

# PKI és SSL mérés

Crysys Lab - oktat@crysys.hu

2008. szeptember 12.

## Tartalomjegyzék

<b>1. Elméleti összefoglaló</b>	<b>2</b>
1.1. PKI . . . . .	2
1.2. SSL . . . . .	2
1.3. Programok digitális aláírása . . . . .	2
<b>2. Mérési környezet</b>	<b>2</b>
2.1. OpenSSL környezet . . . . .	3
2.2. Apache HTTPd környezet . . . . .	3
2.3. Java környezet . . . . .	4
<b>3. Feladatok</b>	<b>4</b>
3.1. Hitelesítő központ felállítása . . . . .	4
3.2. Webkiszolgáló beüzemelése SSL motorral . . . . .	4
3.2.1. Tanúsítvány kérelem készítése a webkiszolgálónak . . . . .	4
3.2.2. Tanúsítvány készítése a webkiszolgálónak a kérelem alapján . . . . .	4
3.2.3. A webkiszolgáló beállítása . . . . .	4
3.2.4. Hitelesítő központ adatainak felvétele a böngészőbe . . . . .	4
3.2.5. SSL hálózati forgalom vizsgálata . . . . .	5
3.2.6. Tanúsítvány igénylése a böngészőnek . . . . .	5
3.2.7. Webkiszolgáló tanúsítványának visszavonása . . . . .	5
3.2.8. Visszavonási lista frissítése a böngészőben . . . . .	5
<b>4. Szorgalmi feladatok</b>	<b>5</b>
4.1. SSL alapú hálózati program fejlesztése . . . . .	5
4.1.1. SSL alapú program készítése . . . . .	5
4.1.2. SSL alapú program üzembehelyezése . . . . .	5
4.1.3. Összegzés . . . . .	5
4.2. Digitálisan aláírt program . . . . .	5
<b>5. Beugró kérdések</b>	<b>6</b>
<b>6. További információk</b>	<b>6</b>
6.1. OpenSSL használata . . . . .	6
6.2. Apache webkiszolgáló . . . . .	7
6.3. Java Keytool . . . . .	7
6.4. Kapcsolódó anyagok, ajánlott olvasmányok . . . . .	8

## 1. Elméleti összefoglaló

A mérés két tárgya közeli kapcsolatban van egymással: az SSL használja a PKI infrastruktúráját, a PKI fogalma az SSL és hasonló protokollok által támasztott igények következtében jött létre.

Minden rendszerben alapvető kérdés a másik fél azonosítása: személyi igazolvánnyal tudjuk igazolni kilétünket ismeretleneknek. Az elektronikus, internetes világban természetesen hasonló igények jelentek meg. Ezen cél megvalósulását segítik a PKI (Public Key Infrastructure) rendszerek.

Az SSL célja a PKI által biztosított elemekkel történő szabványos kommunikáció, melynek segítségével nem biztonságos csatornán keresztül is tud két fél egymással kommunikálni.

### 1.1. PKI

A PKI rendszerek a mindennapi életben megtalálható igazolványkészítés és kiadás intézményének felel meg. Hitelesítő központok tanúsítványt adnak személyeknek, entitásoknak, hogy más személyek illetve entitások felé igazolni tudják kilétüket, elektronikus formában.

A tanúsítványok tartalmazzak leíró adatokat a tulajdonosra vonatkozóan (név, cím, ...), kriptográfiai kulcsokat, a kibocsátó digitális aláírását (pecsét).

### 1.2. SSL

Az SSL protokoll célja, hogy személyek/entitások elektronikus adatszéréjét hitelesen, titkosítva biztosítsák a rendelkezésre álló tanúsítványok alapján, az igényeknek megfelelően.

Előnye az SSL-nek, hogy az alkalmazásnak transzparens módon nyújtja a kriptográfiai szolgáltatásokat, azaz egyedül a kapcsolat létesítése után az alkalmazás egyszerű TCP kapcsolatként használhatja az összeköttetést a hálózat másik pontján levő alkalmazással.

