Apache-alapú WebDAV szerver, LDAP és SSL szolgáltatásokkal

Saqib Ali

saqib@seagate.com

Offshore XML/XHTML Development

Ez a HOGYAN egy Apache-alapú WebDAV szerver telepítését írja le, LDAP-al az azonosításhoz, és SSL-el a titkosításhoz.

Tartalomjegyzék

1. Bevezetés

Jelen dokumentum célja, hogy felépítsünk egy Apache + MySQL + PHP + WebDAV -alapú webes alkalmazásszervert, amely LDAP szerverek használatával végzi az azonosítást (authentication). A dokumentum felfedi a titkosított LDAP tranzakciókezelés egyes részleteit is.

> **Megjegyzés::** Ha bármilyen problémával találkozol az Apache vagy valamely modul telepítésénél, lépj kapcsolatba velem a <saqib@seagate.com> e-mail címen.

1.1. Erről a dokumentumról

Ez a dokumentum eredetileg 2001-ben készült. Azóta számos frissítés és bővítés történt. Köszönet minden közreműködőnek a frissítésekért és javításokért.

Eme dokumentum XML kódja megtalálható a http://www.xml-dev.com:8080/cocoon/mount/docbook/Apache-WebDAV-LDAP-HOWTO.xml webhelyen.

A dokumentum utolsó változata a http://www.xml-dev.com:8080/cocoon/mount/docbook/Apache-WebDAV-LDAP-HOWTO.html honlapon található.

1.2. Közreműködők

Ha szeretnél közreműködni a HOGYAN karbantartásában, letöltheted az XML kódot a http://www.xml-dev.com:8080/cocoon/mount/docbook/Apache-WebDAV-LDAP-HOWTO.xml webhelyről, és elküldheted a frissített kódot a saqib@seagate.com e-mail címre A SZERZŐK LISTÁJÁBAN ÉS A VÁLTOZÁSOK TÖRTÉNETÉBEN A TE NEVEDDEL :). Ez megkönnyíti számomra a kapcsolatfelvételt mindazokkal akik frissítették/javították a dokumentumot. Köszönöm.

1.3. Mi az Apache?

Az Apache egy nyílt forráskódú http szerver modern operációs rendszerekre, amilyen a UNIX és a Windows NT. Http szolgáltatásokat nyújt a jelenlegi HTTP szabványoknak megfelelően.

Az Apache szabadon/ingyenesen letölthető a http://httpd.apache.org/ webhelyről.

1.4. Mi a WebDAV?

A WebDAV egy Web enabled Distributed Authoring and Versioning, vagyis Web alapú Elosztott Szerzői és Változatnyilvántartó rendszer. Együttműködési környezetet biztosít azoknak a felhasználóknak, akik szerkesztik/karbantartják egy webszerver fájljait. Technikailag a DAV a http protokoll kiterjesztése.

Íme egy rövid leírás a DAV által biztosított bővítésekről:

Felülírási védelem: Zárolási és feloldási mechanizmus az "elveszett frissítés" probléma kiküszöbölésére. A DAV protokoll mind a megosztott, mind a kizárólagos zárolásokat támogatja.

Tulajdonságok: Meta-adatok (cím, tárgy, készítő, stb.)

Nevek karbantartása: Fájlok másolása, átnevezése, mozgatása és törlése.

Hozzáférés-szabályozás (Access Control; AC): A hozzáférés korlátozása bizonyos erőforrásokhoz. Jelenleg a DAV feltételezi az AC meglétét, és nem biztosít túl erős azonosítási mechanizmust.

Változatnyilvántartás: Dokumentumok revíziójának nyilvántartása. Még nem megvalósított.

1.5. Mi a PHP?

A PHP (rekurzív betűszó "PHP: Hypertext Preprocessor"; "PHP: Hiperszöveg Előfeldolgozó") széleskörűen használt, nyíl forráskódú, általános célú szkript-nyelv, amely különösen Web-es fejlesztéseknél alkalmazható és beágyazható a HTML-be.

A PHP megtalálható a http://www.php.net webhelyen.

