

# SDK Secret Manager Design Pattern

- 1 Giới thiệu
  - 1.1 Mục tiêu của SDK
  - 1.2 Cấu trúc module
  - 1.3 Các thư viện hỗ trợ
  - 1.4 Đối tượng sử dụng
  - 1.5 Tính huống áp dụng
- 2 Technology Stack
  - 2.1 Custom Provider Connector (AWS Connector, GCP Connector, Vault Connector)
    - 2.1.1 Mục tiêu
    - 2.1.2 Cấu trúc mỗi connector bao gồm:
    - 2.1.3 Ưu điểm
  - 2.2 Secret Manager Registry
    - 2.2.1 Mục tiêu
    - 2.2.2 Chức năng chính
    - 2.2.3 Ưu điểm
  - 2.3 SDK Client + Spring AutoConfiguration Pattern
- 3 Project Structure (Maven)
  - 3.1 description
  - 3.2 Project overview
  - 3.3 Module responsibility
  - 3.4 Module Detail description
    - 3.4.1 jcanary-secure-config-interface
    - 3.4.2 jcanary-secure-config-aws-adapter
    - 3.4.3 jcanary-secure-config-gcp-adapter
    - 3.4.4 jcanary-secure-config-vault-adapter
    - 3.4.5 jcanary-secure-config-core
    - 3.4.6 jcanary-encrypt-interface
- 4 Processing Flow
  - 4.1 Activity Flow
  - 4.2 Data flow
- 5 Configuration Management
  - 5.1 Provider Configuration
  - 5.2 Vault Configuration
  - 5.3 AWS Configuration
  - 5.4 GCP Configuration
  - 5.5 Encrypt Configuration
- 6 Usage Manual
  - 6.1 Module configuration
  - 6.2 yaml configuration
  - 6.3 Bean usage manual

## Giới thiệu

Trong dự án phát triển ứng dụng Spring Boot, tôi đã xây dựng một **SDK Secret Manager** tùy chỉnh để lấy các thông tin bí mật (secrets) từ ba hệ thống quản lý bí mật hàng đầu: **HashiCorp Vault**, **Google Cloud Platform (GCP) Secret Manager**, và **AWS Secrets Manager**. SDK này cho phép ứng dụng truy xuất các bí mật, chẳng hạn như thông tin xác thực cơ sở dữ liệu (spring.datasource.username, spring.datasource.password), một cách an toàn và sử dụng chúng để cấu hình các kết nối, như DataSource. SDK đảm bảo rằng các bí mật được lấy từ các hệ thống quản lý bí mật chuyên dụng, tăng cường bảo mật và loại bỏ rủi ro từ việc hard-code hoặc lưu trữ trong file cấu hình.

SDK Secret Manager của tôi mang lại các lợi ích chính:

- **Đa nền tảng:** Hỗ trợ lấy bí mật từ HashiCorp Vault, GCP Secret Manager, và AWS Secrets Manager thông qua một giao diện thống nhất.
- **Bảo mật:** Lưu trữ thông tin nhạy cảm trong các hệ thống quản lý bí mật an toàn.
- **Dễ tích hợp:** Cung cấp API đơn giản để lấy bí mật và sử dụng trong ứng dụng Spring Boot.
- **Tính linh hoạt:** Cho phép ứng dụng truy xuất bí mật từ nhiều nguồn mà không cần thay đổi logic chính.

## Mục tiêu của SDK

SDK Secret Manager được thiết kế để:

- Cung cấp một giao diện chung để lấy bí mật từ HashiCorp Vault, GCP Secret Manager và AWS Secrets Manager.
- Tích hợp với ứng dụng Spring Boot để sử dụng bí mật trong các cấu hình kết nối, như DataSource.

- Đảm bảo tính bảo mật bằng cách truy xuất bí mật trực tiếp từ các hệ thống quản lý bí mật.
- Hỗ trợ debug và giám sát thông qua logging chi tiết để theo dõi việc lấy bí mật.

## Cấu trúc module

Dự án jca-secure-config được tổ chức thành các module sau:

- jcanary-secure-config-interface**: Định nghĩa các giao diện và hợp đồng (contract) cho việc lấy bí mật, được sử dụng bởi các adapter.
- jcanary-secure-config-core**: Chứa logic cốt lõi để phối hợp giữa các adapter và tích hợp với Spring Boot.
- jcanary-secure-config-gcp-adapter**: Module adapter để lấy bí mật từ GCP Secret Manager.
- jcanary-secure-config-vault-adapter**: Module adapter để lấy bí mật từ HashiCorp Vault.
- jcanary-secure-config-aws-adapter**: Module adapter để lấy bí mật từ AWS Secrets Manager.
- jcanary-secure-config-encrypt-adapter**: Module adapter để lấy bí mật được mã hóa.

## Các thư viện hỗ trợ

SDK sử dụng các thư viện sau để tương tác với các hệ thống quản lý bí mật và tích hợp với Spring Boot:

- Spring Boot** (3.4.2): Cung cấp nền tảng để xây dựng ứng dụng và quản lý bean.
- Spring Cloud** (2024.0.1): Hỗ trợ tích hợp với các hệ thống bên ngoài, bao gồm spring-cloud-starter-vault-config (4.2.0) cho HashiCorp Vault.
- Google Cloud Secret Manager** (2.51.0): SDK để truy xuất bí mật từ GCP Secret Manager.
- Vault Java Driver** (5.1.0): Thư viện để tương tác với HashiCorp Vault.
- AWS SDK Secrets Manager** (2.25.22): SDK để lấy bí mật từ AWS Secrets Manager.
- Jackson Databind** (2.18.3): Xử lý JSON khi làm việc với bí mật.
- SLF4J** (2.0.16) và **Logback** (1.5.16): Hỗ trợ logging chi tiết.
- Lombok** (1.18.36): Giảm mã boilerplate trong code Java.

## Đối tượng sử dụng

- Lập trình viên**: Các nhà phát triển sử dụng SDK để tích hợp việc lấy bí mật từ HashiCorp Vault, GCP Secret Manager, hoặc AWS Secrets Manager vào ứng dụng Spring Boot, đảm bảo cấu hình kết nối an toàn (như DataSource).
- Kiến trúc sư hệ thống**: Những người thiết kế hoặc đánh giá giải pháp bảo mật, cần một SDK đa nền tảng để quản lý bí mật trong các hệ thống phân tán.
- Kỹ sư kiểm thử (QA)**: Những người cần hiểu cách SDK tương tác với Vault, GCP, và AWS để viết kịch bản kiểm thử cho các luồng liên quan đến cấu hình và kết nối.
- Thành viên mới trong nhóm**: Những người cần tìm hiểu nhanh về cách quản lý bí mật trong dự án, tận dụng tài liệu và API đơn giản của SDK để làm quen với hệ thống.

