spring配置中经常使用placeholder来加载一个应用配置文件(.properties)，但是其中的各种密码以明文显示出来总该是不好。

不过可以利用jasypt这个框架来扩展这个加密功能，需要用到jasypt中的icu4j-version.jar、jasypt-version-lite.jar、jasypt-version.jar和jasypt-spring31-version.jar

首先，注释掉原有的placeholder加载方式

1. <!--<context:property-placeholder location="/WEB-INF/config.properties"/> -->

然后使用jasypt为spring相应版本实现的placeholder

1. <!-- decrypt password in config.properties -->
2. **<bean** id="environmentVariablesConfiguration"
3. class="org.jasypt.encryption.pbe.config.EnvironmentStringPBEConfig"**>**
4. **<property** name="algorithm" value="PBEWithMD5AndDES" **/>**
5. **<property** name="password" value="root" **/>**
6. **</bean>**
7. **<bean** id="configurationEncryptor" class="org.jasypt.encryption.pbe.StandardPBEStringEncryptor"**>**
8. **<property** name="config" ref="environmentVariablesConfiguration" **/>**
9. **</bean>**
10. **<bean** id="propertyConfigurer"
11. class="org.jasypt.spring31.properties.EncryptablePropertyPlaceholderConfigurer"**>**
12. **<constructor-arg** ref="configurationEncryptor" **/>**
13. **<property** name="locations"**>**
14. **<list>**
15. **<value>**/WEB-INF/config.properties**</value>**
16. **</list>**
17. **</property>**
18. **<property** name="fileEncoding" value="utf-8" **/>**
19. **</bean>**

最后，修改.properties配置中的明文密码为密文，这个需要自己写一个main方法

1. **public** **static** **void** main(String[] args) {
2. //PBEWithMD5AndDES
3. BasicTextEncryptor encryptor = **new** BasicTextEncryptor();
4. encryptor.setPassword("root");
5. String encrypted = encryptor.encrypt("xxxx");
6. System.out.println(encrypted);
7. }

然后将输出的密文替换原来的密码

1. jdbc.password=ENC(jHv0WdiTLJFmOO08RQtUpg==)

这样的密文虽然还是很容易被decode出来，但终究不是明文显示。

个人认为最好的办法可能是 自己去实现一个spring的 place holder，利用md5来匹配配置文件中的密文是否正确。

1. **import** org.jasypt.util.text.BasicTextEncryptor;
2. **import** org.jasypt.util.text.StrongTextEncryptor;

5. **public** **class** EncypterTest {
7. **public** **static** **void** main(String[] args) {
8. //加密
9. BasicTextEncryptor textEncryptor = **new** BasicTextEncryptor();
10. textEncryptor.setPassword("password");
11. String newPassword = textEncryptor.encrypt("123456");
12. System.out.println(newPassword);
13. //        解密
14. BasicTextEncryptor textEncryptor2 = **new** BasicTextEncryptor();
15. textEncryptor2.setPassword("password");
16. String oldPassword = textEncryptor2.decrypt(newPassword);
17. System.out.println(oldPassword);
18. System.out.println("--------------------------");
19. StrongTextEncryptor ste = **new** StrongTextEncryptor();
20. //加密
21. ste.setPassword("password");
22. String encyptedResult= ste.encrypt("123456");
23. System.out.println("encyptedResult:"+encyptedResult);
24. //解密
25. String dencyptedResult = ste.decrypt(encyptedResult);
26. System.out.println(dencyptedResult);

29. }
30. }

http://liu-shui8.iteye.com/blog/508615