1.6. Mi a MySQL?

A MySQL a legnépszerűbb nyílt forráskódú SQL adatbázis-kezelő, a MySQL AB fejleszti, terjeszti és támogatja.

A MySQL adatbázismotor letölthető a http://www.MySQL.com/ webhelyről.

1.7. Mire van szükségünk?

A cél eléréséhez szükséges eszközök:

i. C Compiler, például GCC
ii. Apache 2 Web szerver
iii. LDAP Module az Apache-hoz
iv. iPlanet LDAP lib fájlok
v. SSL motor
vi. PHP
vii. MySQL adatbázismotor

Megjegyzés:: Mindezen csomagok szabadon hozzáférhetők és letölthetők a Net-ről.

1.8. Feltételezések

A dokumentum feltételezi, hogy a következők már telepítve vannak a rendszereden:

i. gzip vagy gunzip - megtalálható a http://www.gnu.org webhelyen
ii. gcc és GNU make - megtalálható a http://www.gnu.org webhelyen

1.9. Magyar fordítás

A magyar fordítást Kilián Magdolna (mailto:souly1@freemail.hu_NO_SPAM) készítette (2003.03.28). A lektorálást Szijjártó László (mailto:laca@janus.gimsz.sulinet.hu_NO_SPAM) végezte el (2003.07.09). Utoljára Daczi (dacas) László (mailto:dacas@freemail.hu_NO_SPAM) frissítette (2003.12.10). A dokumentum legfrissebb változata megtalálható a Magyar Linux Dokumentációs Projekt (http://tldp.fsf.hu/index.html) honlapján.

2. Követelmények

Le kell töltened és fordítanod (compile) néhány csomagot. Ez a HOGYAN elmagyarázza a fordítási folyamatot, de tudnod kell forráskódból telepíteni.

2.1. Alapok

Szükséged van Solarisra/Linuxra és GNU CC fordítóra a gépen. A GNU gnzip és GNU tar szintén szükséges.

2.2. Apache 2.0.46

Az Apache egy HTTP szerver, Web-es alkalmazások kiszolgálására használjuk. Töltsd le az Apache 2.0.46 forráskódot a http://www.apache.org/dist/httpd/ webhelyről.

2.3. OpenSSL

Töltsd le az OpenSSL csomagot a http://www.openssl.org/source/ webhelyről. A legutolsó verziót töltsd le. Az OpenSSL telepítést az SSL könyvtárak mod_ssl fordítására használjuk Apache-csal, valamint SSL bizonyítványok karbantartására a webszerveren. Töltsd le az OpenSSL forráskódot gzippelt fájlként a /tmp/downloads könyvtárba.

2.4. iPlanet LDAP programkönyvtár

Töltsd le az iPlanet LDAP SDK csomagot a http://wwws.sun.com/software/download/products/3ec28dbd.html honlapról. Az iPlanet LDAP SDK csomagot fogjuk használni, mert ez tartalmazza az ldaps-hoz szükséges programkönyvtárakat (LDAP az SSL felett).

2.5. mod_auth_ldap

Az mod_auth_ldap csomagot az LDAP támogatás Apache-ba fordítására fogjuk használni. Töltsd le a http://www.muquit.com/muquit/software/mod_auth_ldap/mod_auth_ldap_apache2.html honlapról.

2.6. MySQL adatbázismotor

Töltsd le a soron következő MySQL csomagot a http://www.MySQL.com/downloads/index.html honlapról.

2.7. PHP

Töltsd le a PHP forráskódját a http://www.php.net/downloads.php webhelyről.

3. Telepítés

Először ellenőrizd le néhány telepítési feltétel meglétét, majd kezdd meg a telepítést.

3.1. Telepítési feltételek

Az alkalmazásszerver tervünk szerinti telepítéséhez szükségesek az SSL és LDAP programkönyvtárak. Az SSL motorra is szüksége van az Apach 2.x-nek, az SSL tanúsítványok kezeléséhez/használatához.