## Tình huống áp dụng

- Phát triển microservice cần bảo mật cao**: Khi xây dựng các ứng dụng Spring Boot yêu cầu truy xuất thông tin xác thực cơ sở dữ liệu hoặc API key từ HashiCorp Vault, GCP Secret Manager, hoặc AWS Secrets Manager, SDK cung cấp một giao diện thống nhất để lấy bí mật an toàn, tránh hard-code.
- Triển khai đa nền tảng**: Trong các hệ thống phân tán sử dụng nhiều nhà cung cấp đám mây (AWS, GCP) hoặc Vault nội bộ, SDK cho phép quản lý bí mật từ các nguồn khác nhau mà không cần thay đổi mã nguồn ứng dụng.

## Technology Stack

### Custom Provider Connector (AWS Connector, GCP Connector, Vault Connector)

Hệ thống được thiết kế với kiến trúc **plug-and-play**, cho phép tích hợp linh hoạt các **Secret Provider Connector** tùy theo nhu cầu triển khai và môi trường thực tế.

#### Mục tiêu

- Trừu tượng hoá các cách truy xuất secrets của từng nền tảng (AWS, GCP, Vault...).
- Đảm bảo interface sử dụng đồng nhất, không phụ thuộc vào backend provider cụ thể.
- Cho phép mở rộng / thay thế / tùy chỉnh từng connector một cách độc lập.

Cấu trúc mỗi connector bao gồm:

- **Client:**  
Giao tiếp trực tiếp với dịch vụ secrets tương ứng (VD: AWS SDK, GCP SDK, Vault HTTP API).
- **Provider:**  
Triển khai interface `SecretProvider`, đảm nhận việc map request vào client, xử lý logic, trả về secrets dưới dạng chuẩn hoá.
- **DTO:**  
Định nghĩa các đối tượng request/response dùng nội bộ trong từng connector.
- **Configuration:**  
Cấu hình tự động khởi tạo bean (@Configuration, @ConditionalOnProperty) để bật/tắt connector tùy môi trường.
- **Properties:**  
Cấu hình mở rộng cho từng provider: keyPath, auth, region, credential, prefix...
- **Holder (Optional):**  
Lưu cache client/provider hoặc state cục bộ, giảm chi phí khởi tạo lặp lại.

## Ưu điểm

- **Dễ mở rộng:** chỉ cần tạo một connector mới theo cấu trúc trên nếu muốn hỗ trợ thêm provider khác.
- **Không ảnh hưởng đến các module khác:** mọi thay đổi đều đóng gói trong adapter riêng biệt.
- **Dễ kiểm thử:** có thể mock từng connector hoặc inject stub/fake provider trong unit test.

## Secret Manager Registry

`SecretProviderRegistry` đóng vai trò là **registry trung tâm** để quản lý và điều phối các **Secret Provider** đến từ các connector như AWS, GCP, Vault... trong hệ thống.

## Mục tiêu

- Tập trung hóa việc đăng ký và truy xuất các `SecretProvider` tương ứng với từng provider type (VD: aws, gcp, vault).
- Cho phép lựa chọn provider theo prefix hoặc tên định danh đã được cấu hình.
- Tăng tính **mở rộng** và **linh hoạt** cho hệ thống quản lý secrets.

## Chức năng chính

- **Đăng ký provider:**  
Mỗi connector khi được khởi tạo sẽ tự động đăng ký `SecretProvider` tương ứng vào registry.
- **Truy xuất provider:**  
Cho phép lấy đúng `SecretProvider` theo `PrefixProviderEnums` (enum phân biệt các provider), hỗ trợ pattern như: `aws://key_name`, `gcp://key_name`, `vault://key_name`.
- **Tùy biến:**  
Có thể mở rộng hoặc override provider logic bằng cách đăng ký lại provider mới theo nhu cầu (ví dụ: mock trong test).

## Ưu điểm

- **Decoupling** hoàn toàn giữa phía người dùng (consumer) và backend provider implementation.
- Cho phép quản lý nhiều provider cùng lúc trong một ứng dụng.
- Dễ dàng mở rộng để tích hợp thêm các provider mới chỉ bằng cách đăng ký thêm vào registry.

## SDK Client + Spring AutoConfiguration Pattern

Hệ thống áp dụng mô hình **Secure SDK** kết hợp với **Spring Boot AutoConfiguration** nhằm:

1. **Tách biệt logic truy xuất secrets với business logic của ứng dụng**
  - Toàn bộ việc truy xuất và quản lý secrets (Vault, AWS Secrets Manager, GCP Secret Manager...) được đóng gói trong SDK.
  - Giúp các service sử dụng (auth, identity, product...) không cần quan tâm đến chi tiết triển khai hay backend cụ thể.
2. **Tái sử dụng dễ dàng thông qua cấu hình động**
  - SDK hỗ trợ cấu hình linh hoạt qua `application.yml` (hoặc biến môi trường) cho các tham số như: auth type, path, prefix provider...
  - Cho phép chuyển đổi giữa các provider (AWS ↔ GCP ↔ Vault) mà không cần chỉnh sửa mã nguồn.
3. **Mở rộng module dễ dàng**
  - Khi tích hợp thêm provider mới, chỉ cần:
    - Định nghĩa `SecretProvider` mới.
    - Thêm `AutoConfiguration` tương ứng.

- Không ảnh hưởng đến các phần còn lại của hệ thống.

#### 4. Tùy biến linh hoạt qua Spring Profile / Bean Injection

- Có thể override các bean (ví dụ: client HTTP, provider registry, credential provider...) cho từng môi trường: dev, test, prod.
- Hỗ trợ inject client tùy chỉnh hoặc sử dụng Spring Cloud Vault/AWS SDK tùy ý.

#### 5. Giao tiếp backend không bị ràng buộc bởi giao thức

- Dễ dàng tích hợp với các phương thức giao tiếp khác (REST, gRPC, Kafka...) bằng cách mở rộng provider tương ứng.
- Phản xử lý secrets vẫn giữ nguyên logic truy xuất và mapping cấu hình qua environment.