### 1.3. Programok digitális aláírása

A PKI másik használatos területe a digitális aláírások készítése, a digitális aláírások ellenőrizhetősége. Különösen a férgek és vírusok korában fontos lehet programokat digitálisan aláírni, hogy azok eredete ismert legyen annak érdekében, hogy el lehessen dönteni, szabad futtatni vagy sem, illetve milyen jogosultságokat kap a program (rendszer erőforrásaihoz történő hozzáférés, hálózatkezelés...).

Digitálisan aláírt programok például a Microsoft Windows operációs rendszerek által használt eszközmeghajtók, Java alkalmazások.

## 2. Mérési környezet

A mérés Linux operációs rendszer alatt kell elvégezni, melyen a méréshez szükséges szoftverek megtalálhatók. A mérés sikeres elvégzéséhez szükséges elméleti ismeretek:

- nyilvános és titkos kulcsú titkosítás: különbségek, előnyök, hátrányok, sajátosságok, alkalmazási területek
- lenyomat függvények: tulajdonságok, elvárások
- PKI: felépítés, szabványok, algoritmusok
- SSL protokoll
- Unix felhasználói alapismeretek (alapelvek, fájlok kezelése, bash)
- webes alapismeretek (HTTP protokoll, apache webkiszolgáló)

- TCP/IP hálózati alapismeretek
- Java programozási alapismeretek: IO, hálózatkezelés, JSSE (Java Secure Socket Extension), AWT, Applet

A feladatok elvégzéséhez adottak a szoftverek, melyek alapvető ismerete szintén szükséges:

- OpenSSL: a hitelesítő központ infrastruktúráját biztosítja a mérés során
- Apache HTTPd + mod-ssl: webkiszolgáló a https protokollhoz
- OpenSSH: távoli bejelentkezés a kiszolgálóra
- Debian Linux: a kiszolgálón használt operációs rendszer
- Mozilla vagy Firefox: HTTPS protokoll ügyfél oldali alkalmazása
- JDK 1.4: fejlesztés Java nyelven (java, javac, keytool, jarsigner)

Megjegyzés: az OpenSSL nem alkalmas "éles" hitelesítő központ funkciójának betöltésére (lásd man (1) ca), így csak a mérés labor keretein belül, demonstrációs célokra használatos.

Minden csoportnak önálló környezet áll rendelkezésére, melyet szabadon állíthat be a szükségleteknek megfelelően. Amennyiben egy gépet több csoport használ, a TCP/IP portokat természetesen egyeztetni kell egymással.

## 2.1. OpenSSL környezet

Az OpenSSL környezet a \$HOME/meres-pki-ssl/openssl könyvtárban található meg a szükséges fájlokkal. A beállításokban ezt az útvonalat meg kell adni alapútvonalnak (\$HOME/meres-pki-ssl/openssl/openssl.cnf, [CA\_default] dir opció).

Azon OpenSSL parancsok esetén, melyek az openssl.cnf-ben tárolt adatokat igénylik, a *-config* kapcsolónak a \$HOME/meres-pki-ssl/openssl/openssl.cnf útvonalat kell megadni paraméterül.

## 2.2. Apache HTTPd környezet

Az Apache HTTPd környezet a \$HOME/meres-pki-ssl/apache könyvtárban található. Ezen belül a könyvtárak funkciója a következő:

**conf:** beállítások

**data:** a HTTP protokollal kiszolgált adatok

**log:** napló fájlok

**run:** aktuális futó folyamat állapota

Az **apachectl** segítségével lehet egyszerűen elindítani, leállítani a kiszolgálót, a beállításokat ellenőrizni.

A httpd.conf és apachectl fájlokban testre kell szabni a beállításokat:

- alapkönyvtár (\$HOME/meres-pki-ssl/apache),
- HTTP és HTTPS portok, melyen a kiszolgáló fogadja a kéréseket.

A data/cert könyvtárban található 3 darab fájl, melyek a böngésző tanúsítványok létrehozását hivatottak segíteni.