3.1.1. iPlanet LDAP SDK

Jelentkezz be root felhasználóként, a su parancs használatával:

```
$ su -
```

Hozzd létre az `/usr/local/iplanet-ldap-sdk.5` könyvtárat. Másold az `ldapcsdk5.08-Linux2.2_x86_glibc_PTH_OPT.OBJ.tar.gz` fájlt a `/tmp/downloads` könyvtárból az `/usr/local/iplanet-ldap-sdk.5` könyvtárba.

```
# mkdir /usr/local/iplanet-ldap-sdk.5
# cp /tmp/downloads/ldapcsdk5.08-Linux2.2_x86_glibc_PTH_OPT.OBJ.tar /usr/local/iplanet-ldap-sdk.5
# cd /usr/local/iplanet-ldap-sdk.5
# tar -xvf ldapcsdk5.08-Linux2.2_x86_glibc_PTH_OPT.OBJ.tar
```

Most az összes szükséges iPlanet LDAP lib fájlnak a megfelelő könyvtárban kell lennie.

3.1.2. OpenSSL motor

Ezután az OpenSSL motort kell telepítened.

Az OpenSSL az SSL/TLS protokoll nyílt forráskódú megvalósítása. Az OpenSSL szükséges az SSL tanúsítványok létrehozásához és kezeléséhez a webszerveren. A telepítés a lib fájlokhoz is szükséges, ezeket az SSL modul az Apache kiszolgálására használja.

Lépj be abba a könyvtárba, ahova az OpenSSL forráskódjának fájljait tetted.

```
 # cd /tmp/download
# gzip -d openssl.x.x.tar.gz
# tar -xvf openssl.x.x.tar
# cd openssl.x.x
# make
# make test
# make install
```

A **make install** lefutása után az openssl futtatható fájljai az `/usr/local/ssl` könyvtárban lesznek.

3.2. MySQL

A MySQL telepítése elég egyszerű. A letöltött futtatható állományokat a megfelelő könyvtárba kell tenni.

Kezdetként hozz létre egy user:group csoportot a MySQL démon számára, majd másold be a fájlokat a megfelelő könyvtárakba.

```
 # groupadd MySQL
# useradd -g MySQL MySQL
# cd /usr/local
# gunzip < /path/to/MySQL-VERSION-OS.tar.gz | tar xvf -
# ln -s full-path-to-MySQL-VERSION-OS MySQL
```

Ezután futtasd az install_db szkriptet, és állítsd be a fájlok jogosultságait.

```
 # cd MySQL
# scripts/MySQL_install_db
# chown -R MySQL .
```

3.2.1. A MySQL indítása

Most indítsd el a MySQL kiszolgálót a telepítés ellenőrzéséhez.

```
 # bin/MySQLd_safe --user=MySQL &
```

Ellenőrizd a MySQL démon futását, a ps -ef parancs használatával. A következő kimenetnek kell megjelennie:

```
# ps -ef | grep MySQL
root      3237     1  0 May29 ?        00:00:00 /bin/sh bin/safe_MySQLd
MySQL     3256  3237  0 May29 ?        00:06:58 /usr/local/MySQL/bin/MySQLd --defaults-extra-file=/u
```

3.2.2. A MySQL leállítása

A MySQL kiszolgáló leállításához kövesd az alábbi útmutatást:

```
# cd /usr/local/MySQL
# ./bin/MySQLadmin -u root -p shutdown
```

3.2.3. A Data Directory helyének meghatározása

A MySQL démon minden információt egy "Data Directory" nevű könyvtárban tárol. Ha követted a fenti útmutatást, a Data Directory megtalálható az `/usr/local/MySQL/data` könyvtár alatt.

A Data Directory helyének meghatározásához használd a **MySQLadmin** segédprogramot, az alábbi módon:

```
# /usr/local/MySQL/bin/MySQLadmin variables -u root --password={your_password} | grep datadir
```

3.3. Apache 2.0

Kezdetnek állíts be néhány FLAGS-et a fordító számára.

```
# export LDFLAGS="-L/usr/local/iplanet-ldap-sdk.5/lib/ -R/usr/local/iplanet-ldap-sdk.5/lib/:/usr/loc
# export CPPFLAGS="-I/usr/local/iplanet-ldap-sdk.5/include"
```