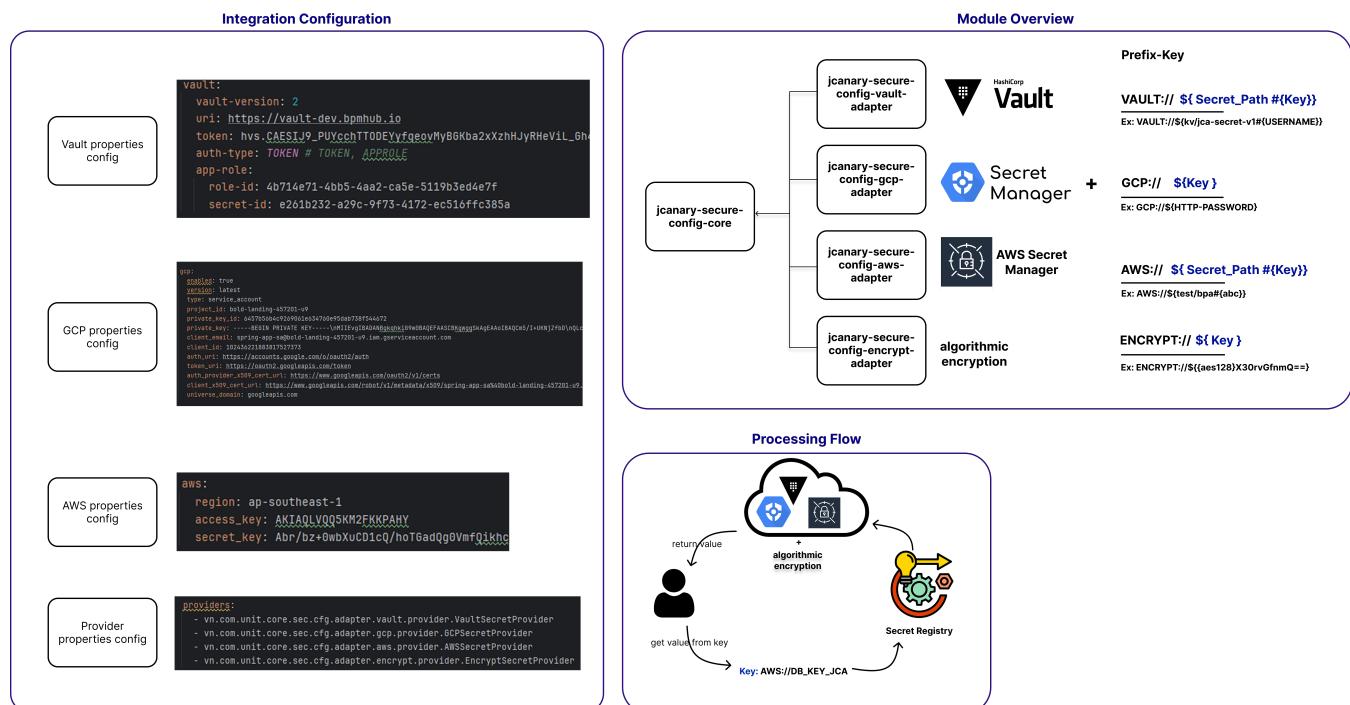
## Project Structure (Maven)

### description

Dựa trên cấu hình, dự án "jcanary-secure-config" là một SDK Java (được quản lý qua Maven) với các đặc điểm sau:

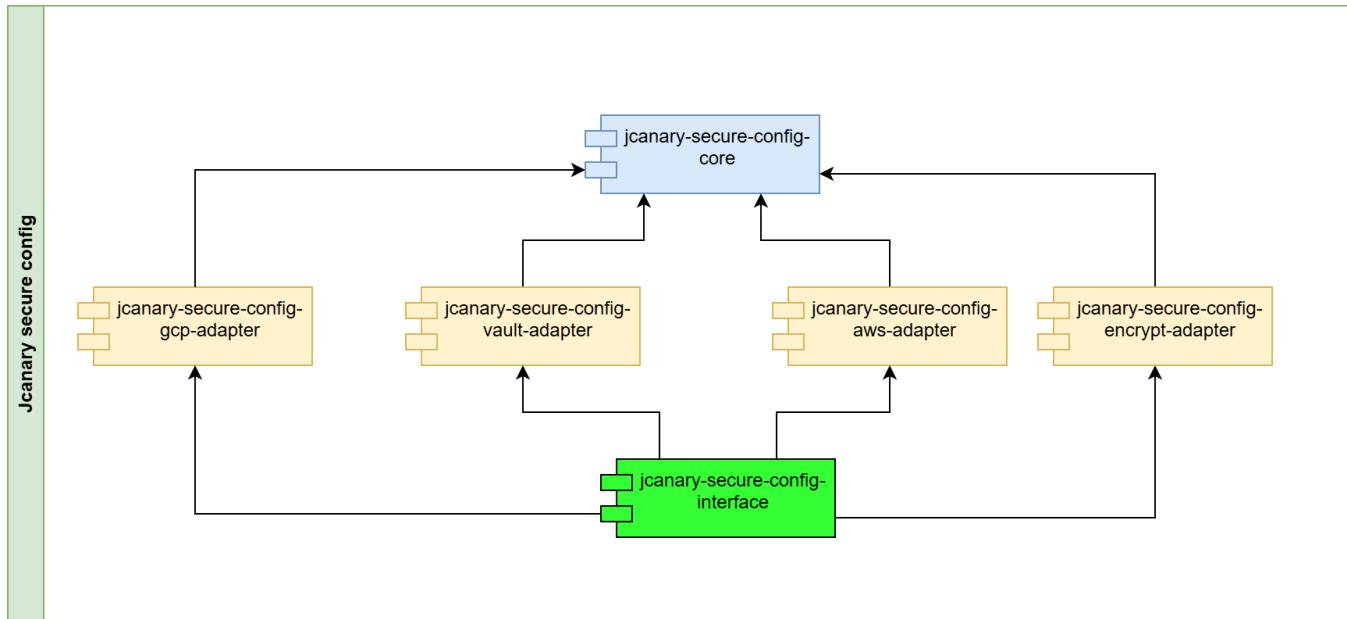
- Mục đích:** Cung cấp một cách thức thống nhất để quản lý và truy xuất các bí mật (secrets) từ nhiều nguồn khác nhau (HashiCorp Vault, GCP Secret Manager, AWS Secrets Manager, và mã hóa nội bộ).
- Tính linh hoạt:**
  - Hỗ trợ nhiều nhà cung cấp bí mật thông qua các providers.
  - Cho phép bật/tắt hoặc cấu hình từng nguồn bí mật thông qua adapters.
- Tính bảo mật:**
  - Tích hợp với các dịch vụ quản lý bí mật chuyên dụng (Vault, GCP, AWS) đảm bảo các thông tin nhạy cảm được lưu trữ và truy xuất an toàn.
  - Hỗ trợ mã hóa nội bộ (EncryptSecretProvider) để bảo vệ dữ liệu ngay cả khi không sử dụng dịch vụ bên ngoài.
- Cách sử dụng:**
  - Các ứng dụng Java khác có thể thêm SDK này làm dependency trong pom.xml và sử dụng nó để truy xuất cấu hình bảo mật.
  - Cấu hình YAML (hoặc tương tự) được sử dụng để định nghĩa các nhà cung cấp và adapter, giúp dễ dàng tùy chỉnh hành vi của SDK.

## Project overview



Ghi chú: Do Encrypt provider sẽ được cấu hình cùng với JCA source nên các cấu hình environment là không cần thiết.

