GLOBAL 2007

Обучение Java

Тема **Безопасность в JAVA**

Часть **Подпись jar-файлов**

Автор **ASKIL (omendba@gmail.com)**

26.02.2007

Ключи – важный момент в модели безопасности Java, потому что они позволяют создавать или проверять цифровую подпись.

Цифровая подпись создается частным ключом, затем передается с помощью электроники. Когда цифровая подпись получена, она должна быть проверена с помощью публичного ключа, который соответствует частному ключу, с помощью которого была создана подпись.

Цель ключевой системы управления двойная. Когда необходимо подписать цифровой подписью что-нибудь, ключевая система управления должна иметь частный ключ, с помощью которого будет производиться подпись. Когда необходимо проверить цифровую подпись, ключевая система управления должна иметь публичный ключ, который будет использоваться для проверки.

Существуют три элемента ключевой системы управления:

• **Ключи**

Ключи в ключевой системе управления могут использоваться для криптографических целей, но вообще их можно использовать для подписи данных, типа файлов JAR.

• **Сертификаты**

Сертификаты используются для проверки связи между публичным ключом и юридическим лицом. Проверка цифровой подписи требует публичного ключа, принадлежащего юридическому лицу, которое создавало цифровую подпись; сертификат проверяет, что публичный ключ сам не был подделан и действительно принадлежит доверяемому лицу.

• **Субъекты**

Субъекты – абстракция людей, компаний, или любого другого юридического лица, которое могло бы иметь ключ. Цель ключевой системы управления состоит в том, чтобы связать субъекты с их ключами. Эта связь должна быть сохранена в ключевой базе данных keystore.

Для создания и управления сертификатами предусмотрена программа keytool. Эта программа управляет хранилищами ключей, базами данных сертификатов и закрытых ключей. Каждый элемент хранилища ключей имеет псевдоним.

В программе keytool для идентификации владельцев ключей и создателей сертификатов используются имена стандарта X.500:

CN – общее имя;

OU – организация подразделений;

O – организация;

L – город;

ST – область, штат, провинция;

C – страна.

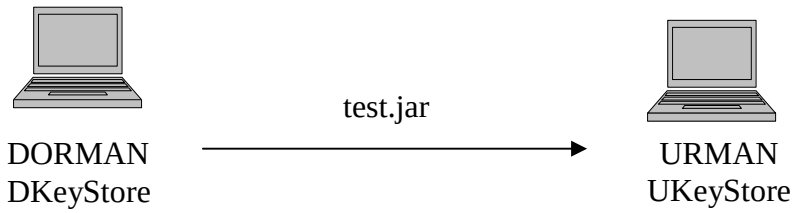
При работе с программой keytool необходимо понимать следующие признаки:

• Псевдоним (alias). Это – имя, которое можно использовать для того, чтобы сослаться на юридическое лицо в базе данных keystore.

• Сертификаты, которые ручаются за идентичность субъекта. Они также предусматривают публичный ключ.

• Частный ключ. Частный ключ может быть защищен паролем.

Подпись jar-файлов



URMAN хочет удостовериться, что присланный test.jar является неподдельным.

DORMAN создает хранилище ключей DKeyStore, а URMAN – UkeyStore.

```
keytool -genkey -keystore c:/DkeyStore -alias DorAlias -keyalg RSA
Enter keystore password:  PASSWORD1
What is your first and last name?
  [Unknown]:  dor
What is the name of your organizational unit?
  [Unknown]:  dor
What is the name of your organization?
  [Unknown]:  dor
What is the name of your City or Locality?
  [Unknown]:  dor
What is the name of your State or Province?
  [Unknown]:  dor
What is the two-letter country code for this unit?
  [Unknown]:  dor
Is CN=dor, OU=dor, O=dor, L=dor, ST=dor, C=dor correct?
  [no]:  yes

Enter key password for <DorAlias>
        (RETURN if same as keystore password):  PASSWORD1
```

```
keytool -genkey -keystore c:/UkeyStore -alias UrmAlias -keyalg RSA
Enter keystore password:  PASSWORD2
What is your first and last name?
  [Unknown]:  urm
What is the name of your organizational unit?
  [Unknown]:  urm
What is the name of your organization?
  [Unknown]:  urm
What is the name of your City or Locality?
  [Unknown]:  urm
What is the name of your State or Province?
  [Unknown]:  urm
What is the two-letter country code for this unit?
  [Unknown]:  urm
Is CN=urm, OU=urm, O=urm, L=urm, ST=urm, C=urm correct?
  [no]:  yes

Enter key password for <UrmAlias>
        (RETURN if same as keystore password):  PASSWORD2
```