## 2.3. Java környezet

A Java fejlesztéshez 1.4.2-es fejlesztői és futtató környezet áll rendelkezésre. A forrásfájlok szerkesztésére rendelkezésre álló szerkesztők: vim, joe, nano, emacs, pico, mcedit. Igény esetén a fájlokat le lehet tölteni a munkaállomásra és ott szerkeszteni, majd futtatás céljából feltölteni a lefordított fájlokat.

Az 1.4-es Java környezet tartalmazza a JSSE-t az SSL alapú hálózati kapcsolatok létesítéséhez, így annak beállítása és használata nem igényel többlet munkát.

A Java a tanúsítványokat saját formátumú kulcstárban tárolja, melyet a keytool paranccsal lehet kezelni.

JAR csomagok aláírása a jarsigner paranccsal végezhető el.

A Java fejlesztéshez szükséges fájlok helye a \$HOME/meres-pki-ssl/java könyvtár, a szükséges parancsok (java, javac, keytool, jarsigner) helyen bent van a \$PATH környezeti változóban.

## 3. Feladatok

A kitűzött feladatok célja, hogy az SSL alapú kommunikáció és az ahhoz szükséges PKI infrastruktúra technikai oldalát ismertessék a hallgatónak. A feladatok nem térnek ki a PKI rendszereknél szükséges adminisztratív teendőkre, mint például a tanúsítvány igénylőjének megfelelő azonosítására, a titkos kulcsuk tárolására és kezelésre vonatkozó előírásokra illetve azoknak a betartására, a jogi háttérre.

### 3.1. Hitelesítő központ felállítása

Hozzon létre OpenSSL segítségével egy hitelesítő központot, mely alkalmas kiszolgálók és böngészők számára tanúsítványt kibocsátani!

Milyen lépések szükségesek?

Ügyeljen a kulcs tárolására! Mire kell figyelni? Mekkora kulcsot célszerű használni?

A létrehozott hitelesítő központ miben nem felel meg a valós elvárásoknak?

### 3.2. Webkiszolgáló beüzemelése SSL motorral

#### 3.2.1. Tanúsítvány kérelem készítése a webkiszolgálónak

Készítsen egy tanúsítvány kérelmet webkiszolgálónak! Mire kell a névválasztásnál figyelni? Milyen egyéb szempontok vannak?

#### 3.2.2. Tanúsítvány készítése a webkiszolgálónak a kérelem alapján

A kérelem alapján adjon ki tanúsítványt a hitelesítő központtal a webkiszolgáló számára! Milyen paraméterek lehetségesek a kiállításnál?

#### 3.2.3. A webkiszolgáló beállítása

A kiadott tanúsítvány alapján állítsa be a webkiszolgálót, hogy HTTPS protokollal is ki tudja szolgálni a kérésüket. A nem titkosított kapcsolathoz a 80X0, a titkosítotthoz 80X1-es portot használja, ahol X a mérőhely számát jelenti. Vigyázzon arra, hogy ne fogadja el tartósan a böngésző számára a szerver által küldött tanúsítványt!

Milyen szempontokat kell figyelembe venni a kulcstárolási módjához? Tesztelje a beállításokat böngészővel! Mit tapasztal? Ellenőrizze az SSL kapcsolat létét a hálózati forgalom megfigyelésével!

#### 3.2.4. Hitelesítő központ adatainak felvétele a böngészőkbe

Milyen haszna van a hitelesítő központ adatainak rögzítésének? Milyen előnyei és hátrányai vannak? Milyen esetben célszerű rögzíteni az adatokat a böngészőben, melyekben nem?

### **3.2.5. SSL hálózati forgalom vizsgálata**

Az SSL Handshake lehallgatásával állapítsa, hogy mely opcionális üzenetek kerülnek átvitelre és melyek nem. Magyarázza meg a tapasztaltakat!

Milyen rejtjelezési és integritás védelmi algoritmusban állapodik meg a webböngésző és a kiszolgáló?

### **3.2.6. Tanúsítvány igénylése a böngészőnek**

Igényeljen és bocsásson ki tanúsítvány a böngészőjének! Állítsa be a webkiszolgálót, hogy az oldalakat csak érvényes böngésző tanúsítvány esetén szolgáltatassa ki!

Milyen esetekben célszerű ügyfél oldali tanúsítványt használni? Soroljon fel néhányat!