## Module responsibility



## Module Detail description

Dự án jca-secure-config được tổ chức thành các module sau, mỗi module đảm nhiệm một vai trò cụ thể trong SDK Secret Manager để hỗ trợ lấy bí mật từ HashiCorp Vault, GCP Secret Manager, và AWS Secrets Manager:

### jcanary-secure-config-interface

`vn.com.unit.core.sec.cfg.dto`

Package chứa các class Dto các input truyền vào interface và các dto parent.

Class	Mô tả
EnvSecretProperties	Dto chứa các tham số chung
SecretDto	DTO dùng để truyền thông tin các secret giữa các lớp hoặc module, có thể chứa key, value, type, v.v.

`vn.com.unit.core.sec.cfg.enums`

Enum định nghĩa các loại prefix provider khác nhau (ví dụ: VAULT, AWS, LOCAL...) để phân loại source của secrets.

Class	Mô tả
PrefixProviderEnums	Enum định nghĩa các loại prefix provider khác nhau (ví dụ: VAULT, AWS, LOCAL...) để phân loại source của secrets.

`vn.com.unit.core.sec.cfg.interfaces`

Định nghĩa các phương thức giao tiếp với secret provider.

Class	Mô tả
EnvSecretClient	Interface định nghĩa các phương thức giao tiếp với secret provider (ví dụ: lấy secret theo prefix, đọc thông tin môi trường...), có thể được implement bởi client cụ thể như VaultClient.
SecretProvider	Interface định nghĩa contract cho bất kỳ class nào cung cấp secrets (có thể là HashiCorp Vault, AWS Secrets Manager, hay cấu hình nội bộ...), nhằm hỗ trợ mở rộng theo chuẩn Strategy Pattern.

### jcanary-secure-config-aws-adapter

`vn.com.unit.core.sec.cfg.adapter.aws.client`

Class	Mô tả
AWSecretClient	class định nghĩa client để tương tác với AWS Secrets Manager. Có thể dùng để truy xuất secret theo key, prefix...

`vn.com.unit.core.sec.cfg.adapter.aws.configuration`

Class	Mô tả
AWSAutoConfiguration	Cấu hình Spring Boot tự động (@Configuration) giúp khởi tạo các bean liên quan đến AWS (client, provider, properties...) nếu thỏa điều kiện. Đây là entrypoint cho Spring context khi sử dụng module này.

`vn.com.unit.core.sec.cfg.adapter.aws.dto`

Class	Mô tả
AWSecretRequestDto	Đối tượng truyền dữ liệu chứa thông tin truy vấn secret từ AWS Secrets Manager, ví dụ: secret name, region, key prefix, version, v.v.

`vn.com.unit.core.sec.cfg.adapter.aws.holder`

Class	Mô tả
AWSecretClientHolder	Class dùng để giữ một instance của AWSecretClient theo kiểu singleton/static holder (hoặc có thể cache client theo region /profile). Hữu ích khi dùng ở nhiều nơi mà không cần inject trực tiếp.

`vn.com.unit.core.sec.cfg.adapter.aws.properties`

Class	Mô tả
AWSKeyManagementProperties	Mapping cấu hình từ application.yml/application.properties liên quan đến AWS key management hoặc secrets config. Có thể dùng @ConfigurationProperties.

`vn.com.unit.core.sec.cfg.adapter.aws.provider`

Class	Mô tả
AWSecretProvider	Implementation của SecretProvider interface (từ module jcanary-secure-config-interface), dùng để thực hiện logic cụ thể lấy secrets từ AWS. Đây là nơi tương tác chính với AWS SDK.

## jcanary-secure-config-gcp-adapter

`vn.com.unit.core.sec.cfg.adapter.gcp.client`

Class	Mô tả
GcpSecretClient	Lớp hoặc interface chịu trách nhiệm tương tác trực tiếp với <b>Google Secret Manager API</b> để lấy thông tin các secrets.

`vn.com.unit.core.sec.cfg.adapter.gcp.configuration`

Class	Mô tả
GCPAutoConfiguration	Lớp cấu hình tự động trong Spring Boot, tự động khởi tạo các bean như GcpSecretClient, GCPSecretProvider, properties.

`vn.com.unit.core.sec.cfg.adapter.gcp.dto`

Class	Mô tả
GcpSecretRequestDto	Đối tượng truyền dữ liệu (Data Transfer Object) chứa thông tin yêu cầu để lấy secrets từ GCP, như secretId, projectId, version...

`vn.com.unit.core.sec.cfg.adapter.gcp.holder`

Class	Mô tả
GcpSecretClientHolder	Singleton hoặc cache class để giữ instance GcpSecretClient. Có thể dùng static hoặc context để tránh khởi tạo lại nhiều lần.

`vn.com.unit.core.sec.cfg.adapter.gcp.properties`

Class	Mô tả
GcpKeyManagementProperties	Lớp ánh xạ cấu hình từ application.yml chứa thông tin như GCP projectId, prefix, cấu hình xác thực với GCP...

`vn.com.unit.core.sec.cfg.adapter.gcp.provider`

Class	Mô tả
GCPSecretProvider	Triển khai interface SecretProvider, thực hiện logic <b>truy vấn thực tế</b> từ GCP Secret Manager. Đây là nơi xử lý chính khi app cần lấy secret từ GCP.

## jcanary-secure-config-vault-adapter

`vn.com.unit.core.sec.cfg.adapter.vault.client`

Class	Mô tả
VaultSecretClient	Định nghĩa client để tương tác trực tiếp với Vault Server thông qua HTTP API (hoặc thư viện Vault Java SDK). Là cầu nối giữa hệ thống và Vault để truy vấn secret theo key, path...

`vn.com.unit.core.sec.cfg.adapter.vault.configuration`

Class	Mô tả
VaultAutoConfiguration	Spring Boot auto-configuration class: giúp tự động cấu hình và khởi tạo các bean như VaultSecretClient, VaultSecretProvider

`vn.com.unit.core.sec.cfg.adapter.vault.dto`

Class	Mô tả
VaultSecretRequestDto	DTO truyền dữ liệu chứa thông tin yêu cầu để truy xuất một secret từ Vault

`vn.com.unit.core.sec.cfg.adapter.vault.holder`

Class	Mô tả
VaultSecretClientHolder	Singleton holder class hoặc caching layer để lưu trữ và quản lý VaultSecretClient đã được khởi tạo. Hữu ích khi cần tránh tạo lại connection hoặc client nhiều lần.

`vn.com.unit.core.sec.cfg.adapter.vault.properties`

Class	Mô tả
AppRole	Cấu hình liên quan đến AppRole authentication của Vault, gồm roleId, secretId.
AuthType	Enum định nghĩa các loại authentication Vault hỗ trợ (ví dụ: TOKEN, APPROLE)
VaultKeyManagementProperties	Lớp ánh xạ cấu hình từ application.yml/application.properties liên quan đến Vault

`vn.com.unit.core.sec.cfg.adapter.vault.provider`

Class	Mô tả
VaultSecretProvider	Lớp chính triển khai interface SecretProvider từ interface chung. Đây là nơi xử lý lấy secrets thực tế từ Vault.

