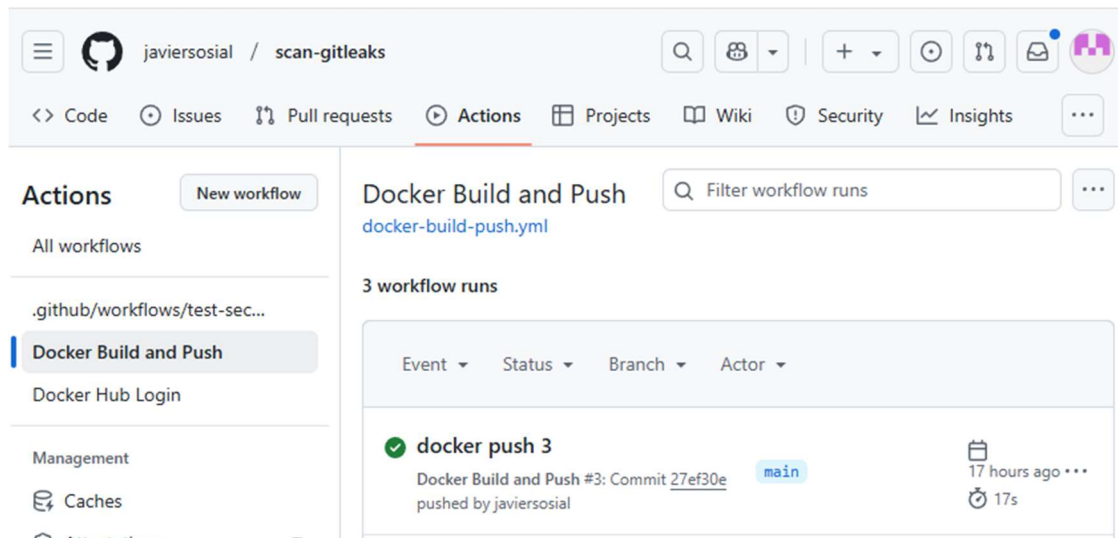
Setelah menambahkan konfigurasi workflow, simpan file `docker-build-push.yml` dan lakukan commit serta push ke repositori GitHub.

Setelah Anda melakukan push ke branch yang ditentukan di konfigurasi workflow, GitHub Actions akan otomatis menjalankan workflow tersebut.

Workflow juga bisa dijalankan secara manual melalui tab "Actions" di repositori GitHub dengan cara :

- Buka tab "Actions" di repositori GitHub.
- Pilih workflow "Docker Hub Login".
- Klik tombol "Run workflow".

Untuk melihat hasil dari workflow apakah berhasil atau tidak juga dapat dilihat melalui tab actions pada repository tersebut. Pilih Action yang ingin dilihat hasilnya. Jika berhasil akan terlihat status centang hijau dan status sukses.



Terlihat detail tahapan CI/CD telah berjalan semua. Job build and push sudah bertanda centang hijau. Ini menandakan CI/CD menggunakan github action telah berhasil.

Menambahkan security dalam proses CI/CD untuk menerapkan devsecops dan dijelaskan step by step.

Tambahan metode security yang digunakan adalah **Gitleaks** dan **Privy** untuk menerapkan DevSecOps.

Pada CI/CD yang sudah dibuat di atas, akan ditambahkan gitleaks. **Gitleaks** adalah tool keamanan sumber terbuka yang digunakan untuk mendeteksi kebocoran informasi sensitif di dalam repository Git. Tool ini berguna untuk menemukan API keys, password, credential, atau secret lainnya yang mungkin tidak sengaja ter-commit ke dalam repository.

❖ Kenapa Gitleaks penting?

- ✓ Mencegah kebocoran data sebelum masuk ke repository publik/private.
- ✓ Bisa digunakan dalam CI/CD pipeline untuk scanning otomatis.
- ✓ Cepat dan mudah digunakan untuk memeriksa seluruh riwayat Git.

