Segurança Informática e nas Organizações

Guiões das Aulas Práticas

João Paulo Barraca[1] e Hélder Gomes[2]

[1]Departamento de Eletrónica, Telecomunicações e Informática
[2]Escola Superior de Tecnologia e Gestão de Águeda
Universidade de Aveiro

2015–2016

Conteúdo

2

Assinaturas digitais com o Cartão de Cidadão em Java

Resumo:

- Assinaturas digitais.
- Dispositivos PKCS#11
- Assinaturas digitais com o Cartão de Cidadão.

2.1 Introdução

Para a realização deste trabalho é necessário ter o Java instalado no seu computador, tanto o JRE (*Java Run-Time Environment*, necessário para executar aplicações Java), como o JDK (*Java Developement Kit*, necessário para programar aplicações Java). Pode obtê-los na página oficial de distribuição da versão SE (*Standard Edition*) do Java[1]. Poderá também ter instalado um IDE da sua preferência, como o NetBeans[2], por exemplo.

Nota1: O JDK inclui o JRE.
Nota2: O NetBeans e o JDK do Java 8 já estão instalados na máquina virtual da disciplina.

Na realização deste trabalho poderá ser útil visualizar o conteúdo de ficheiros em binário. Para esse efeito, e se estiver em ambiente Linux, pode utilizar a aplicação `GHex`, que já está instalada na máquina virtual da disciplina.

2.2 Software do Cartão de Cidadão

Para a realização deste trabalho é necessário o software do Cartão de Cidadão. Este software já está instalado na máquina virtual da disciplina. Caso não use esta máquina virtual, tem de proceder à sua instalação, podendo descarregá-lo do respetivo sítio (`www.cartaodecidadao.pt`), tendo o cuidado de escolher uma versão adequada ao sistema operativo onde o vai instalar.

2.3 Norma PKCS#11

A norma PKCS#11 define o modo como interagir com dispositivos criptográficos em *smart cards*, como é o caso do cartão de cidadão. O Java inclui um fornecedor (*provider*) de serviços de segurança que concretiza o modelo interação definido pelo PKCS#11, o SUNPKCS11. No entanto, para sistemas **Windows de 64 bits** este *provider* apenas está disponível no JAVA 8. Caso o seu sistema não tenha esta versão de Java instalada, deve fazer a sua atualização ou,em alternativa, mudar para um ambiente Windows de 32 bits

[1] `http://www.oracle.com/technetwork/java/javase/downloads/index.html`
[2] `http://www.netbeans.org`

ou Linux de 32 ou 64 bits.

2.3.1 SUNPKCS11 *provider* e o Cartão de Cidadão

Para utilizar o Cartão de Cidadão através deste *provider* é necessário adicionar a este último uma forma de associar uma biblioteca de funções desenhadas para o Cartão de Ciadão a um nome alusivo ao mesmo, o que é feito através de um *security provider* e de um ficheiro de configuração respetivo.

Consequentemente, deve começar por gerar um ficheiro de configuração com um nome e localização à sua escolha, e com o seguinte conteúdo (para sistemas Linux):

```
name=CartaoCidadao
library=/usr/local/lib/libpteidpkcs11.so
```

A biblioteca indicada faz parte do software do Cartão de Cidadão e concretiza (parte d)a interface PKCS#11 para interagir com esse cartão. Se realizar este trabalho em ambientes Windows o caminho e o nome desta biblioteca são diferentes:

```
name=CartaoCidadao
library=C:\Windows\system32\pteidpkcs11.dll
```

Em seguida deve adicionar ao ficheiro de configuração da segurança do Java (`$JAVA_HOME/lib/security/java.security`, onde `JAVA_HOME` é a pasta `jre*` dentro da pasta do JDK que está a usar) um novo *security provider* e o seu ficheiro de configuração (o que criou anteriormente). Isto é feito adicionando, imediatamente a seguir ao último *provider* listado em `java.security`, uma linha semelhante à seguinte (adapte o número do provider para ser o imediatamente a seguir ao número do último listado no seu ficheiro e adapte o nome e localização do ficheiro de configuração à sua situação específica):

```
security.provider.10=sun.security.pkcs11.SunPKCS11 /home/user/cfgCC_PKCS11
```

Nota: Na máquina virtual da disciplina o `jre` encontra-se na pasta `/opt/java/jdk1.8.0_60/jre`. Para quem não usa a máquina virtual da diciplina, o `jdk` do OpenJDK é tipicamente instalado em `/usr/lib/jvm`.