Ezután csomagold ki az Apache 2.0 forrásfájljait, és futtasd a `configure` szkriptet.

```
# cd /tmp/download
# gzip -d httpd-2.0.46.tar.gz
# tar -xvf httpd-2.0.46.tar
# cd httpd-2.0.46
#./configure --enable-so --with-ssl --enable-ssl --enable-rewrite --enable-dav
```

Ezután add ki a make parancsot

```
# make
# make install
```

3.3.1. Az Apache indítása

```
# /usr/local/apache2/bin/apachectl start
```

3.3.2. Az Apache leállítása

```
# /usr/local/apache2/bin/apachectl stop
```

3.4. mod_auth_ldap

Csomagold ki a modauthldap_apache2.tar.gz fájlt.

```
cd /tmp/download
# gzip -d modauthldap_apache2.tar.gz
# tar -xvf modauthldap_apache2.tar
# cd modauthldap_apache2
```

Most állítsd be és telepítsd a mod_auth_ldap csomagot.

```
# ./configure --with-apxs=/usr/local/apache2/bin/apxs  --with-ldap-dir=/usr/local/iplanet-ldap-sdk.5
# make
# make install
```

3.5. Tanúsítvány adatbázis az LDAPS:// részére

Le kell töltened a cert7.db és key7.db adatbázisokat a http://www.xml-dev.com/xml/key3.db és http://www.xml-dev.com/xml/cert7.db webhelyről és el kell helyezned az `/usr/local/apache2/sslcert/` könyvtárban.

3.6. PHP

Csomagold ki a PHP forrásfájlokat.

```
gzip -d php-xxx.tar.gz
tar -xvf php-xxx.tar
```

Állítsd be, majd futtasd a make parancsot.

```
cd php-xxx
./configure --with-MySQL --with-apxs=/usr/local/apache2/bin/apxs
```

Fordítsd le a forráskódot.

```
# make
# make install
```

Másold a php.ini fájlt a megfelelő könyvtárba.

```
cp php.ini-dist /usr/local/lib/php.ini
```

4. A WebDAV szolgáltatások telepítése és beállítása

Ez egy könnyű rész. Ebben a fejezetben engedélyezni fogjuk a WebDAV szolgáltatást az Apache egy főkönyvtárában.

4.1. Módosítások az `/usr/local/apache/conf/httpd.conf` fájlban

Ellenőrizd a következő Apache direktívák meglétét az `/usr/local/apache/conf/httpd.conf` fájlban:

```
Addmodule mod_dav.c
```

Amennyiben nincs benne, add hozzá. Ez jelzi az Apache számára a DAV képesség meglétét. A direktívát mindenképp konténeren (container) kívül kell elhelyezni.

Ezt követően meg kell adnod azt, hogy az Apache hol tárolja a DAVLockDB fájlt. Ez egy zárolási adatbázis a WebDAV-hoz, ezt írhatóvá kell tenned a httpd processz számára.

A DAVLock fájlt én az `/usr/local/apache/var` könyvtárban tárolom. Én ezt a könyvtárat más célokra is használom. Add hozzá a következő sort az `/usr/local/apache/conf/httpd.conf` fájlhoz, annak meghatározásához, hogy a DAVLockDB fájl az `/usr/local/apache/var` könyvtárban van:

```
DAVLockDB       /usr/local/apache/var/DAVLock
```

Az utasítást a tárolón kívül helyezd el.

4.2. Egy DAVLockDB könyvtár létrehozása

Mint fent említettem, egy könyvtárat kell létrehoznod a DAVLockDB fájl számára, majd írhatóvá kell tenned a webszerver folyamat számára. Általában a webszerver folyamat "*nobody*" felhasználói néven fut. Ellenőrizd ezt a következő parancs használatával:

```
ps -ef | grep httpd
```

Az `/usr/local/apache` alatt hozz létre egy könyvtárat, és állítsd be a hozzáférési jogokat, a következő parancsok használatával:

```
# cd /usr/local/apache
# mkdir var
# chmod -R 755 var/
# chown -R nobody var/
# chgrp -R nobody var/
```

4.3. A DAV engedélyezése

A DAV engedélyezése pofonegyszerű. Az Apache főkönyvtára alatti könyvtár DAV engedélyezéséhez, add hozzá annak a bizonyos könyvtárnak a tárolójához a következő direktívát:

```
DAV On
```

Ez engedélyezi a DAV-ot arra a könyvtárra és alkönyvtáraira.

