УДК 004.056.55

студент В.П. Мойсеенко
ГВУЗ «Национальный горный университет»

ШИФРОВАНИЕ В JAVA. АРХИТЕКТУРА, ВОЗМОЖНОСТИ И ПРИМЕНЕНИЕ

В данных тезисах содержатся основные понятия об архитектуре классов пакета Java JCE. В тексте приведены примеры использования пакета JCE для симметричного шифрования данных.

Язык Java, начиная с версии 1.2, обладает встроенным набором классов поддержки криптографии. С течением времени и выходом новых версий языка, поддержка криптографии расширялась, дополняясь новыми алгоритмами и стандартами.

В версии JAVA SE 6 все классы, реализующие криптографические функции объединены в пакете javax.crypto. Данный пакет содержит 18 основных классов, 1 интерфейс, а также 5 дополнительных классов для поддержки исключительных ситуаций.

Компания Sun, начиная с версии Java SE 1.2 выделила классы поддержки криптографии в отдельный фреймворк под названием JCE (Java Cryptography Extensions), тем самым обеспечив независимое развитие JCE. Основные заявленные принципы архитектуры фреймворка [3,с.5]:

– Независимость организации и взаимодействия;

– Независимость от конкретного алгоритма, возможность добавления новых алгоритмов.

На сегодняшний день поддерживаются следующие симметричные алгоритмы шифрования: RC2, RC5, Blowfish, DES, Triple DESede, AES Rijndael[4,с.20].

Однако возможности JCE не ограничиваются лишь симметричными алгоритмами шифрования. Данный фреймворк предоставляет полный инструментарий для создания электронных цифровых подписей и сертификатов с использованием асимметричной криптографии.

Рассмотрим подробнее симметричное шифрование, его возможности и использование в прикладных приложениях.

Стремясь обеспечить максимальную независимость работы от определенного алгоритма шифрования, разработчики JCE предприняли следующий шаг: строковое задание типа алгоритма и параметров его работы, с последующей инициализацией универсального класса со стандартным набором методов, которые не зависят от алгоритма шифрования.

В процессе шифрования\дешифрования участвуют понятия «ключ» и «алгоритм шифрования», соответственно в JCE эти понятия представлены в виде классов Key и Cipher. Инициализация алгоритма шифрования происходит следующим образом:

```
Cipher aes_cipher = Cipher.getInstance(“AES/ECB/NoPadding”);
```

Рассмотрим строковый параметр:

– AES – тип алгоритма;

– ECB – режим работы алгоритма (в данном случае используется режим «электронная кодовая книга»);

– NoPadding – тип заполнения.

В общем случае можно выделить два типа строкового параметра для инициализации алгоритма шифрования:

– «алгоритм/режим/заполнение»;

– «алгоритм».

При инициализации и задании недопустимых или неверных параметров алгоритма шифрования возникают следующие исключительные ситуации на этапе инициализации: NoSuchAlgorithmException, NoSuchPaddingException.

Инициализация ключа требует передачу типа алгоритма шифрования, как строкового параметра конструктора, а также массива байт в качестве секретного ключа.

Байтовый массив должен быть соответствующей длины для выбранного алгоритма шифрования, в противном случае при операции шифрования / дешифрования возникнет исключительная ситуация IllegalBlockSizeException.

```
String key_string=”Sonnenuntergang!”;
```

```
SecretKey secret_key = new
SecretKeySpec(key_string.getBytes(),”AES”);
```

Вариант функции шифрования будет выглядеть следующим образом:

```
public String encrypt(String str)
{
     Cipher cipher = Cipher.getInstance(CIPHER);
     cipher.init(Cipher.ENCRYPT_MODE, secret_key);
     String res = new String(cipher.doFinal(str.getBytes()));
     return res;
}
```

При этом необходимо учитывать требование, чтобы длина шифруемой строки удовлетворяла требованиям алгоритма, в противном случае возникнет исключительная ситуация IllegalBlockSizeException.

Вариант функции дешифрования аналогичен функции шифрования. Отличие состоит в константах, задающих режим шифрования или дешифрования. Этими константами могут быть Cipher.ENCRYPT_MODE, Cipher.DECRYPT_MODE. Соответственно при дешифровании строки следует использовать константу Cipher.DECRYPT_MODE.

В приведенных примерах показано удобство работы с JCE, а также широкие возможности его применения. JCE применяется в таких известных проектах как Sun Glassfish, Apache-SSL, а также в ряде других веб-серверов и приложений.

Перечень литературы:

1. Хорстман К., Корнел Г. Java2.Том 1 М.: Вільямс. 2006. 890 с.
2. Хорстман К., Корнел Г. Java2.Том 2 М.: Вільямс. 2006. 1166 с.
3. Java Cryptography Architecture API Specification & Reference. 2002
4. Java Cryptography Extension (JCE) Reference Guide for the Java 2 SDK, Standard Edition, v 1.4. 2003