Gitleaks adalah metode security untuk memastikan tidak ada secrets yang tertulis di kode repositori. Gitleaks ditambahkan pada awal tahapan CI/CD. Jika Gitleaks gagal, berarti ada kebocoran data seperti secrets, maka tahapan selanjutnya seperti build dan push image tidak akan dilanjutkan. Berikut adalah file workflow yang sudah ditambahkan Gitleaks :

```
gitleaks-docker-build-push.yml
1  name: Gitleaks Docker Build and Push
2
3  on:
4  push:
5    branches: [main] # Ganti dengan branch yang sesuai
6
7  jobs:
8    build-and-push:
9      runs-on: ubuntu-latest
10     steps:
11       - name: Checkout kode
12         uses: actions/checkout@v3
13
14       - name: Install dan Jalankan Gitleaks
15         uses: gitleaks/gitleaks-action@v2
16         with:
17           config-path: .github/gitleaks.toml # Opsional, jika ingin cu
18           fail: true # Gagal jika ada kebocoran secrets
19           verbose: true
20
21       - name: Login ke Docker Hub
22         uses: docker/login-action@v2
23         with:
24           username: ${ secrets.DOCKER_USERNAME }}
25           password: ${ secrets.DOCKER_PASSWORD }}
26
27       - name: Build image Docker
28         run: docker build -t javiersosial/hello-world-gitaction:1.0 .
29
30       - name: Push image Docker ke Docker Hub
31         run: docker push javiersosial/hello-world-gitaction:1.0
32
```

Penjelasan

1. GitLeaks akan mengecek repo sebelum build Docker.
2. Jika ada kebocoran secrets, pipeline akan gagal (fail: true).
3. Bisa menambahkan konfigurasi GitLeaks sendiri dengan file `.github/gitleaks.toml`.

Dengan ini, accidental leaks di repo GitHub bisa dicegah sebelum masuk ke Docker Hub. Lakukan commit dan push ke github untuk menjalankan action tersebut.

Buka Github Action untuk melihat hasil action dari workflow yang baru saja di commit.

The screenshot shows the GitHub Actions interface for the repository 'javiersosial / scan-gitleaks'. The workflow 'Gitleaks Docker Build and Push' is displayed, with a specific run titled 'menambahkan gitleaks pada docker build push #2' that has completed successfully. The summary section shows the workflow was triggered by a push 1 hour ago, with a status of 'Success', a total duration of 22s, and 1 artifact. The job 'build-and-push' is highlighted, showing it completed in 12s. The job summary indicates 'No leaks detected' with a green checkmark icon. The workflow file 'gitleaks-docker-build-push.yml' is shown as being triggered on a push event.

Pada github action dapat terlihat workflow CI/CD tersebut sukses dan tidak terdapat leak atau kebocoran. Dengan ini berarti Gitleaks berhasil melakukan scan pada repo tersebut. Tahapan Gitleaks tersebut sukses karena no leaks detected. Setelah tahapan Gitleaks sukses maka proses CI/CD dilanjutkan sesuai workflow yaitu dilanjutkan dengan tahapan build dan push ke Docker Hub.

Image baru berhasil dipush ke repo DockerHub melalui tahapan CI/CD workflow Github Action.

Name	Last Pushed	Contains	Visibility
javiersosial/hello-world-gitaction	about 1 hour ago	IMAGE	Public

Kemudian digunakan juga **Trivy** sebagai metode security dalam menerapkan DevSecOps. **Trivy** adalah open-source vulnerability scanner yang digunakan untuk mendeteksi kerentanan keamanan (vulnerabilities) dalam container images, kode sumber, dan dependency lainnya.

🔑 Kenapa Trivy Penting?

- ✓ Mendeteksi Kerentanan CVE (Common Vulnerabilities and Exposures).
- ✓ Cepat dan Mudah Digunakan → Bisa langsung digunakan tanpa konfigurasi rumit.
- ✓ Mendukung Berbagai Platform → Bisa scan Docker image, Kubernetes, Git repositories, file system, dan cloud infrastructure (AWS, GCP, Azure).
- ✓ Terintegrasi dengan CI/CD → Bisa digunakan di GitHub Actions, Jenkins, GitLab CI/CD, dll.

