GLOBAL 2007

Обучение Java

Тема **Безопасность в JAVA**

Часть **Шифрование на основе пароля**

Автор **ASKIL (omendba@gmail.com)**

5.03.2007

Шифрование на основе пароля (PBE) получает ключ шифрования из пароля. Чтобы получить ключ из пароля, отнимающего много времени у хакера, большинство PBE алгоритмов используют случайное число salt для создания ключа.

Чтобы использовать шифрование на основе пароля (PBE) как определено в PKCS#5, мы должны определить salt и итеративный счетчик.

В данном случае сохранять ключ на диск не имеет смысла, т.к. в ключе содержится секретный пароль, и объект SecretKey может его выдать.

Создать симметричный ключ можно с помощью следующей процедуры:

```
    /**
     * Создать симметричный ключ.
     * @param p <B>Пароль</B>
     * @param alg <B>Алгоритм ключа</B>
     * @return Ключ
     */
    public SecretKey CreateSymmetricKeyWithPassword(char [] p, String alg)
    {
      try
      {
       PBEKeySpec pbeKeySpec = new PBEKeySpec(p);
       Arrays.fill(p,'0');
       SecretKeyFactory kf = SecretKeyFactory.getInstance(alg);
       return kf.generateSecret(pbeKeySpec);
      }
      catch(Exception e){GetError(e.getMessage()); return null;}
    }
```

Процедура шифрования файла на диске:

```
    /**
     * Зашифровать файл.
     * @param sk <B>Ключ</B>
     * @param infile <B>Файл для шифрования</B>
     * @param outfile <B>Шифрованный файл</B>
     * @param alg <B>Алгоритм ключа</B>
     */
    public void CryptFile(SecretKey sk, String infile, String outfile, String alg)
    {
      try
      {
        Cipher pbeCipher = Cipher.getInstance(alg);
        //Смесь для шифрования
        byte [] salt = { (byte)0x70, (byte)0x43, (byte)0xdd, (byte)0x34,
                         (byte)0xfa, (byte)0x98, (byte)0xaa, (byte)0x67};
        int count = 100; //Число итераций
        PBEParameterSpec pbeParameterSpec = new PBEParameterSpec(salt, count);
        pbeCipher.init(Cipher.ENCRYPT_MODE, sk, pbeParameterSpec);
        //-------------------------------------------------------------------//
        FileInputStream in = new FileInputStream(infile);
        FileOutputStream out = new FileOutputStream(outfile);
        CipherOutputStream cout = new CipherOutputStream(out, pbeCipher);
        byte [] buffer = new byte[pbeCipher.getBlockSize()];
        int bytefromfile = in.read(buffer); //Узнали, сколько байт прочитано из потока
        while (bytefromfile!= -1)
        {
          cout.write(buffer,0,bytefromfile); //Сколько прочитали, столько и записали
          bytefromfile = in.read(buffer);
        }
        cout.close(); //Шифруем последние байты, число которых <
                      //размера cipher.getBlockSize()
        in.close();
        out.close();
```

```
      }
      catch(Exception e){GetError(e.getMessage());}
    }
```

Процедура дешифрования файла на диске:

```
    /**
     * Расшифровать файл.
     * @param sk <B>Ключ</B>
     * @param infile <B>Файл для дешифрования</B>
     * @param outfile <B>Расшифрованный файл</B>
     * @param alg <B>Алгоритм ключа</B>
     */
    public void DecryptFile(SecretKey sk, String infile, String outfile, String alg)
    {
      try
      {
        Cipher pbeCipher = Cipher.getInstance(alg);
        //Смесь для дешифрования
        byte [] salt = { (byte)0x70, (byte)0x43, (byte)0xdd, (byte)0x34,
                         (byte)0xfa, (byte)0x98, (byte)0xaa, (byte)0x67};
        int count = 100; //Число итераций
        PBEParameterSpec pbeParameterSpec = new PBEParameterSpec(salt, count);
        pbeCipher.init(Cipher.DECRYPT_MODE, sk, pbeParameterSpec);
        //-----------------------------------------------------------------//
        FileInputStream in = new FileInputStream(infile);
        FileOutputStream out = new FileOutputStream(outfile);
        CipherInputStream cin = new CipherInputStream(in, pbeCipher);
        byte[] buffer = new byte[pbeCipher.getBlockSize()];
        int flag = cin.read(buffer); //Узнали, сколько байт прочитано из потока
        while (flag!= -1)
        {
          out.write(buffer,0,flag); //Сколько прочитали, столько и записали
          flag = cin.read(buffer);
        }
        cin.close();
        out.close();
        in.close();
      }
      catch(Exception e){GetError(e.getMessage());}
    }
```

DORMAN зашифровывает файл:

```
      String alg = "PBEWithSHA1AndDESede";
      char [] pass = "PASSWORD1".toCharArray();
      SecretKey sk = CreateSymmetricKeyWithPassword(pass,alg);
      CryptFile(sk,"c:/test.pdf","c:/crypt",alg);
```

URMAN расшифровывает его:

```
      String alg = "PBEWithSHA1AndDESede";
      char [] pass = "PASSWORD1".toCharArray();
      SecretKey sk = CreateSymmetricKeyWithPassword(pass,alg);
      DecryptFile(sk,"c:/crypt","c:/test1.pdf",alg);
```