Посмотрим, что получится, если DORMAN в свое хранилище добавит еще одну пару ключей.

```
keytool -genkey -keystore c:/DkeyStore -alias DorAlias2 -keyalg DSA
Enter keystore password:  PASSWORD1
What is your first and last name?
  [Unknown]:  dor2
What is the name of your organizational unit?
  [Unknown]:  dor2
What is the name of your organization?
  [Unknown]:  dor2
What is the name of your City or Locality?
  [Unknown]:  dor2
What is the name of your State or Province?
  [Unknown]:  dor2
What is the two-letter country code for this unit?
  [Unknown]:  dor2
Is CN=dor2, OU=dor2, O=dor2, L=dor2, ST=dor2, C=dor2 correct?
  [no]:  yes

Enter key password for <DorAlias2>
        (RETURN if same as keystore password):  PASSWORD1
```

Все сертификаты DKeyStore в машинном представлении:

```
keytool -list -rfc -keystore c:/DKeyStore
Enter keystore password:  PASSWORD1

Keystore type: jks
Keystore provider: SUN

Your keystore contains 2 entries

Alias name: doralias
Creation date: 25.02.2007
Entry type: keyEntry
Certificate chain length: 1
Certificate[1]:
-----BEGIN CERTIFICATE-----
MIICGjCCAYMCBEXhnqAwDQYJKoZIhvcNAQEEBQAwVDEMMAoGA1UEBhMDZG9yMQwwCgYDVQQIEwNk
b3IxDDAKBgNVBAcTA2RvcjEMMAoGA1UEChMDZG9yMQwwCgYDVQQLEwNkb3IxDDAKBgNVBAMTA2Rv
cjAeFw0wNzAyMjUxNDM1MTJaFw0wNzA1MjYxNDM1MTJaMFQxDDAKBgNVBAYTA2RvcjEMMAoGA1UE
CBMDZG9yMQwwCgYDVQQHEwNkb3IxDDAKBgNVBAoTA2RvcjEMMAoGA1UECxMDZG9yMQwwCgYDVQQD
EwNkb3IwgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAJ2giDbbhhUy2/mDhEmklJtmlWjIKfw9
bq+eoFrd+QCMJ00im++xV8foTcKkpmBXclU+a7mO6uAFuHs0bnoqLeLqjwWgcLLcRb4SyUWc6zRq
cyVVNI0cb3IyKa69WERpHw9CowFfbloW9DRuftzWkcKRs/sMdmjKb0yh2eeFC6QLAgMBAAEwDQYJ
KoZIhvcNAQEEBQADgYEADmBz3SXVFnX6Bbu0wLIszfmKNxcHziXyLzhIOmFcUXW9UtbaFJzLIBPU
YCPKzg6eBWxmeJiUsiPcMd1NcFF6Uocc18tO+OAq2t9Ta4bVIOwkV/+PL2Vd9hMbq3K1EXo24LXF
ZfteBXpdSaDCwb951X5ah9pzMwSjvGy1fYHdeUg=
-----END CERTIFICATE-----


****************************************
****************************************


Alias name: doralias2
Creation date: 25.02.2007
Entry type: keyEntry
Certificate chain length: 1
Certificate[1]:
-----BEGIN CERTIFICATE-----
MIIC6DCCAqYCBEXhnzEwCwYHKoZIzjgEAwUAMFoxDTALBgNVBAYTBGRvcjIxDTALBgNVBAgTBGRv
cjIxDTALBgNVBAcTBGRvcjIxDTALBgNVBAoTBGRvcjIxDTALBgNVBAsTBGRvcjIxDTALBgNVBAMT
BGRvcjIwHhcNMDcwMjI1MTQzNzM3WhcNMDcwNTI2MTQzNzM3WjBaMQ0wCwYDVQQGEwRkb3IyMQ0w
CwYDVQQIEwRkb3IyMQ0wCwYDVQQHEwRkb3IyMQ0wCwYDVQQKEwRkb3IyMQ0wCwYDVQQLEwRkb3Iy
MQ0wCwYDVQQDEwRkb3IyMIIBtzCCASwGByqGSM44BAEwggEfAoGBAP1/U4EddRIpUt9KnC7s5Of2
```

```
EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7
ECLCT7up1/63xhv4O1fnxqimFQ8E+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUA
l2BQjxUjC8yykrmCouuEC/BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdR
WVeOutRZT+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAMX6e8octDxun1AJU
S8w7W/BVyzBihUDvxUBzYiYPcTprr+eXQJovmsOnb5x4junBEbpb64n2fcF7UwregGeVkoHuvwIw
6KkAOWTCNz96bLxfJFHTioXt+gKwi2nPcP9WPaghz2GfA8WK7UXNe0wlb6fJBDZQs5elRqe3DDIR
mlcwCwYHKoZIzjgEAwUAAy8AMCwCFDRwXswDvAhOFOsp+n3mGdKR2a/EAhQ+Z4+yrFKgJDCoz8Ib
StKQ5K2WVA==
-----END CERTIFICATE-----


*****************************************
*****************************************
```