🔧 Cara Kerja Trivy

1. **Mengumpulkan Data** → Mengambil metadata dari image atau sistem yang dipindai.
2. **Menganalisis Vulnerability** → Mencocokkan data dengan database CVE.
3. **Menampilkan Hasil** → Laporan berupa **severity level**:
 - **CRITICAL** → Harus segera diperbaiki.
 - **HIGH** → Berisiko tinggi.
 - **MEDIUM** → Bisa berdampak, tapi tidak mendesak.
 - **LOW** → Minor vulnerability.

Pada workflow ini Trivy akan melakukan scan image hasil perintah docker build. Jika scan image oleh Trivy sukses, yang berarti image aman dari kebocoran secrets dan low vulnerability, maka image akan di push ke docker hub. Jika hasil scan Trivy menyatakan image tidak aman, maka tahapan CI/CD gagal dan image tidak akan di push ke docker hub.

Berikut adalah workflow **GitHub Actions** yang menjalankan tahapan **GitLeaks** → **Build Docker Image** → **Scan dengan Trivy** → **Push ke Docker Hub** hanya jika semua tahap sukses.

```
name: Gitleaks-Docker Build-Trivy & Push

on:
  push:
    branches: [main]

jobs:
  # 🔍 1 Scan kode sumber dengan GitLeaks
  secret-scan:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout kode
        uses: actions/checkout@v3

      - name: Install dan Jalankan GitLeaks
        uses: gitleaks/gitleaks-action@v2
        with:
          fail: true
          verbose: true
```


🛠️ 2 Build Docker Image jika kode bersih

build-image:

runs-on: ubuntu-latest

needs: secret-scan # Build hanya berjalan jika GitLeaks sukses

steps:

- name: Checkout kode

uses: actions/checkout@v3

- name: Login ke Docker Hub

uses: docker/login-action@v2

with:

username: \${ secrets.DOCKER_USERNAME }

password: \${ secrets.DOCKER_PASSWORD }

- name: Build image Docker

run: docker build -t javiersosial/hello-world-gitaction:1.0 .

- name: Save Docker Image ke File

run: docker save -o image.tar javiersosial/hello-world-gitaction:1.0

- name: Upload Docker Image untuk Scan

uses: actions/upload-artifact@v4

with:

name: docker-image

path: image.tar

🕒 3 Scan Image dengan Trivy sebelum push ke Docker Hub

scan-image:

runs-on: ubuntu-latest

needs: build-image # Scan hanya berjalan jika build sukses

steps:

- name: Download Docker Image

uses: actions/download-artifact@v4

with:

name: docker-image

- name: Install Trivy

run: |

sudo apt-get install -y curl gnupg

curl -fsSL https://aquasecurity.github.io/trivy-repo/deb/public.key | sudo tee /etc/apt/trusted.gpg.d/trivy.asc > /dev/null

echo "deb https://aquasecurity.github.io/trivy-repo/deb \$(lsb_release -sc) main" | sudo tee -a /etc/apt/sources.list.d/trivy.list

sudo apt-get update

sudo apt-get install -y trivy

- name: Load Docker Image

run: docker load -i image.tar

- name: Scan Docker Image dengan Trivy

```
run: trivy image --exit-code 1 --severity CRITICAL,HIGH javiersosial/hello-world-  
gitaction:1.0
```

🚀 4 Push Image ke Docker Hub jika Scan Trivy berhasil

push-image:

runs-on: ubuntu-latest

needs: scan-image # Push hanya berjalan jika Trivy scan sukses

steps:

- name: Download Docker Image

uses: actions/download-artifact@v4

with:

name: docker-image

- name: Load Docker Image

run: docker load -i image.tar

- name: Push image Docker ke Docker Hub

run: docker push javiersosial/hello-world-gitaction:1.0

🔗 Bagaimana Workflow Ini Bekerja? Workflow terdiri dari 4 Jobs :

1 Secret Scan dengan GitLeaks

- GitLeaks akan mendeteksi kebocoran secrets dalam kode.
- Jika ditemukan secrets yang bocor, pipeline akan gagal dan berhenti di sini.