A következő példa beállítás engedélyezi a DAV és LDAP azonosítást/hitelesítést az `/usr/local/apache/htdocs/DAVtest` könyvtárra. Rakd be az `/usr/local/apache/conf/httpd.conf` fájlba.

```
 DavLockDB /tmp/DavLock
<Directory "/usr/local/apache2/htdocs/DAVtest">
Options Indexes FollowSymLinks
AllowOverride None
order allow,deny
allow from all
AuthName "SMA Development server"
AuthType Basic
LDAP_Debug On
#LDAP_Protocol_Version 3
#LDAP_Deref NEVER
#LDAP_StartTLS On
LDAP_Server you.ldap.server.com
#LDAP_Port 389
# Ha az SSL aktív, meg kell adnod az LDAP SSL portot, ez általában 636
LDAP_Port 636
LDAP_CertDbDir /usr/local/apache2/sslcert
Base_DN "o=SDS"
```

```
UID_Attr uid
DAV On
#require valid-user
require valid-user
#require roomnumber "123 Center Building"
#require filter "(&(telephonenumber=1234)(roomnumber=123))"
#require group cn=rcs,ou=Groups
</Directory>
```

4.4. Egy "DAVtest" nevű könyvtár létrehozása

Mint egy korábbi részben említettem, minden DAV könyvtárnak írhatónak kell lennie a webszerver folyamat által. Ebben a példában feltételezzük, hogy a webszerver "*nobody*" név alatt fut. Ez az általános. A felhasználó megtekintéséhez (akinek neve alatt a webszerver fut) használd a

```
# ps -ef | grep httpd
```

parancsot.

Hozz létre egy tesztkönyvtárat "`DAVtest`" néven az `/usr/local/apache2/htdocs` könyvtár alatt:

 # mkdir /usr/local/apache/htdocs/DAVtest

Változtasd meg a hozzáférési jogokat a könyvtárban, az legyen írható-olvasható a httpd folyamat számára. Feltételezve, hogy a httpd "*nobody*" felhasználónév alatt fut, használd a következő parancsokat:

```
# cd /usr/local/apache/htdocs
# chmod -R 755 DAVtest/
# chown -R nobody DAVtest/
# chgrp -R nobody DAVtest/
```

4.5. Az Apache újraindítása

Végül le kell futtatnod az Apache-hoz mellékelt konfigurációs tesztrutint, a `httpd.conf` fájl szintaxisának ellenőrzéséhez:

```
# /usr/local/apache/bin/apachectl configtest
```

Ha hibaüzenetet kapsz, akkor ellenőrizd le, hogy minden utasítást helyesen követtél-e. Ha nem tudod kitalálni a hiba okát, írj nekem (a hibaüzenetet is írd meg) a saqib@seagate.com (mailto:saqib@seagate.com) e-mail címre.

Ha a konfiguráció tesztje sikeres, indítsd el az Apache webszervert:

 # /usr/local/apache/bin/apachectl restart

Most van egy WebDAV engedélyezett Apache szervered LDAP hitelesítéssel és SSL titkosítással.

4.6. A WebDAV protokoll tesztelése

Nagyon fontos, hogy a most telepített WebDAV teljesen összhangban legyen a WebDAV-2 protokollal. Ha nem teljesen kompatibilis, akkor a WebDAV alkalmazások kliens oldala nem fog rendesen működni.

A kompatibilitás teszteléséhez a Litmus nevű eszközt használjuk. A Litmus a WebDAV protokoll tesztelője, amely azt vizsgálja, hogy összhangban van-e egy szerver az RFC2518-ben leírt WebDAV protokollal.

Töltsd le a Litmus forráskódját a http://www.webdav.org/neon/litmus/ webhelyről, majd másold be a /tmp/downloads könyvtárba.