DORMAN должен экспортировать открытый ключ в виде сертификата и отослать его URMAN. При пересылке открытого ключа, его могут перехватить и подделать. Поэтому открытый ключ снабжается отпечатком. После получения публичного ключа, URMAN должен позвонить DORMAN и сверить отпечатки. Если они совпадают, значит, публичный ключ именно от DORMAN, а не от злоумышленника.

DORMAN экспортирует публичный ключ:

```
keytool -export -keystore c:/DKeyStore -alias DorAlias -file c:/DorPubCert.crt
Enter keystore password:  PASSWORD1
Certificate stored in file <c:/DorPubCert.crt>
```

DORMAN смотрит отпечаток публичного ключа (дайджест с помощью алгоритмов MD5 и SHA1):

```
keytool -printcert -file c:/DorPubCert.crt
Owner: CN=dor, OU=dor, O=dor, L=dor, ST=dor, C=dor
Issuer: CN=dor, OU=dor, O=dor, L=dor, ST=dor, C=dor
Serial number: 45e19ea0
Valid from: Sun Feb 25 17:35:12 MSK 2007 until: Sat May 26 18:35:12 MSD 2007
Certificate fingerprints:
         MD5:  9A:40:46:01:B5:56:28:D0:9D:AA:8F:F6:5D:FB:6E:54
         SHA1: 33:F4:CA:F1:59:5B:DB:38:1A:5F:2A:89:8A:CC:48:08:83:B5:72:29
```

URMAN, получив открытый ключ от DORMAN, должен проделать то же самое. Если отпечатки совпадают, то публичный ключ получен без изменений от DORMAN.

Все теперь можно подписывать jar-файлы. Итак, DORMAN необходимо подписать файл MegaCrypt.jar:

```
jarsigner -keystore c:/DKeyStore -signedjar c:/SignMegaCrypt.jar
          c:/MegaCrypt.jar DorAlias
Enter Passphrase for keystore: PASSWORD1

Warning: The signer certificate will expire within six months.
```

На диске C:/ появится новый файл SignMegaCrypt.jar. Основное отличие от неподписанного файла состоит в том, что в папке META-INF появились два файла: DORALIAS.RSA и DORALIAS.SF.