2 Build Docker Image

- Hanya berjalan jika GitLeaks sukses.
- Docker image dibuat lalu disimpan sebagai file **image.tar** untuk digunakan di job berikutnya.

3 Scan Image dengan Trivy

- Trivy memindai image dari file image.tar yang sudah dibuat sebelumnya.
- Scan akan gagal jika ada kerentanan tingkat HIGH atau CRITICAL.
- Jika Trivy gagal, pipeline tidak akan lanjut ke tahap push.

4 Push Image ke Docker Hub

- Hanya berjalan jika GitLeaks dan Trivy sukses.
- Docker image diambil dari file image.tar dan di-push ke Docker Hub.

🔗 Keuntungan Workflow Ini

- ✓ Mencegah secrets bocor sebelum build Docker image.
- ✓ Docker image hanya dibuat jika kode sudah aman.
- ✓ Trivy memastikan image bebas dari kerentanan HIGH & CRITICAL sebelum di-push.
- ✓ CI/CD lebih aman dan hanya mengizinkan image yang lolos scan untuk di-deploy.

Dengan workflow ini, **hanya Docker image yang sudah aman yang akan masuk ke Docker Hub.**

Setelah workflow dijalankan melalui commit dan push, terjadi failure pada job scan image.

The screenshot shows the GitHub Actions workflow summary for 'gitleaks-docker-build-push.yml'. The workflow was triggered via push 1 hour ago by user 'javiersosial' on the 'main' branch. The status is 'Failure'. The total duration is 1m 17s, and there are 2 artifacts. The workflow consists of three jobs: 'secret-scan' (successful), 'build-image' (successful), and 'scan-image' (failed). The 'secret-scan' job summary shows 'No leaks detected' with a green checkmark.

Failure pada job scan image menandakan image yang di-build dan di-scan oleh Trivy dianggap tidak aman oleh Trivy sehingga proses workflow dihentikan.

Jika kita klik job scan image tersebut, akan terlihat detail penyebab failure yang terjadi.

The screenshot shows the details of the 'scan-image' job, which failed 1 hour ago in 36s. The job title is 'Scan Docker Image dengan Trivy'. The output shows the following information:

```
29
30 javiersosial/hello-world-gitaction:1.0 (debian 12.9)
31 =====
32 Total: 10 (HIGH: 8, CRITICAL: 2)
33
34
```

Berdasarkan detail tersebut, image yang dibuild memiliki 8 High issues dan 2 critical. Hal ini berarti kita harus mencoba mengganti image yang berasal dari Dockerfile pada repo tersebut.

Setelah mencari referensi online, akan dicoba mengganti base image. Pada Dockerfile awalnya digunakan Debian 12.9, akan diganti dengan image **Alpine**. Image Alpine biasanya lebih kecil dan memiliki lebih sedikit vulnerability.

Dockerfile dengan image Debian 12.9 yang memiliki High dan critical vulnerability

```
# Menggunakan image dasar Nginx dari Docker Hub
FROM nginx:latest

# Menyalin file konfigurasi Nginx custom ke dalam container (optional)
# Misalnya jika Anda ingin menambahkan konfigurasi khusus
# COPY nginx.conf /etc/nginx/nginx.conf

# Atau menyalin file HTML statis untuk ditampilkan oleh Nginx
# COPY ./html /usr/share/nginx/html

# Menampilkan port yang digunakan oleh Nginx
EXPOSE 80

# Menjalankan Nginx dalam mode foreground
CMD ["nginx", "-g", "daemon off;"]
```