## jcanary-secure-config-core

`vn.com.unit.core.sec.cfg.adapter.autoconfiguration`

Class	Mô tả
EnvBeanAutoconfiguration	Chứa các cấu hình config tạo bean tự động cho adapter chính.

`vn.com.unit.core.sec.cfg.adapter.constant`

Class	Mô tả
EnvConstant	Chứa các constant

`vn.com.unit.core.sec.cfg.adapter.processor`

Class	Mô tả
AdapterEnvironmentProcessor	<b>Custom EnvironmentPostProcessor:</b> được chạy rất sớm trong lifecycle của Spring Boot để: <ul style="list-style-type: none"> <li>Inject các secrets từ provider vào Spring Environment.</li> <li>Gắn kết dữ liệu lấy từ AWS/GCP/Vault vào application.yml hoặc bean config runtime.</li> <li>Có thể mapping theo prefix như: aws:db/password, vault:api/key.</li> </ul>

`vn.com.unit.core.sec.cfg.adapter.registry`

Class	Mô tả
SecretProviderRegistry	Registry để quản lý nhiều SecretProvider được đăng ký từ các adapter khác nhau (AWSSecretProvider, GCPSecretProvider, VaultSecretProvider...).
JcaPropertiesSource	Triển khai custom PropertySource cho Spring Boot. Lớp này là cầu nối giữa Spring Environment và secrets được lấy từ provider.

## jcanary-encrypt-interface

`vn.com.unit.core.sec.cfg.adapter.encrypt.aesdecrypt`

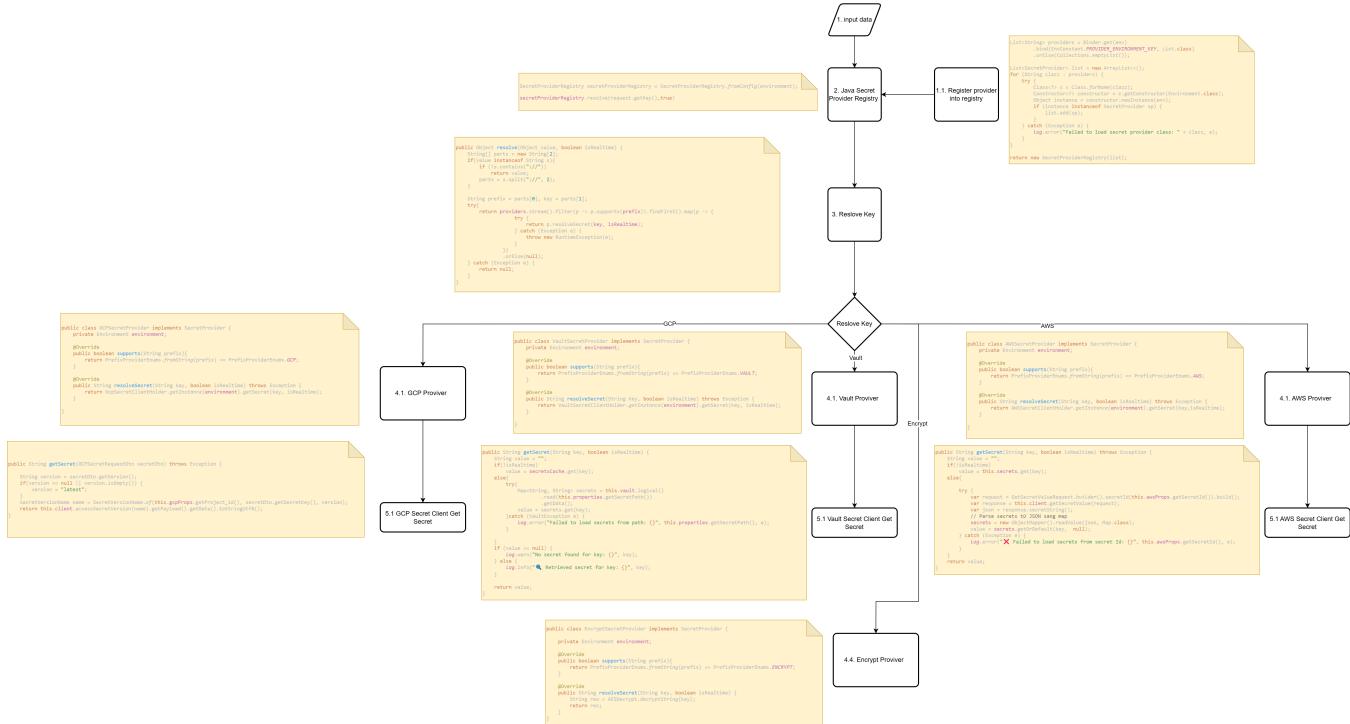
Class	Mô tả
AESDecrypt	Class quyết định xử lý các secret được mã hóa và giải mã trong source.