Használd a gzip és tar programokat a kicsomagoláshoz:

```
# cd /tmp/downloads
# gzip -d litmus-0.6.x.tar.gz
# tar -xvf litmus-0.6.x.tar
# cd litmus-0.6.x
```

A Litmus fordítása és telepítése egyszerű:

```
# ./configure
# make
# make install
```

A **make install** parancs a bináris fájlokat az `/usr/local/bin`, a súgó fájljait pedig az `/usr/local/man` könyvtárba teszi.

A most telepített WebDAV szerver teszteléséhez használd a

```
# /usr/local/bin/litmus http://you.dav.server/DAVtest userid passwd
```

parancsot.

5. A WebDAV szerver üzemeltetése

Ebben a részben megvitatjuk a különböző kezelési feladatokat - például LDAP belépés ellenőrzése, és hogyan dolgozunk Apache-on DAV módszerrel.

A legtöbb konfigurációs változást a DAV-hoz a `httpd.conf` fájl használatával tesszük. Ez a fájl az `/usr/local/apache/conf/httpd.conf` könyvtárban található.

A `httpd.conf` egy szöveges konfigurációs fájl, amelyet az Apache használ. Szerkesztéséhez bármely szövegszerkesztőt használhatsz, én leginkább a vi-t szoktam. Készíts egy másolatot erről a fájlról, mielőtt megváltoztatod.

Miután a `httpd.conf` fájlban elvégezted a változtatásokat, az Apache szervert újra kell indítanod az **/usr/local/apache/bin/apachectl restart** paranccsal. Mielőtt újraindítanád, teszteld a httpd.conf érvényességét az **/usr/local/apache/bin/apachectl configtest** paranccsal.

5.1. A DAV megosztott hozzáférésének korlátozása

Az előző részben, amikor létrehoztuk a DAVtest megosztást, az LDAP-ot hitelesítési célból használtuk. Azonban bárki, aki hitelesíti magát, az LDAP- ot használva a felhasználói azonosítójával/jelszavával, hozzáférhet ahhoz a mappához.

A **require** direktíva használatával a httpd.conf fájlban limitálhatod adott egyének vagy csoportok hozzáférését.

Ha megnézed a DAVtest konfigurációt az előző részből :

```
<Directory /usr/local/apache/htdocs/DAVtest>
Dav On
#Options Indexes FollowSymLinks

AllowOverride None
order allow,deny
allow from all
AuthName "LDAP_userid_password_required"
AuthType Basic
<Limit GET PUT POST DELETE PROPFIND PROPPATCH MKCOL COPY MOVE LOCK UNLOCK>
Require valid-user
</Limit>
LDAP_Server ldap.server.com
LDAP_Port 389
Base_DN "o=ROOT"

UID_Attr uid
</Directory>
```

Láthatod, hogy a **require** direktíva értéke **valid-user**. Ami azt jelenti, hogy bármely hitelesített felhasználónak hozzáférése van ahhoz a mappához.

5.1.1. Hozzáférés korlátozása egyéni UID alapján

LDAP UID-t is használhatjuk a DAV mappa hozzáférésének korlátozására.

A **require valid-user** direktíva megváltoztatható **require user 334455 445566** értékre.

Ez a 334455 és 445566 UID-vel rendelkező felhasználókra korlátozza a hozzáférést. Senki másnak nem lesz hozzáférése ehhez a mappához.

5.1.2. Hozzáférés korlátozása csoportok tagjai számára

A **require** direktívát használhatod csoportok tagjai hozzáférésének korlátozására. Ezt megteheted az LDAP csoportok vagy az LDAP szűrők használatával. A szűrőt az LDAP filter szintaxis szerint kell felépíteni.

5.2. Írás jog korlátozása DAV megosztás alatt

Lehetséges a DAV megosztások forrásainak szerkesztését bizonyos személyekre korlátozni, mindemellett bárki megnézheti ezeket az erőforrásokat (például fájlokat - dacas). Ezt könnyen teheted a **<Limit>** címke használatával a `httpd.conf` fájlban.

```
<Directory /usr/local/apache/htdocs/DAVtest>
Dav On
#Options Indexes FollowSymLinks

AllowOverride None
order allow,deny
allow from all
AuthName "LDAP_userid_password_required"
```

```
AuthType Basic
<Limit GET PUT POST DELETE PROPFIND PROPPATCH MKCOL COPY MOVE LOCK UNLOCK>
Require valid-user
</Limit>
LDAP_Server ldap.server.com
LDAP_Port 389
Base_DN "o=ROOT"

UID_Attr uid
</Directory>
```