Dockerfile dengan image Alpine :

```
# Menggunakan Nginx berbasis Alpine (lebih ringan)
FROM nginx:alpine

# Menyalin file konfigurasi Nginx custom (jika diperlukan)
# COPY nginx.conf /etc/nginx/nginx.conf

# Menyalin file HTML statis untuk ditampilkan oleh Nginx (jika ada)
# COPY ./html /usr/share/nginx/html

# Mengekspos port 80 untuk akses HTTP
EXPOSE 80

# Menjalankan Nginx dalam mode foreground
CMD ["nginx", "-g", "daemon off;"]
```

Setelah diganti image tersebut dan action dijalankan kembali, terlihat bahwa job scan-image berhasil dikerjakan. Ini berarti Image Alpine tersebut telah lulus scan Trivy dan dianggap aman untuk dilanjutkan.

Summary

Jobs

secret-scan

build-image

scan-image

push-image

Run details

Usage

Workflow file

Triggered via push 2 hours ago

Status

javiersosial pushed

b2940d7

main

Failure

Total duration

Artifacts

1m 9s

2

gitleaks-docker-build-push.yml

on: push

secret-scan

4s

build-image

secret-scan summary

...

No leaks detected

Pada tampilan di atas terlihat scan image telah berhasil berjalan yang berarti image aman untuk dipakai. Tetapi workflow kemudian gagal melanjutkan ke job push-image. Klik job push-image tersebut untuk melihat detail failurnya.

Jobs

secret-scan

build-image

scan-image

push-image

Run details

Usage

Workflow file

1 error

push-image

failed 1 hour ago in 4s

> Set up job

> Download Docker Image

> Load Docker Image

> Push image Docker ke Docker Hub

1 ▶ Run docker push javiersosial/hello-world-gitaction:1.0

4 The push refers to repository [docker.io/javiersosial/hello-world]

5 c18897d5e3dd: Preparing

6 9af9e76ea07f: Preparing

7 f1f70b13aacc: Preparing

8 252b6db79fae: Preparing

9 c9ce8cb4e76a: Preparing

10 8f3c313eb124: Preparing

11 c1761f3c364a: Preparing

12 08000c18d16d: Preparing

13 c1761f3c364a: Waiting

14 08000c18d16d: Waiting

15 8f3c313eb124: Waiting

16 unauthorized: access token has insufficient scopes

17 Error: Process completed with exit code 1.

Terlihat bahwa pada tahap push image ke Docker Hub terdapat pesan “unauthorized : access token has insufficient scopes”. Error ini berarti GitHub Actions tidak memiliki izin yang cukup untuk **push ke Docker Hub**. Untuk itu perlu dilakukan perbaikan seperti berikut :