`vn.com.unit.core.sec.cfg.adapter.encrypt.provider`

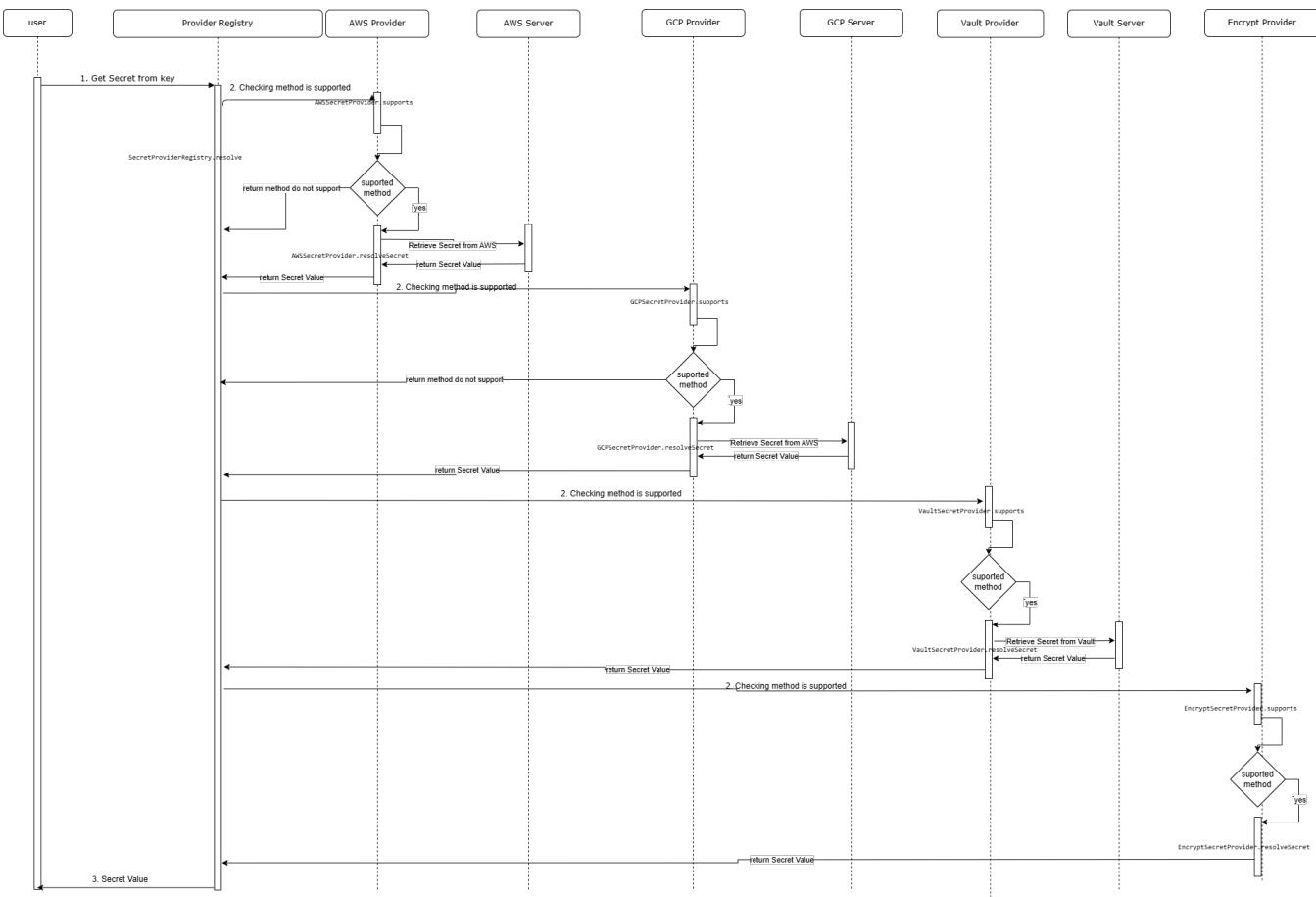
Class	Mô tả
EncryptSecretProvider	Lớp chính triển khai interface SecretProvider từ interface chung. Đây là nơi xử lý secrets c mã hóa.

# Processing Flow

## Activity Flow



## Data flow



## Configuration Management

### Provider Configuration

`application.yaml`

```
jcanary:
  secure-config: # none, vault, aws, gcp, azure, googleSecret
  providers:
    - vn.com.unit.core.sec.cfg.adapter.vault.provider.VaultSecretProvider
    - vn.com.unit.core.sec.cfg.adapter.gcp.provider.GCPSecretProvider
    - vn.com.unit.core.sec.cfg.adapter.aws.provider.AWSSecretProvider
    - vn.com.unit.core.sec.cfg.adapter.encrypt.provider.EncryptSecretProvider
```

### Vault Configuration

### pom

```
<dependency>
    <groupId>vn.com.unit.core</groupId>
    <artifactId>jcanary-secure-config-vault-adapter</artifactId>
</dependency>
```

### application.yaml

```
jcanary:
  secure-config: # none, vault, aws, gcp, azure, googleSecret
  adapters:
    vault:
      uri: http://192.168.253.150:8200/
      token: root
      auth-type: APPROLE # TOKEN, APPROLE
      app-role:
        role-id: 917915e9-419c-43c2-f5f6-3c81a9f4f50b
        secret-id: 953d36bd-a642-51a7-8034-309d11645a83
```

## AWS Configuration

### pom

```
<dependency>
    <groupId>vn.com.unit.core</groupId>
    <artifactId>jcanary-secure-config-aws-adapter</artifactId>
</dependency>
```

### application.yaml

```
key-management: # none, vault, aws, gcp, azure, googleSecret
aws:
  region: ap-southeast-1
  access_key: AKIAQLVQQ5KM2Fxxxxxxxxx
  secret_key: Abr/bz+0wbXuCD1cQ/h0TGaxxxxxxx
```

## GCP Configuration

### pom

```
<dependency>
    <groupId>vn.com.unit.core</groupId>
    <artifactId>jcanary-secure-config-gcp-adapter</artifactId>
</dependency>
```

## application.yaml

```
jcanary:
  secure-config: # none, vault, aws, gcp, azure, googleSecret
    adapters:
      gcp:
        enabled: true
        version: latest
        type: service_account
        project_id: bold-landing-xxxxxxxx
        private_key_id: 6457b56b4c9269061xxxxxxxxxx
        private_key: -----BEGIN PRIVATE KEY-----\xxxxxxxxxxxxxxxxx/I+UKNjZfbD\nQLcxTQmj/xxxxxxxxxxxxxx\nRIE
/VOfvfozo+bJX4yXseX8A9UXc0i+OGEIuzcRpIW2XM8JfNuxybwp9Lzz32+KY\niuXVrtgmzO8DPalVI8suLgMDK0QbzrM
//WGCUDIsXjHCISazUKo
/P9YAYFBgxfp\niZrVyeRcbKsaZbRUY0vn61XbZmWirwmYi6C3RcttAbhwBJPcsjVS31or6onnJyv+\n7jIY6aXxGB4stXdlysUnWKOtsw8vy
2yAW+Nk6BVoVa75DybJcEuhiMs
/Dd+Q+X0u\nrVuP9jKBAgMBAECggEAAGQNVKnnmcdTFe1QYxpEH3K4WCGMA1Jty4qFkvINpQx8\naEPQFYJMW5myqQ4rVb1JIx4ph2JIeNi
eMZNMXI18WzW71275xzHtAIN+82bfP8\nnzJyD96aJAKgtezLuDiVnJFwNiFjsF2skqtELZOXz5sgkFylwy4sUkDquSCC5cMBC\nnoHvXlDpuY
pdoOL24q6fWWAAiletFpnjsZYaYwaHKsfz
/YLWr+1pcKms8KRr0VqCrf\ncQ01m94h6gRbXaXkJDSkn65sWwbL3IZdpvR50kWqssRWnxX1HoZ3mr2FB3K2oRL3b\nnsdP0DKZ7QPO9QA5MaoA
PGszpAn7UOwymwZdry9ZUYQKBgQDrXNDMWdarGbF5zRa\nnqcN5CJ+ZI3C32/zTZETeQM2IU2i
/3ayPoHWQixNvyzLCNPKydljQFeb8k6R9E5h\npLVsCYpSE3ZkKzrsYGIryb6HGSjYZ/VQD1JfQNV8iiCvRjhzsakir9mT
/M2DNA5\nnjsgRoC8oGogYgILtyvqiOyaIQKBgQC1ined2Ap7FF6RopGJa/Jj3D3mwffjNxqC\nn0hQ8I3AFgPc/iR3r6B6qSFdRCHM
/X+dV6pLaVLIJaiduTY9bdmlzXPT61I5wpiDC\nn01MSaIp0/Mod19AYB5Isf10vmBmPPWxmTzmN+z
/UZsRDR8tSGKe6Spk1F5tV3hP6\nnPKfqKs5MYQKBgQCr498dVB/LtHNOF5+dwzQOVfzxQ4
/Tx2crH6SoSN2SFZAX6Bzd\nn3QvH5gYjNcbJ9YVwKdvJujkBnIJDWBBSgY85+vG5JZumZmgfL/kiJ613hHr
/U5H+\n1lwBPrrL7xI1AWOFrpUTo2dLmoI+bDv5wsxdAJ
/qYw4YDusQLdjD2L04QKBgF4X\n1lsfJIiWI+Tsit6wydv+rN6pS+3ohJWyI1lQVSggQPZaAyzFqrfo0ZI1rg9BlNNIj\nnMtpW67BVhPp22MbF
KOjSVWdLeEiK0o/Zj8UKyji4ietV4FWUFv9X5/azLww/GeAt\n
/d3Mi401aTJoWh7vvOf1qkeAzUOvextY9rLNQIBAOGBALcEk5hFi4PJ6E1vKUmI\nnZj1us3vBLUXYeGYUpZzlZyxnqaqQJ8+mVyr9+bnehoYg
vLXS18pcsmxVvcxxHNxi\nrxHgFFvYzjWlQtOBiMqBIVYQcXCpHJ22R7trcDSeWsRLRnB2meNPe9MrIuMnMF
/F\n4R5hHB5TZwlkvODTlY6QJ1D\n----END PRIVATE KEY----\n
  client_email: spring-app-sa@bold-landing-457201-u9.iam.gserviceaccount.com
  client_id: 1024362xxxxxxxxxxxx
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url: https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url: https://www.googleapis.com/robot/v1/metadata/x509/spring-app-sa%40bold-landing-
457201-u9.iam.gserviceaccount.com
  universe_domain: googleapis.com
```