A **<limit>** megváltoztatásával korlátozhatod adott személy írási jogát:

```
<Limit PUT POST DELETE PROPPATCH MKCOL COPY MOVE LOCK UNLOCK>
Require 334455
</Limit>
```

Alapvetően korlátozzuk a 334455 UID-vel rendelkező felhasználó PUT POST DELETE PROPPATH MKCOL COPY MOVE LOCK és az UNLOCK jogát. Mindenki más képes lesz használni a GET és PROPFIND módszert a forrásokon, de mást nem.

6. Az SSL megvalósítása és használata a HTTP forgalom biztonságossá tételére

Manapság a fájlszerveren tárolt adatok biztonsága nagyon fontos. A kompromittált adatok több ezer dollárba is kerülhetnek egy társaságnak. Az utolsó részben LDAP hitelesítési modult fordítottunk az Apache-ba, hogy biztosítsuk a hitelesítési mechanizmust. Bár a HTTP forgalom nem igazán biztonságos, és minden adat tiszta szövegként jelenik meg - ez azt jelenti, hogy az LDAP hitelesítés (userid/passwd) ugyancsak tiszta szövegként megy át. Ez problémát okozhat. Bárki kutathat ezen userid/passwd párosok után és hozzáférhet a DAV állományhoz. Ennek megelőzéséhez titkosítanunk kell a HTTP forgalmat, valójában a HTTP+SSL vagy HTTPS segítségével. Bármi, ami átmegy a HTTPS-en, titkosított lesz, így az LDAP userid/passwd-ben nem kutakodhatnak. A HTTPS a 443-as porton fut. Az előző rész fordítási folyamatának eredményeként az Apache mindkét porton, a 80-ason (normál HTTP) és 443-ason (HTTPS) is fut. Ha csak a DAV-hoz használod majd a szervert, akkor nagyon ajánlott bezárni a 80-as portot. A HOGYAN ezen részében néhány információt nyújtok az SSL-ről és annak üzemeltetéséről egy Apache HTTP szerveren.

6.1. Az SSL bemutatása

Az SSL (Secure Socket Layer; Biztonsági Alréteg) egy protokoll réteg, amely a hálózati (Network layer) és az alkalmazási rétegek (Application layer) között van. Mint neve is sugallja, az SSL mindenféle forgalom titkosítására használható - LDAP, POP, IMAP és legfőképp HTTP.

Íme egy végletekig leegyszerűsített ábra az SSL-el kapcsolatban álló rétegekről.

```
+----------------------------------------+
```

```
|   LDAP   |    HTTP    |   POP   |   IMAP  |
+-----------------------------------------+
|                   SSL                   |
+-----------------------------------------+
|      Hálózati réteg (Network Layer)      |
+-----------------------------------------+
```

6.1.1. Az SSL-ben használt titkosító algoritmusok

Háromféle titkosítási technológiát használnak az SSL-ben: "nyilvános-titkos kulcs" (Public-Private Key), "szimmetrikus kulcs" (Symmetric Key), és "digitális aláírás" (Digital Signature).

"Nyilvános-titkos kulcs" titkosítás - SSL kapcsolat indítása: Ebben az algoritmusban a titkosítás és a visszafejtés nyilvános-titkos kulcspárral történik. A webszerveré a titkos kulcs, a nyilvános kulcsot pedig a tanúsítványban küldi el a kliensnek.

1. A kliens kéri a HTTPS-t használó Web szervertől a tartalmat.
2. A web szerver válaszol egy Digitális Tanúsítvánnyal (Digital Certificate), amiben benne van a szerver nyilvános kulcsa.
3. A kliens ellenőrzi, hogy lejárt-e a tanúsítvány.
4. Ezután a kliens ellenőrzi, hogy a tanúsítvány hatóság (Certificate Authority; továbbiakban CA), amely aláírta a tanúsítványt, megbízott hatóság-e a böngésző listáján. Ez a magyarázata annak, miért van szükségünk egy megbízott CA-tól kapott tanúsítványra.
5. A kliens ellenőrzi, hogy a webszerver teljes domain neve (Fully Qualified Domain Name) megegyezik-e a tanúsítványon lévő közönséges névvel (Common Name).
6. Ha minden megfelelő, létrejön az SSL kapcsolat.

