Configuration at Tomcat Level and Code changes

Note: If there are passwords already stored in the database without encryption, these passwords need to be encrypted and changed in the database**.** If not changed, exceptions may occur while running the application. Encryption of already stored can be done using the terminal. The steps to do that have been mentioned below in the section “Encryption and Decryption at Unix Terminal”.

Before storing any password in the database, the values must be encrypted and while getting the values back the values must be decrypted.

Configuration must be done in the tomcat.

We store the encryption key which is used for the encryption/decryption of plain text passwords stored in DB and config.properties. This encryption key has to be stored in tomcat’s ***catalina.properties*** file.

Go to the tomcat folder where it is installed .

- Go to the conf folder under the tomcat directory .
- Open the catalina.properties file using vi editor and add the below lines

PASSWORD_KEY=<encryption_key>

Config.properties changes:

Following property values in config.properties file have to be encrypted and put.

avanseus.mongodb.password

avanseus.mongodb.keystore.password

avanseus.mongodb.admin.password

avanseus.ldap.bindPassword

avanseus.ldap.keystore.password

Application-Level Changes:

The API of the above has been developed for the encryption and decryption of the passwords which we want to store in the encrypted form in the database.

Code Changes:

Create the instance of the class using the code below given.

```
try {
        StandardPasswordEncryptorUtil standardPasswordEncryptorUtil =
        StandardPasswordEncryptorUtil.getInstance();

  } catch (ApplicationException e) {

                e.printStackTrace();

        }
```

Encryption: This has to be done before storing the values in the database

```
String encrypted = standardPasswordEncryptorUtil.encrypt("Avanseus");

System.out.println(encrypted);
```

Decryption: This has to be done while extracting the encrypted values from the database.

```
String decrypted = standardPasswordEncryptorUtil.decrypt(encrypted);

System.out.println(decrypted);
```

Encryption and Decryption at Unix Terminal.

Download the Jar file from the link provide

https://github.com/jasypt/jasypt/releases/download/jasypt-1.9.3/jasypt-1.9.3-dist.zip

The above jar file is a JASYPT jar which is used for the encryption and decryption of the input provided.

- Download the jar from the above link and extract the files.
- Give the execute permission to the files inside the bin folder.

Commands to encrypt and decrypt the given input.

- Go to the folder where you have extracted the file.
- Go to the /bin/ directory.

Encryption:

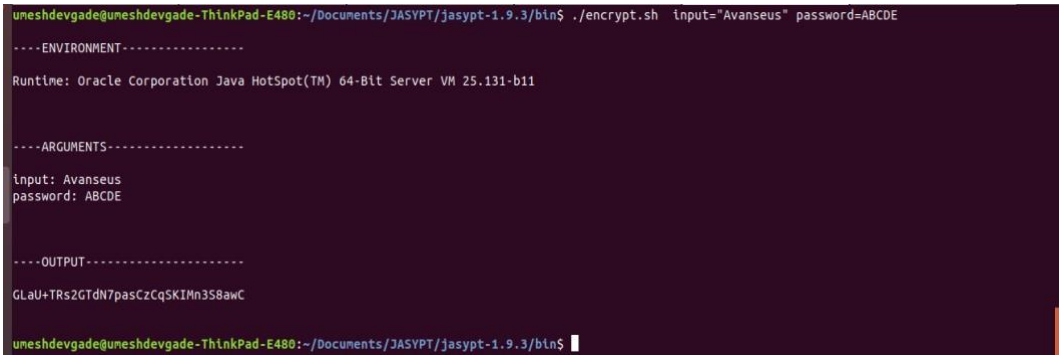
Command: ./encrypt.sh input="text_to_be_encrypted" password=key_for_encryption

Input: The input which you want to encrypt.

Password: This is the key which is used for the encryption. The same key will be used for decryption as well.

Example:

./encrypt.sh input="Avanseus" password=ABCDE



Decryption:

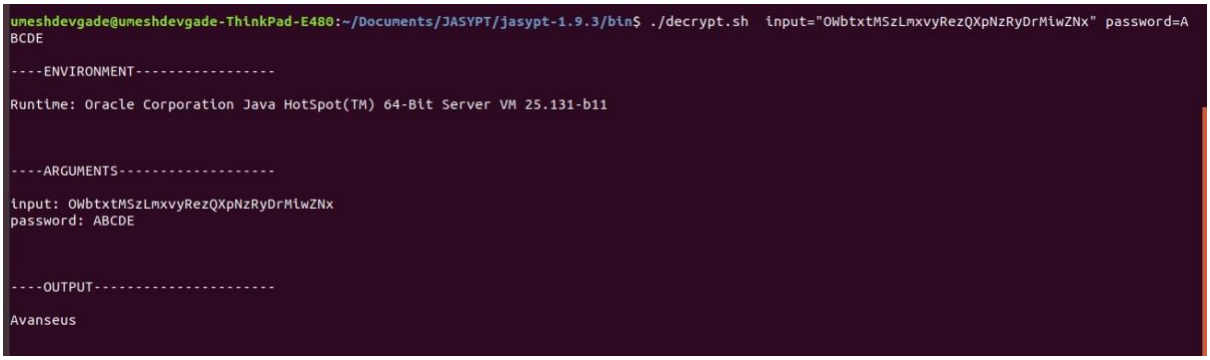
Command: ./decrypt.sh input="encrypted_text" password=Key_used_for_encryption

Input: Input here is the values you want to decrypt.

Password: The key which was used for the encryption. The same key should be used for decryption as well.

Example:

./decrypt.sh input="OWbtxtMSzLmxvyRezQXpNzRyDrMiwZNx" password=ABCDE



Note: Please put the values of “input” & “password” in single quotes if there are special characters in them. Please also verify the encrypted text by decrypting it always.