# Resposta das Questões D

**d) Demonstração de Proteção de Configurações Sensíveis**

A proteção de configurações sensíveis (como senhas de banco de dados ou chaves de API) no Spring Cloud Config é feita através da criptografia dos valores no repositório Git.

**1. Estratégia**

O Config Server utiliza uma chave secreta (encrypt.key) para:

1. Criptografar o valor sensível.
2. Armazenar o valor criptografado no arquivo YAML do Git, prefixado com {cipher}.
3. Descriptografar o valor em tempo de execução, antes de enviá-lo ao microserviço cliente.

Dessa forma, a senha nunca fica em texto simples no repositório versionado.

**2. Configuração Necessária**

Para habilitar a criptografia, a chave secreta deve ser definida no application.yml do Config Server:

```
# config-server/src/main/resources/application.yml

encrypt:

  key: SuaChaveSecretaDeProducaoAqui
```

**3. Criptografia (Exemplo de Comando)**

Para obter o valor criptografado de uma senha, você faria uma requisição ao endpoint /encrypt do Config Server (após ele estar rodando):

```
# Comando de exemplo para criptografar "SenhaSuper@Secreta123!"

curl -u admin:admin123 -X POST http://localhost:8888/encrypt -d "SenhaSuper@Secreta123!"
```

**4. Armazenamento no Repositório Git**

O valor retornado pelo comando acima é o que deve ser armazenado no arquivo de configuração do seu serviço (ex: account-service-prod.yml ), usando o prefixo {cipher}:

```
# config-repo/account-service-prod.yml

spring:

  datasource:

    url: jdbc:mysql://prod-db-server:3306/banco_digital

    username: admin_prod

    # O valor abaixo é o resultado da criptografia

    password:
'{cipher}AQBkKzI5dGZUMXhHMkJMN2dUQTBSMlJCK3dFVmVjVWJqVXJYMEVOdHJ
UUVpxUTBkMjJvdWJlQzBvVlJOWGJLRzBSMUJMVWpRPT0='
```

Resumo da Proteção: O Config Server garante que o cliente receba a senha em texto simples, mas o repositório Git armazena apenas o valor criptografado, protegendo o segredo.