2.4 Obter o *security provider* do Cartão de Cidadão

Crie um programa que liste todos os *security providers* suportados no seu sistema. Para o efeito inclua as seguintes linhas de código no seu projecto:

```
Provider[] provs = Security.getProviders();
for(int i = 0; i < provs.length; i++){
    System.out.println( i + " - Nome do provider: " + provs[i].getName() );
}
```

Compile e execute o seu programa e veja a lista de *security providers* obtida. Consegue identificar qual o *provider* do Cartão de Cidadão?

2.5 Conteúdo do Cartão de Cidadão

Para se poder realizar operações criptográficas utilizando o Cartão de Cidadão é necessário criar um `KeyStore` e iniciá-lo, o que pode se fazer utilizando o seguinte código (`prov` é uma variável com o *security provider* do Cartão de Cidadão):

```
KeyStore ks = KeyStore.getInstance( "PKCS11", prov );
ks.load( null, null );
```

Utilizando o `KeyStore` pode listar o conteúdo do Cartão de Cidadão (em termos de objetos PKCS#11). Para isso adicione o seguinte código:

```
Enumeration<String> als = ks.aliases();
while (als.hasMoreElements()){
    System.out.println( als.nextElement() );
}
```

A informação que obtém é a identificação dos certificados de chave pública incluídos no Cartão de Cidadão. Através destes identificadores podemos identificar qual o certificado ou chave privada que pretendemos utilizar numa operação criptográfica.

Questão: Na lista de certificados devem constar os dois certificados do cidadão: o de autenticação (CITIZEN AUTHENTICATION CERTIFICATE) e o de assinatura digital (CITIZEN SIGNATURE CERTIFICATE). Porquê a necessidade de dois certificados para o cidadão?

2.6 Assinatura digital

Desenvolva um programa que seja capaz de criar uma assinatura digital de um documento, utilizando o Cartão de Cidadão, e guardá-la num ficheiro próprio, bem como guardar num outro ficheiro o certificado de chave pública correspondente à chave privada com que assinou o documento.

Sugestão: considere a utilização de objetos das classes `KeyStore` e `Signature`, e das interfaces `PrivateKey`, `Certificate`. Use o algoritmo `SHA256withRSA` para realizar a assinatura.

Questão: Se o seu programa realizar multiplas assinaturas sucessivas usando o mesmo Cartão de Cidadão, quantas vezes irá pedir para introduzir o PIN? Faça um programa para fazer a verificação para os dois certificados do cidadão.

Questão: Do ponto de vista de valor legal existe alguma diferença entre as assinaturas produzidas por cada um dos certificados do cidadão?

2.7 Verificação da Assinatura

Desenvolva um programa que seja capaz de verificar uma assinatura digital de um documento. O programa deve ter como entradas o ficheiro com a assinatura digital, o ficheiro com o certificado de chave pública do assinante e o ficheiro original (com o documento que foi assinado). Deve também mostrar no ecrã o nome da entidade que assinou o documento.

Sugestão: considere a utilização da interface `Certificate` e de objetos das classes `Signature`, `X509Certificate` e `CertificateFactory`.

Questão: Será que para verificar a assinatura digital produzida pelo Cartão de Cidadão também temos de usar o *security provider* do cartão de cidadão? Experimente fazer a verificação usando outro *security provider*.

2.8 Cifra

Desenvolva um programa para cifrar um pequeno texto usando uma das chaves públicas do seu Cartão de Cidadão. Explique o que verifica.

2.9 Bibliografia

http://docs.oracle.com/javase/tutorial/security/index.html
http://docs.oracle.com/javase/7/docs/technotes/guides/security/index.
html http://docs.oracle.com/javase/tutorial/security/apisign/index.
html https://mywiki.ncsa.illinois.edu/wiki/Java_PKCS11