Содержимое файла DORALIAS.SF можно просмотреть:

```
Signature-Version: 1.0
Created-By: 1.5.0_09 (Sun Microsystems Inc.)
SHA1-Digest-Manifest-Main-Attributes: nllk/HTMtMIyJNQjOypZ+SsdU0I=
SHA1-Digest-Manifest: NGkdmZQ6nNtc0XU8+yPQa7g2v4k=

Name: megacrypt/Main.class
SHA1-Digest: 72wzk434ji058e6UQea5kyD5ckY=

Name: megacrypt/ClassCrypt.class
SHA1-Digest: DKowwOuFAsvyIfQFcsM9L8x+iZw=
```

Также в главном файле MANIFEST.MF появились строки:

```
Name: megacrypt/Main.class
SHA1-Digest: Ky2ZmMlSloy33mEkDBX8+XVklP4=

Name: megacrypt/ClassCrypt.class
SHA1-Digest: Xmj3x77EvgLJwPTdyV6voX2qnzI=
```

URMAN, получив данный jar-файл теперь может проверить его цифровую подпись:

```
jarsigner -verbose -certs -verify -keystore c:/UKeyStore c:/SignMegaCrypt.jar

         360 Sun Feb 25 17:57:36 MSK 2007 META-INF/MANIFEST.MF
         344 Sun Feb 25 17:57:36 MSK 2007 META-INF/DORALIAS.SF
         866 Sun Feb 25 17:57:36 MSK 2007 META-INF/DORALIAS.RSA
           0 Sun Feb 25 16:37:06 MSK 2007 META-INF/
           0 Sun Feb 25 16:37:06 MSK 2007 megacrypt/
sm      3104 Sun Feb 25 16:37:06 MSK 2007 megacrypt/ClassCrypt.class

      X.509, CN=dor, OU=dor, O=dor, L=dor, ST=dor, C=dor
      [certificate will expire on 26.05.07 18:35]

sm       510 Sun Feb 25 16:37:06 MSK 2007 megacrypt/Main.class

      X.509, CN=dor, OU=dor, O=dor, L=dor, ST=dor, C=dor
      [certificate will expire on 26.05.07 18:35]


  s = signature was verified
  m = entry is listed in manifest
  k = at least one certificate was found in keystore
  i = at least one certificate was found in identity scope

jar verified.

Warning: This jar contains entries whose signer certificate will expire
within six months.
```

Если хакер изменил код, то получим следующее сообщение:

```
jarsigner -verbose -certs -verify -keystore c:/UKeyStore c:/SignMegaCrypt.jar
jarsigner: java.lang.SecurityException: SHA1 digest error for
megacrypt/ClassCrypt.class
```

Приложения на Java должны иметь возможность создавать и проверять цифровые подписи у классов. Подписанные классы позволяют расширить возможности песочницы Java-машины двумя различными способами:

• Файл политики может настаивать, чтобы классы, прибывающие из сети, были подписаны юридическим лицом прежде, чем диспетчер доступа предоставит им доступ к системе. В файле политики должна содержаться следующая директива signedBy:

```
grant signedBy "ABC", codeBase "http://www.abc.com/" {
             java.io.FilePermission "-", "read,write";
};
```

Эта запись позволяет классам, которые были загружены с узла www.abc.com, читать и писать любые локальные файлы при условии, что данные классы были подписаны ABC.

• Менеджер безопасности может взаимодействовать с загрузчиком классов для того, чтобы определить, действительно ли данный класс подписан. Менеджер безопасности тогда свободно предоставляет разрешение.

Есть три необходимых компонента для расширения возможностей песочницы Java с подписанными классами:

1. Метод подписи классов с использованием jarsigner.
2. Загрузчик классов, который знает, как управлять цифровой подписью, связанной с классом.
3. Менеджер службы безопасности или диспетчер доступа, который предоставляет желаемые разрешения, основанные на цифровой подписи.

Подписанный файл Jar имеет три специальных элемента:

1. Декларация (MANIFEST.MF). Содержит список файлов в архиве, которые были подписаны.
2. Файл подписи (XXX.SF, где XXX – имя юридического лица, которое подписало архив), который содержит саму подпись.
3. Блочный файл (XXX.DSA, где XXX – имя юридического лица, которое подписало архив и DSA – название алгоритма подписи). Блочный файл содержит данные подписи в формате PKCS7.

Можно отметить, что для создания подписей PKCS7 в стандартной поставке J2SE нет таких классов, которые смогли бы это сделать.