📘 **Canvas: Phân Tích Sâu – Bảo Mật Container trong GenAI Factory**

---

## 🎯 Mục tiêu: Bảo vệ Đơn vị Triển khai (Container)

Trong hệ thống **GenAI Factory**, Container là đơn vị triển khai cốt lõi trong Kubernetes. Do đó, việc bảo mật Container là một **mục tiêu Hardening bắt buộc** trong lớp `infra/`, nhằm giảm thiểu bề mặt tấn công và đảm bảo tuân thủ các tiêu chuẩn bảo mật doanh nghiệp.

Giai đoạn này tập trung vào hai chiến lược chính: 1. **Hardening Dockerfile (Non-Root User & Multi-Stage Build)** 2. **Quét lỗ hổng bảo mật tự động trong CI/CD Pipeline**

---

## I. 🛡️ Hardening Dockerfile.assistant (Non-Root User & Multi-Stage Build)

Dockerfile là nền tảng của bảo mật container. Việc Hardening giúp đảm bảo image chạy nhẹ, an toàn và tránh rủi ro leo thang đặc quyền.

| Kỹ thuật Hardening | Cơ chế kỹ thuật | Lý do quan trọng |
|---|---|---|
| **Multi-Stage Builds** | Tách biệt giai đoạn Build và Runtime. | ✅ *Giảm kích thước Image & Bề mặt tấn công:* Image cuối cùng chỉ chứa các tệp cần thiết để chạy ứng dụng (mã Python, thư viện cần thiết). Loại bỏ các công cụ build, compiler không cần thiết, giúp giảm lỗ hổng tiềm ẩn. |
| **Non-Root User** | Thêm lệnh `USER appuser` để tạo người dùng không có đặc quyền. | 🧩 *Ngăn chặn Leo thang Đặc quyền:* Nếu bị tấn công, hacker chỉ có quyền hạn thấp bên trong container, không thể tấn công sang kernel của Node Kubernetes. |

### Ví dụ: Dockerfile.assistant (Đã Hardening)

```
# --- GIAI ĐOẠN 1: BUILD ---
FROM python:3.11-slim as builder
WORKDIR /app
```

```
# Cài đặt dependencies build nếu cần
# RUN apt-get update && apt-get install -y build-essential

# Cài đặt dependencies Python
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt

# --- GIAI ĐOẠN 2: RUNTIME ---
FROM python:3.11-slim as runtime
WORKDIR /app

# 1. Tạo người dùng không có đặc quyền (HARDENING)
RUN adduser --disabled-password --gecos '' appuser

# Copy dependencies từ builder stage
COPY --from=builder /usr/local/lib/python3.11/site-packages
/usr/local/lib/python3.11/site-packages
COPY . /app

# 2. Thiết lập quyền sở hữu
RUN chown -R appuser:appuser /app

# 3. Chuyển sang người dùng không có đặc quyền
USER appuser

# Chạy ứng dụng
CMD ["uvicorn", "main:app", "--host", "0.0.0.0", "--port", "8000"]
```

💡 **Kết quả:** Image nhỏ gọn hơn, an toàn hơn và không chạy bằng quyền root — một chuẩn bảo mật bắt buộc cho môi trường Kubernetes.

---

## II. 🔍 Quét Lỗ Hổng (Vulnerability Scanning) trong CI/CD

Chạy container với người dùng không có đặc quyền là cần thiết nhưng **chưa đủ**. Bạn cần đảm bảo image không chứa thư viện lỗi thời hoặc lỗ hổng đã biết.

| Kỹ thuật Hardening | Vị trí trong Dự án | Vai trò Kỹ thuật |
|---|---|---|
| **Static Analysis (Hadolint)** | `infra/cicd/github-actions.yaml` | Phân tích Dockerfile: Hadolint kiểm tra cú pháp và cảnh báo nếu phát hiện các vấn đề bảo mật (ví dụ: sử dụng `USER root`). |
| **Vulnerability Scanning (Trivy/Clair)** | `infra/cicd/github-actions.yaml` | Quét Image sau khi build để tìm các lỗ hổng (CVEs) trong hệ điều hành, thư |

| Kỹ thuật Hardening | Vị trí trong Dự án | Vai trò Kỹ thuật |
| --- | --- | --- |
| | | viện Python, và các gói phụ thuộc. |

### Ví dụ: GitHub Actions Workflow (Đã Hardening)

```yaml
# infra/cicd/github-actions.yaml

jobs:
  build_and_scan:
    steps:
    - name: Run Hadolint (Dockerfile Linter)
      uses: hadolint/hadolint-action@v3
      # Hardening: Fail build nếu Dockerfile không đạt chuẩn bảo mật.

    - name: Build Docker Image
      run: docker build -t genai-assistant:latest .

    - name: Run Trivy Vulnerability Scan (CRITICAL HARDENING)
      uses: aquasec/trivy-action@master
      with:
        image-ref: 'genai-assistant:latest'
        format: 'table'
        exit-code: '1' # Dừng pipeline nếu có lỗ hổng nghiêm trọng
        severity: 'CRITICAL,HIGH'
```

---

## III. 🔒 Tóm tắt Giá trị Bảo mật

- ✅ **Multi-Stage Builds:** Giảm kích thước và bề mặt tấn công.
- ✅ **Non-Root User:** Ngăn chặn leo thang đặc quyền.
- ✅ **Static & Dynamic Analysis:** Phát hiện lỗi cấu hình và CVE sớm trong CI/CD.
- ✅ **Fail-Fast Policy:** Tự động dừng triển khai nếu phát hiện lỗ hổng nghiêm trọng.

---

## 🚀 Kết luận

Bằng việc áp dụng các kỹ thuật bảo mật container này, **GenAI Factory** đạt được: - Bảo mật cấp độ doanh nghiệp cho container. - Tuân thủ tiêu chuẩn DevSecOps & SOC2. - Quản lý rủi ro bảo mật tự động hóa trong CI/CD pipeline.

> **Tóm lại:** Container Hardening biến GenAI Factory trở thành một hệ thống AI an toàn, linh hoạt và sẵn sàng cho môi trường sản xuất.