◆ Perbaikan yang Dilakukan

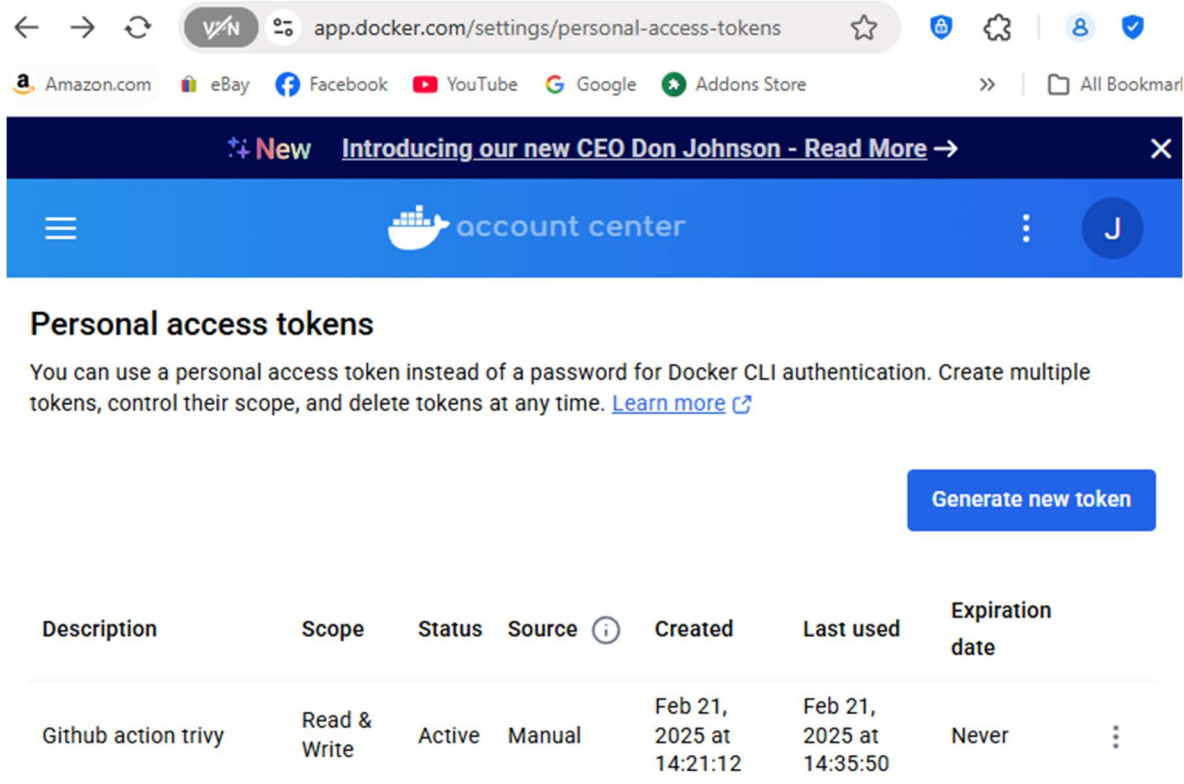
1. Gunakan **docker/login-action@v3** → Versi @v2 sudah cukup stabil, tetapi coba gunakan @v3 untuk memastikan kompatibilitas.
2. Gunakan **DOCKERHUB_TOKEN** bukan **DOCKER_PASSWORD** → Docker Hub menganjurkan penggunaan Access Token, bukan password akun.
3. Pastikan DOCKERHUB_TOKEN memiliki izin **write** untuk bisa melakukan push.

Membuat DockerHub token

Jika password DockerHub login masih menggunakan **password akun Docker**, gantilah dengan **Access Token Docker Hub**:

1. Masuk ke Docker Hub → <https://hub.docker.com/settings/security>
2. Buat Access Token Baru
3. Beri nama token dan pastikan read/write access
4. Simpan token di GitHub Secrets sebagai DOCKERHUB_TOKEN

Dockerhub > Account settings > Personal access tokens > generate new token



The screenshot shows the Docker Hub account settings page for 'Personal access tokens'. The browser address bar shows 'app.docker.com/settings/personal-access-tokens'. The page has a blue header with the Docker logo and 'account center'. Below the header, there's a section titled 'Personal access tokens' with a subtext: 'You can use a personal access token instead of a password for Docker CLI authentication. Create multiple tokens, control their scope, and delete tokens at any time. [Learn more](#)'. A blue button 'Generate new token' is visible. Below this is a table with columns: Description, Scope, Status, Source, Created, Last used, and Expiration date. The table contains one entry: 'Github action trivy' with Scope 'Read & Write', Status 'Active', Source 'Manual', Created 'Feb 21, 2025 at 14:21:12', Last used 'Feb 21, 2025 at 14:35:50', and Expiration date 'Never'.

Description	Scope	Status	Source	Created	Last used	Expiration date
Github action trivy	Read & Write	Active	Manual	Feb 21, 2025 at 14:21:12	Feb 21, 2025 at 14:35:50	Never

Masukkan deskripsi token dan pastikan permission read & write.

Create access token

A personal access token is similar to a password except you can have many tokens and revoke access to each one at any time. [Learn more](#)

Expiration date

None

Optional

Access permissions

Read & Write

Read & Write tokens allow you to push images to any repository managed by your account.