## Encrypt Configuration

### pom

```
<dependency>
  <groupId>vn.com.unit.core</groupId>
  <artifactId>jcanary-secure-config-encrypt-adapter</artifactId>
</dependency>
```

## Usage Manual

### Module configuration

Để tích hợp SDK Secret Manager vào project Spring Boot, bạn cần khai báo các dependency tương ứng trong file pom.xml.

## pom

```
<!-- Bt buc: Adapter chính giúp ng ký các SecretProvider -->
<dependency>
    <groupId>vn.com.unit.core</groupId>
    <artifactId>jcanary-secure-config-core</artifactId>
</dependency>

<!-- Tùy chn: Thêm các adapter tng ng vi provider bn cn tích hp -->
<dependency>
    <groupId>vn.com.unit.core</groupId>
    <artifactId>jcanary-secure-config-gcp-adapter</artifactId>
</dependency>

<dependency>
    <groupId>vn.com.unit.core</groupId>
    <artifactId>jcanary-secure-config-vault-adapter</artifactId>
</dependency>

<dependency>
    <groupId>vn.com.unit.core</groupId>
    <artifactId>jcanary-secure-config-aws-adapter</artifactId>
</dependency>
```

## yaml configuration

## application.yaml

```
jcanary:
  secure-config: # none, vault, aws, gcp, azure, googleSecret
  providers:
    # - vn.com.unit.core.sec.cfg.adapter.vault.provider.VaultSecretProvider
    - vn.com.unit.core.sec.cfg.adapter.gcp.provider.GCPSecretProvider
    - vn.com.unit.core.sec.cfg.adapter.aws.provider.AWSecretProvider
    - vn.com.unit.core.sec.cfg.adapter.encrypt.provider.EncryptSecretProvider
  adapters:
    vault:
      uri: http://192.168.253.150:8200/
      token: root
      auth-type: APPROLE # TOKEN, APPROLE
      app-role:
        role-id: 917915e9-xxxxxxxxxxxxxxxxxxxxxx
        secret-id: 953d36bd-xxxxxxxxxxxxxxxxxxxxxx

    gcp:
      enabled: true
      version: latest
      type: service_account
      project_id: bold-landing-457201-u9
      private_key_id: 6457b5xxxxxxxxxxxxxx
      private_key: -----BEGIN PRIVATE KEY-----\xxxxxxxxxxxxxxxx/I+UKNjZfbD\nQLcxTQmj/xxxxxxxxxxxxxxxx\x\nRIE
/V0VfoZo+xxxxxxxxxxxxxx+xxxxxxxxxxxxxxxxxxxxx+KY\niuXrtgmzO8DPalVI8suLgMDK0QbzrM//WGCJdIsXjHCISazUKo
/P9YAYFBgxfp\niZrVyeRcbKsaZbRUY0vn61XbZmWirwmYi6C3RcttAbhwBjPcsjVS3lor6onnJyv+\n7jIY6aXxGB4stXdlysUnWKOtsw8vy
2yAW+Nk6BVoVa75DybJcEuhImS
/Dd+Q+x0u\nrVuP9jkBAgMBAECggEAAgQNVKnncdTFE1QYxpEH3K4WCGMALJty4qFkvInPQx8\nAEQPQFYJYJmW5myQ4rVb1JIx4ph2JIEni
eMZNMXI18WzW71275xzHtAIN+82bfP8\nnzJyD96aJAkgtezLuDiVnJFwNiFjsf2skqtELZOXz5sgkFylwy4sUkDquSCC5cMBC\nnoHvXlDpuY
pdoOL24q6fWWAiletpfnjsZYaYwaHKsfz
/YLWr+ipcKms8Krr0VqCrf\qnQ01m94h6gRbXaXkJDSkn65sWwbL3IZdpvR50kWqssRWnxXlHoZ3mr2FB3K2oRL3b\nSDp0DKZ7QPO9QA5MaoA
PGszpAn7UOwymwZdry9ZUYQKBgQDrXNdMWODarGbf5zRa\nnqcN5CJ+Z13C32/ZTZEToQM2IU2i
/3ayPoHWQixNvyzLCNPKydljQFeb8k6R9E5h\npLVscypSE3ZkKzrsYGIrybj6HGSjYZ/VQD1JfQNV8iiCvRjhzsakir9mT
/M2DNa5\njsgRoZC8oGogYgILtyvqiOyaIQKBgQClined2Ap7FF6RopGJa/Jj3D3mwffjNxqC\nn0hQ8I3AFgPc/iR3r6B6qSFdRCHM
/X+dV6pLaVLIJaiduTY9bdmlzXPT61I5wpiDC\nn01MSaIp0/Mod19AYB5Isf10vmBmPPWxmTzmN+z
/UZsRDR8tSGKe6Spk1F5tV3hP6\lnPKfqKs5MYQKBgQCr498dvB/Lthnof5+dwzQOVfzxQ4
/Tx2crH6SoSN2SFZAX6BzD\ln3Qvh5gYjNcbJ9YVwKdvJujkBnIJDWBBSgY85+vG5JZumZmgfL/kij613hHr
/U5H+\n1WwBPrrL7xI1AWOFrpUTo2dLmoI+bDv5wsxdAjj
/qYw4YDusQLjdD2L04QKBgF4X\lnlsfJIiWI+Tsit6wydv+rN6pS+3ohJWyi1lQVsgqQPZaAyZfqrfo0ZI1rg9BlNNIj\nMtpW67BVhPp22Mbf
KOjSVWdLeEiK0o/Zj8UKyji4ietV4FWUFv9X5/azLww/GeAt\nnt
/d3Mi401aTJoWh7vvOf1qkeAzUOvextY9rLNQIBAoGBALcEk5hFi4PJ6E1vKUmI\nnZj1us3vBLUXYeGYUpZzlZyxnqaqQJ8+mVyr9+bnehOYg
vLXs18pcsmxVvcxxHNxi\lnrxHgFVyzjWlQtObiMqBIVYQcXCpHJ22R7trcDSeWsRLRnb2meNPe9MrIuMnMF
/F\ln4R5hHB5TZwlkiVODTIy6QJ1D\n----END PRIVATE KEY----\n
  client_email: spring-app-sa@bold-landing-457201-u9.iam.gserviceaccount.com
  client_id: 10243xxxxxxxxxxxxxxxxxxxxxx
  auth_uri: https://accounts.google.com/o/oauth2/xxxxxxxxxxxxxxxxxxxxxx
  token_uri: https://oauth2.googleapis.com/xxxxxxxxxxxxxxxxxxxxxx
  auth_provider_x509_cert_url: https://www.googleapis.com/xxxxxxxxxxxxxxxxxxxxxx
  client_x509_cert_url: https://www.googleapis.com/robot/v1/metadata/x509/xxxxxxxxxxxxxxxxxxxxxx
  universe_domain: googleapis.com

aws:
  region: ap-southeast-1
  access_key: acxxxxxxxxxxxxxxxxxxxxxx
  secret_key: Abr/bz+0wbXuCD1cQ/xxxxxxxxxxxxxxxxxxxxxx

# mt ví d s dng GCP  ly key DB_KEY_JCA t secret manager
spring:
  datasource:
    username-gcp: GCP://${DB_USERNAME}:DB_KEY_JCA}
    username-vault: VAULT://${kv/jca-secret-management-v1#{AUTH-URL}}
    username-aws: AWS://${test/bpa/api#{AUTH_TYPE}}
```

