GEHEIMES SICHER IN DER CLOUD

Javaland 2018

Dominik Schadow bridgingIT


Secrets
stored securely in
cloud environments

```
spring:
  datasource:
    name: myDatabase
    username: myDatabaseUser
    password: mySuperSecretDatabasePassword
management:
    context-path: /admin
    security:
      enabled: true
logging:
    level: warn
```

ID	USERNAME	PASSWORD	SECRET_NUMBER	SECRET_TEXT
1	Arthur	kvgkIu7ZuPIdK9G7WUA duTvd9TinwRlvA6foux mgxMZwUsPUdW6	42	Dear diary
2	Zaphod	wC28772M7AYVwLe2BOu dFl8VBo59KS5H1MbY9i riZpQhP6KCd33	42	I don’t have any secrets
3	Slarti	eHkuCs817pYySnk0aKl zDeZDCSiUSedCOABqcE sRVYzS1Uc8RzK	42	Yeehaa
4	Ford	3kQkFnjyt008yrIjnjf tZewS6j8yKIbywJYzvs 3HOGqtfYcAVV0	42	Is someone reading this?

Technical Credentials


Embedded Configuration

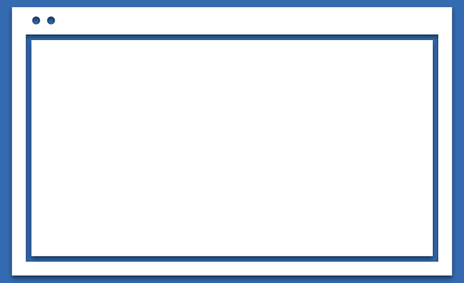
```
<dependency>
  <groupId>
    com.github.ulisesbocchio
  </groupId>
  <artifactId>
    jasypt-spring-boot-starter
  </artifactId>
  <version>
    1.18
  </version>
</dependency>
```

```
$ encrypt.sh input="mySuperSecretDatabasePassword" password="sample-password"

----ENVIRONMENT-----------------

Runtime: Oracle Corporation Java HotSpot(TM) 64-Bit Server VM 25.144-b01



----ARGUMENTS-------------------

input: mySuperSecretDatabasePassword
password: sample-password



----OUTPUT----------------------

p12MjKuU5xk7Pn0OdxJf627ssaNyov2U30MRKlVjQb97Cbkfi5+/uQ==
```

```
spring:
  datasource:
    name: myDatabase
    username: myDatabaseUser
    password: ENC(p12MjKuU5xk7Pn0OdxJf627ssa
              Nyov2U30MRKlVjQb97Cbkfi5+/uQ==)
management:
    context-path: /admin
    security:
       enabled: true
logging:
    level: warn
```