Cancel

Generate

Setelah itu token baru akan muncul. Copy token tersebut ke notepad karena token tersebut tidak dapat terlihat lagi setelah ditutup.

Buat secret baru di Github dengan nama DOCKERHUB_TOKEN lalu paste token dari dockerhub tadi sebagai value atau isi dari secret tersebut. Cara membuat secret Github sudah dijelaskan di awal.

Jika secrets DOCKERHUB_TOKEN sudah dibuat, ganti secrets DOCKER_PASSWORD pada file workflow menjadi DOCKERHUB_TOKEN. Berikut adalah bagian dari file workflow yang sudah diperbaharui secretnya.


```

21 # 📁 2 Build Docker Image jika kode bersih
22 build-image:
23   runs-on: ubuntu-latest
24   needs: secret-scan # Build hanya berjalan jika GitLeaks sukses
25   steps:
26     - name: Checkout kode
27       uses: actions/checkout@v3
28
29     - name: Login ke Docker Hub
30       uses: docker/login-action@v3
31       with:
32         username: ${ secrets.DOCKER_USERNAME }
33         password: ${ secrets.DOCKERHUB_TOKEN } # Gunakan token, bukan password
34
35     - name: Build image Docker
36       run: docker build -t javiersosial/hello-world-gitaction:1.0 .
37
38     - name: Save Docker Image ke File
39       run: docker save -o image.tar javiersosial/hello-world-gitaction:1.0
40
41     - name: Upload Docker Image untuk Scan
42       uses: actions/upload-artifact@v4
43       with:
44         name: docker-image
45         path: image.tar

```

```

71 # 🚀 4 Push Image ke Docker Hub jika Scan Trivy berhasil
72 push-image:
73   runs-on: ubuntu-latest
74   needs: scan-image # Push hanya berjalan jika Trivy scan sukses
75   steps:
76     - name: Download Docker Image
77       uses: actions/download-artifact@v4
78       with:
79         name: docker-image
80
81     - name: Load Docker Image
82       run: docker load -i image.tar
83
84     - name: Login ulang ke Docker Hub sebelum push
85       uses: docker/login-action@v3
86       with:
87         username: ${ secrets.DOCKER_USERNAME }
88         password: ${ secrets.DOCKERHUB_TOKEN }
89
90     - name: Push image Docker ke Docker Hub
91       run: docker push javiersosial/hello-world-gitaction:1.0
92

```

Setelah mengganti secrets pada bagian bagian tersebut, dilakukan commit dan push untuk menjalankan action kembali.

The screenshot shows the GitHub Actions interface for the repository 'javiersosial / scan-gitleaks'. The workflow 'Gitleaks-Docker Build-Trivy & Push' is displayed, with the job 'push-image' highlighted. The job status is 'succeeded 2 hours ago in 9s'. A detailed log for the 'push-image' job is shown, listing the following steps:

- Set up job
- Download Docker Image
- Load Docker Image
- Login ulang ke Docker Hub sebelum push
- Push image Docker ke Docker Hub
- Post Login ulang ke Docker Hub sebelum push
- Complete job

Terlihat job push-image telah berhasil berjalan. Ini berarti image berhasil di push ke DockerHub secara aman.

The screenshot shows the Docker Hub interface for the repository 'javiersosial/hello-world-gitaction'. The image is listed with the following details:

- Name: javiersosial/hello-world-gitaction
- Last Pushed: about 2 hours ago
- Contains: IMAGE
- Visibility: Public
- Size: 1m 17s

Terlihat image hello-world-gitaction pada repo Dockerhub menandakan workflow telah berhasil berjalan hingga melakukan push ke Docker hub.

Pada proses yang telah dilalui terlihat bagaimana gitleaks dan terutama trivy bekerja untuk menjaga keamanan dari kebocoran secrets dan mengamankan image yang digunakan dari vulnerability pada rangkaian CI/CD melalui github action.

Hal ini menandakan bahwa satu proses CI/CD yang menerapkan proses devsecops didalamnya telah dibuat dan telah dijelaskan step by step.