## Phân tích cấu trúc YAML

jcanary

- Đây là định danh đặc thù của tổ chức (theo yêu cầu, không phân tích chi tiết).
- Trong ngữ cảnh này, jcanary là namespace chính, có thể đại diện cho tổ chức hoặc dự án tổng thể, và secure-config là một module hoặc tính năng cụ thể bên trong.

#### adapters

- Đây chứa cấu hình nối kết của Vault, GCP, AWS khi cần sử dụng những secret manager nào thì cấu hình nối kết của phần đó cần được thêm vào trong yaml.

#### providers

- Danh sách các lớp Java (fully qualified class names) triển khai giao diện hoặc logic để cung cấp bí mật (secrets) từ các nguồn khác nhau. Các nhà cung cấp này chịu trách nhiệm truy xuất thông tin nhạy cảm (như mật khẩu, API key, hoặc các giá trị cấu hình) từ các hệ thống bên ngoài hoặc cơ chế mã hóa nội bộ.
- Các nhà cung cấp được liệt kê:
  - **vn.com.unit.core.sec.cfg.adapter.vault.provider.VaultSecretProvider:**
    - Nhà cung cấp này tích hợp với **HashiCorp Vault**, một công cụ phổ biến để quản lý bí mật. Nó có thể truy xuất các bí mật từ Vault server để sử dụng trong ứng dụng.
    - Gói vn.com.unit.core.sec.cfg.adapter.vault cho thấy đây là một adapter tùy chỉnh được phát triển bởi tổ chức (UNIT) để tương tác với Vault.
  - **vn.com.unit.core.sec.cfg.adapter.gcp.provider.GCPSecretProvider:**
    - Nhà cung cấp này tích hợp với **Google Cloud Platform (GCP) Secret Manager**, một dịch vụ quản lý bí mật trên GCP.
    - Tương tự, đây là một adapter tùy chỉnh để truy xuất bí mật từ GCP Secret Manager.
  - **vn.com.unit.core.sec.cfg.adapter.aws.provider.AWSSecretProvider:**
    - Nhà cung cấp này tích hợp với **AWS Secrets Manager** hoặc một dịch vụ tương tự của AWS (như AWS Systems Manager Parameter Store).
    - Adapter này cho phép ứng dụng truy xuất bí mật từ các dịch vụ AWS.
  - **vn.com.unit.core.sec.cfg.adapter.encrypt.provider.EncryptSecretProvider:**
    - Nhà cung cấp này có thể liên quan đến việc mã hóa hoặc giải mã các bí mật tại chỗ (local encryption) thay vì dựa vào dịch vụ bên ngoài.
    - Nó có thể sử dụng các thuật toán mã hóa (như AES) để bảo vệ dữ liệu cấu hình trước khi lưu trữ hoặc sau khi truy xuất.
- **Ý nghĩa:**
  - Danh sách providers cho thấy SDK "jcanary-secure-config" được thiết kế để hỗ trợ nhiều nguồn bí mật khác nhau (Vault, GCP, AWS, và mã hóa nội bộ). Điều này mang lại tính linh hoạt, cho phép ứng dụng chuyển đổi giữa các nhà cung cấp bí mật mà không cần thay đổi mã nguồn.
  - Các adapter đều thuộc gói vn.com.unit.core.sec.cfg.adapter, ám chỉ rằng đây là các thành phần tùy chỉnh do tổ chức phát triển, có thể dựa trên các SDK chính thức của Vault, GCP, hoặc AWS.

## Bean usage manual

Tích hợp vào dự án để lấy key từ source

### Implementation.java

```
# khai báo bean ã c nh nghe SecretProviderRegistry
private final SecretProviderRegistry secretProviderRegistry;
# s dng bean ly mt secret value t key "AWS://${test/bpa/api#{AUTH_TYPE}} , GCP://KEY_EXAMPLE, VAULT://${kv/jca-secret-management-v1#{AUTH_URL}}"
secretProviderRegistry.resolve("AWS://${test/bpa/api#{AUTH_TYPE}}")
```