jasypt.encryptor.password

System property



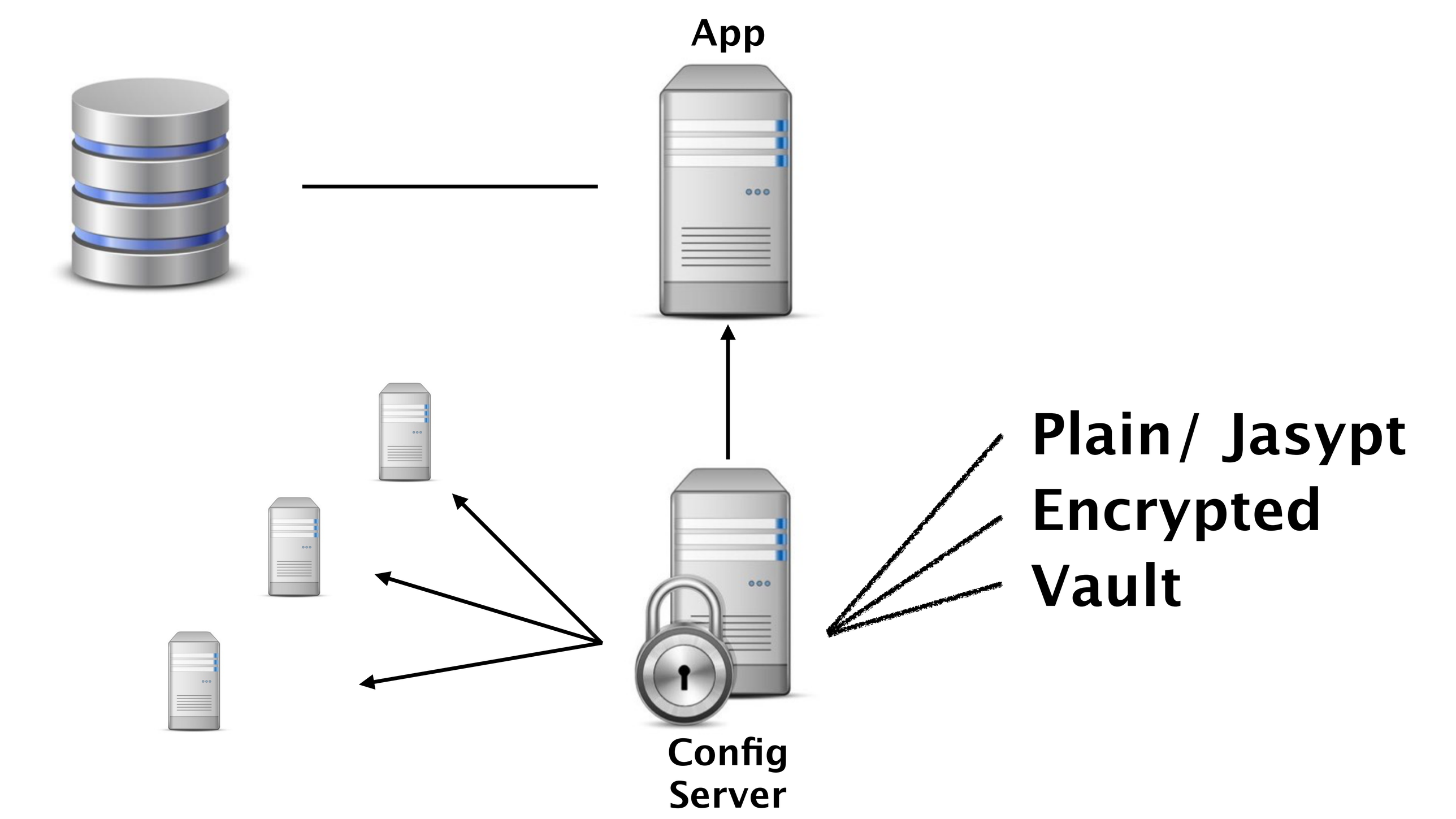
Command line argument

Environment variable

Encrypted

Encryption password
Distribution
Jasypt security

External Configuration


App
Plain/ Jasypt
Encrypted
Vault
Config
Server

Config Server Cryptography

Properties starting with {cipher} will be decrypted before returned to client

Symmetric Encryption
encrypt.key property

Asymmetric Encryption
key (PEM or Java KeyStore) and encrypt.key-store.* properties

```
$ curl localhost:8888/encrypt -u user:secret -d client-password
AQA1zqFrxmYnfu4GHRYgXSowMNZ2O97kntv5nwt2VKlmuh0+rZQNYA8INI+yOJ9L2mIqTpuUwpZpRgTs
jAx+MtgW5d09yNPoQGWYRWQ9Y+xTvzR99j8jF7ShryC4fzOkX+I2zsAqQ+36qucwGyJkKegAAmcOVClN
rCeh9owrPXI0jShlYTyTFx9+mmZ7ICvhD7f57PZax2WzlKzcLzHkxBK6+vHrQjdykUm/DCo3vAOnwXgV
mZAz1gKdlSl5B3DfDRwqXNcI8b9avX2b4VtzqpamdUhzJgSFJOvPO5JRbZZ5zMKQO+2G3u5QiYrRShks
/2l7FxPHXr12BHsiiOh9FbhbJNYNfFxiopPe7lbOjJreJfFFeBAl2Ez+78vYz0nhkZs=
```

```
spring:
  datasource:
    name: client-db
    username: client-user
    password: '{cipher}AQA1zqFrxmYnfu4GHRYgX
SowMNZ2O97kntv5nwt2VKlmuh0+rZQNYA8INI+yOJ9
L2mIqTpuUwpZpRgTsjAx+MtgW5d09yNPoQGWYRWQ9Y
+xTvzR99j8jF7ShryC4fzOkX+I2zsAqQ+36qucwGyJ
kKegAAmcOVClNrCeh9owrPXI0jShlYTyTFx9+mmZ7I
3vAOnwXgVmZAz1gKdlSl5B3DfDRwqXNcI8b9avX2b4
VtzqpamdUhzJgSFJOvPO5JRbZZ5z+78vYz0nhkZs='
```

```
{
 "name": "config-client",
 "profiles": ["cipher"],
 "propertySources": [{
  "name": "https://github...config-client-cipher.yml",
  "source": {
   "application.name": "Config Client",
   "application.profile": "Cipher",
   "spring.datasource.name": "client-db",
   "spring.datasource.password": "client-password",
   "spring.datasource.username": "client-user",
   "spring.h2.console.enabled": true } }]
}
```

```
{
 "name": "config-client",
 "profiles": ["cipher"],
 "propertySources": [{
  "name": "https://github...config-client-cipher.yml",
  "source": {
   "application.name": "Config Client",
   "application.profile": "Cipher",
   "spring.datasource.name": "client-db",
   "spring.datasource.password": "<n/a>",
   "spring.datasource.username": "client-user",
   "spring.h2.console.enabled": true } }]
}
```

Demo

Encrypted
Distributed

Encryption credentials
Config server security
Availability

A tool for managing secrets

VAULT

"... centrally store, secure, and tightly control access to secrets across distributed infrastructure, applications, and humans."