Mennyiben változott az SSL Handshake folyamata?

### **3.2.7. Webkiszolgáló tanúsítványának visszavonása**

Milyen esetben kerülhet visszavonásra egy tanúsítvány? Vonja vissza a tanúsítványt és frissítse a visszavonási listát!

### **3.2.8. Visszavonási lista frissítése a böngészőben**

Frissítse a visszavonási listát a böngészőben! Mit tapasztalt előtte, mit utána? (Ügyeljen a helyes MIME típus beállításokra!)

## **4. Szorgalmi feladatok**

### **4.1. SSL alapú hálózati program fejlesztése**

Milyen esetben célszerű SSL alapú hálózati programot fejleszteni? Milyen előnyei és hátrányai vannak az SSL alapú programoknak, milyen szempontokra kell figyelni a fejlesztés és üzemeltetés folyamán?

#### **4.1.1. SSL alapú program készítése**

Készítsen egyszerű hálózati programot, mely a bemeneti adatokat a konzolról veszi, a konzolra írja az eredményeket, majd egészítse ki oly módon, hogy a hálózati kommunikáció SSL csatornán keresztül történjen! Milyen módosításokra van szükség? Törekedjen az egyszerűsége, használjon öröklődést!

#### **4.1.2. SSL alapú program üzembehelyezése**

Készítse el a szükséges tanúsítványokat és a Java által használt kulcstárakat! Ellenőrizze a program futását!

#### **4.1.3. Összegzés**

Összegezze a feladatok során szerzett tapasztalatokat, milyen módszereket mely esetekben célszerű használni, milyen előnyökkel, hátrányokkal jár!

### **4.2. Digitálisan aláírt program**

Készítsen egy egyszerű Java kisalkalmazást (applet), mely megnyit egy ablakot, csomagolja be (.jar fájl), majd írja alá digitális! A böngészőben ellenőrizze az eredményt! Ellenőrizze az erőforráshasználat korlátozást! (fájlok, hálózat)

## 5. Beugró kérdések

1. Milyen hosszúságúnak javasolja egy nyilvános kulcsú titkosításhoz használt kulcsokat? röviden indokolja választát!
2. A PKI-nek milyen alkalmazási területei vannak?
3. Milyen módszerrel titkosítana egy teljes CD tartalmát? Hogyan hitelesítené?
4. Soroljon fel legalább 3 szimmetrikus kulcsú titkosító algoritmust! Ismertesse főbb jellemzőit!
5. Soroljon fel legalább 2 kriptográfiai lenyomatfüggvényt! Mik a jellemzőik?
6. Mire használható az SSL? Milyen rétegben (szinten) használható?
7. Mi a JSSE? Mi a JCE és mi a jelentősége?
8. Mi a hitelesítő központ? Mi a tanúsítvány?

## 6. További információk

### 6.1. OpenSSL használata

Kulcspár létrehozása:

```
openssl genrsa -des3 -out mykey.key 512
```

Önaláírt tanúsítvány készítése:

```
openssl req -new -x509 -days 1 -key ca.key -out ca.crt
```

Tanúsítvány kérelem létrehozása:

```
openssl req -new -key server.key -out server.csr
```

Tanúsítvány kiadása:

```
openssl ca -out server.crt -in server.csr
```

Tanúsítvány és kulcs összevonása egy fájlba:

```
openssl pkcs12 -export -inkey client.key -in client.crt -out client.p12
```

Tanúsítvány visszavonása:

```
openssl ca -revoke cert.crt
```

Visszavonási lista készítése:

```
openssl ca -gencrl -out crl.pem
```

Tanúsítvány konvertálása PEM-ről DER formátumba:

```
openssl crl -inform PEM -outform DER -in cert.pem -out cert.der
```

## 6.2. Apache webkiszolgáló

A kiszolgáló indítása:

```
apachectl start
```

A kiszolgáló leállítása:

```
apachectl stop
```

A kiszolgáló újraindítása:

```
apachectl restart
```

A kiszolgáló beállításainak tesztelése:

```
apachectl configtest
```

A beállítások a conf/httpd.conf fájlban találhatók.  
SSLVerifyClient SSLCertificateFile

## 6.3. Java Keytool

Kulcstár létrehozása:

```
keytool -genkey -keystore my.keystore -keysize 1024 -alias alias
```

Tanúsítvány kérelem létrehozása:

```
keytool -certreq -keystore my.keystore -file req.csr -alias alias
```

Tanúsítvány importálása:

```
keytool -import -file cert.der -keystore my.keystore -alias alias
```

Kulcstár listázása:

```
keytool -list -keystore my.keystore
```

Java indítása megadott kulcstárral:

```
java -Djavax.net.ssl.keyStore=my.keystore  
-Djavax.net.ssl.keyStorePassword=password MainClass
```

JAR aláírása:

```
jarsigner -keystore my.keystore applet.jar alias
```

Java indítása megadott tanúsítványtárral:

```
java -Djavax.net.ssl.trustStore=my.keystore  
-Djavax.net.ssl.trustStorePassword=password MainClass
```

SSL debug üzenetek kiírása:

```
java -Djavax.net.debug=ssl MainClass
```

## 6.4. Kapcsolódó anyagok, ajánlott olvasmányok

- **SSL - Secure Socket Layer**  
<http://www.hit.bme.hu/~buttyan/courses/BMEVIHI4372/ssl.pdf>  
[http://www.hit.bme.hu/~buttyan/courses/BMEVIHI9367/SSL\\_TLS.ppt](http://www.hit.bme.hu/~buttyan/courses/BMEVIHI9367/SSL_TLS.ppt)
- **Java security**  
[http://www.hit.bme.hu/~buttyan/courses/BMEVIHI9367/Java\\_security.ppt](http://www.hit.bme.hu/~buttyan/courses/BMEVIHI9367/Java_security.ppt)
- **The PKI page**  
<http://www.pki-page.org>
- **OpenSSL man oldalak**  
man openssl
- **OpenSSL honlap**  
<http://www.openssl.org>
- **Apache webkiszolgáló dokumentációja**  
<http://httpd.apache.org/docs/>
- **mod-ssl dokumentáció**  
<http://www.modssl.org/>
- **Java™ 2 SDK, Standard Edition Documentation**  
<http://java.sun.com/j2se/1.4.2/docs/>
- **Java™ Secure Socket Extension (JSSE) - Reference Guide**  
<http://java.sun.com/j2se/1.4.2/docs/guide/security/jsse/JSSERefGuide.html>
- **Java Secure Socket Extension (JSSE)**  
<http://java.sun.com/products/jsse/>
- **keytool - Key and Certificate Management Tool**  
<http://java.sun.com/j2se/1.4.2/docs/tooldocs/solaris/keytool.html>  
<http://java.sun.com/j2se/1.4.2/docs/tooldocs/windows/keytool.html>
- **java - the Java application launcher**  
<http://java.sun.com/j2se/1.4.2/docs/tooldocs/windows/java.html>  
<http://java.sun.com/j2se/1.4.2/docs/tooldocs/linux/java.html>
- **javac - Java programming language compiler**  
<http://java.sun.com/j2se/1.4.2/docs/tooldocs/solaris/javac.html>  
<http://java.sun.com/j2se/1.4.2/docs/tooldocs/windows/javac.html>
- **jarsigner - JAR Signing and Verification Tool**  
<http://java.sun.com/j2se/1.4.2/docs/tooldocs/solaris/jarsigner.html>  
<http://java.sun.com/j2se/1.4.2/docs/tooldocs/windows/jarsigner.html>
- **The Java Tutorial**  
<http://java.sun.com/docs/books/tutorial/index.html>
- **The Java Tutorial - Trail: Custom Networking**  
<http://java.sun.com/docs/books/tutorial/networking/>
- **Mozilla: NSS and SSL Error Codes**  
<http://www.mozilla.org/projects/security/pki/nss/ref/ssl/sslerr.html>
- **Security Focus: An Introduction to OpenSSL, Part Three: PKI- Public Key Infrastructure**  
<http://www.securityfocus.com/infocus/1466>