All Data is Always Encrypted

- Internal key encrypts everything with AES
  - Key never leaves the system
- Selected storage backend never sees plain text
  - File, Amazon DynamoDB, Consul, …

Secret Backends

- Store and create secrets
  - Lease time for new secrets
  - Access control policies for secrets
- Dynamic secrets (AWS, …)
- Token revocation

Audit Log is Disabled by Default

- Detailed audit log of all authenticated client interaction
- Sensitive data hashed

```
$ vault audit-enable file file_path=./vault_audit.log
Successfully enabled audit backend 'file' with path 'file'!
```

Accessible via **HTTP API** or **CLI**

```
vault server -config vault.conf
```

Shamir's Secret Sharing

```
vault operator init -key-shares=5 -key-threshold=2

Unseal Key 1: Pv/Xx49co4Zmed2McapSOr4jC4iiAvfd5EjvILMySJUB
Unseal Key 2: T00pjFgitbcy+JKOGI6DFgW/0jBdyrVriLdGu7PENbsC
Unseal Key 3: YCOKtRUITlH3h155P5LM+2zLbFgIe4vwrOIhO7OWHqED
Unseal Key 4: rTLOGu3emdWa4QyKysY6Tmice1u4QTEcUFIPlrMzz+cE
Unseal Key 5: glxtI6D0YjNfnsB97dp1owHoxTPt8A+HdAdoFrNh5P0F
Initial Root Token: efe88b79-cf8b-825a-0f6f-ef1ca142782b
```

Visible once after initialization

Unsealing requires **n** configured **unseal keys**

```
vault operator unseal
  Pv/Xx49co4Zmed2McapSOr4jC4iiAvfd5EjvILMySJUB

Sealed: true
Key Shares: 5
Key Threshold: 2
Unseal Progress: 1
Unseal Nonce: 87f350d5-2a25-a821-dc7f-2962fc49fe03
```

```
vault operator unseal
  rTLOGu3emdWa4QyKysY6Tmice1u4QTEcUFIPlrMzz+cE

Sealed: false
Key Shares: 5
Key Threshold: 2
Unseal Progress: 0
Unseal Nonce:
```

Authenticated Access Required

Token (default), LDAP, Username/Password,
GitHub Token, AWS EC2, Certificates, …

```
export VAULT_TOKEN=
efe88b79-cf8b-825a-0f6f-ef1ca142782b
```

mount point

```
vault write secret/config-client
spring.datasource.password=client-db
```

key-value format
(generic backend)

Spring Cloud Vault

```
/secret/{application}/{profile}
/secret/{application}
/secret/{defaultContext}/{profile}
/secret/{defaultContext}
```

```
$ vault write secret/config-client-vault spring.datasource.password=config-client-db-password
Success! Data written to: secret/config-client-vault
```

Demo

Encrypted

Distributed access

Multiple unseal keys

Vault server security

Availability

Personal Credentials

API for Secret Management

VaultTemplate to read, list,
write and delete secrets

```
vaultTemplate.
read(PATH)
```

```
vaultTemplate.
list(PATH)
```

```
vaultTemplate.
write(PATH, data)
```

```
vaultTemplate.
delete(PATH)
```

```
@Autowired
VaultTemplate template;

@PostMapping("/secrets")
ResponseEntity write(Secret secret) {
 Map<String, String> secrets = new HashMap<>();
 secrets.put("password", secret.getPassword());
 // ...

 VaultResponse response =
  template.write("secret/" + userId, secrets);
 // ...
}
```

```
path "secret/*"
{
  capabilities = ["create", "read",
                  "update", "delete",
                  "list"]
}
```

Encrypted

Distributed access

Multiple unseal keys

Access policies

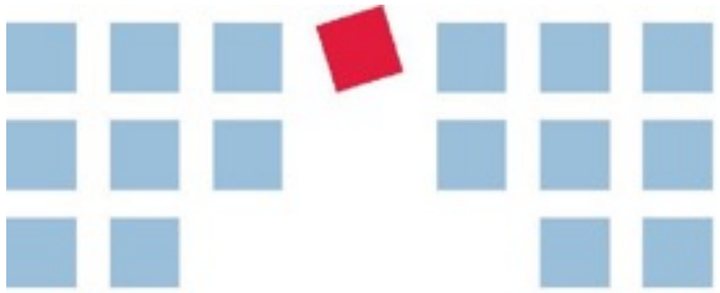
Vault server security

Availability

1. **Jasypt** for non distributed applications with **minimal security** requirements

2. **Spring Cloud Config** for applications with **typical security** requirements

3. **Vault** for applications with **high security** requirements



bridging IT

Marienstr. 17
70178 Stuttgart

dominik.schadow@bridging-it.de
www.bridging-it.de

Blog blog.dominikschadow.de
Twitter @dschadow

Demo Project
https://github.com/dschadow/CloudSecurity

Jasypt
http://www.jasypt.org

Jasypt integration for Spring Boot
https://github.com/ulisesbocchio/jasypt-spring-boot

Spring Cloud
http://projects.spring.io/spring-cloud

Vault
https://www.vaultproject.io

Pictures
http://www.dreamstime.com