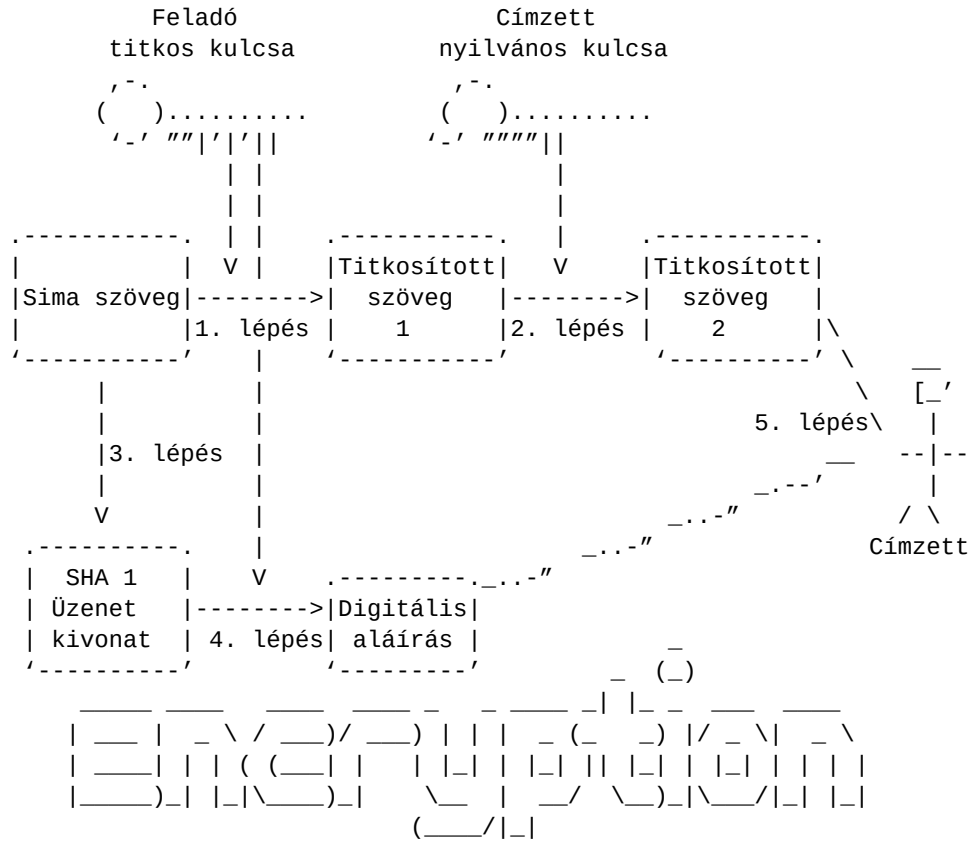
> **Megjegyzés::** Bármi, amit titkos kulccsal titkosítottak, kizárólag a nyilvános kulccsal fejthető vissza. Ennek megfelelően, bármilyen nyilvános kulccsal titkosított dolog, kizárólag a titkos kulccsal fejthető vissza. Elterjedt az a tévhit, miszerint kizárólag nyilvános kulccsal lehet titkosítani és titkos kulccsal visszafejteni. Ez nem így van. Bármelyik kulcs használható titkosításra és visszafejtésre egyaránt (ha annak párját használják visszafejtésre és titkosításra - dacas). Végül is, ha az egyik kulcsot használták titkosításra, a másikat kell használni a visszafejtésre stb. Egy üzenet nem titkosítható és visszafejthető kizárólag a nyilvános kulcs használatával.
>
> *A titkos kulccsal történő titkosítás és a nyilvános kulccsal történő visszafejtés biztosíték a címzetteknek arról, hogy a küldeményt a küldő (a titkos kulcs tulajdonosa) adta fel (mivel a titkos kulcs használatához szükséges jelmondatot csak Ő ismeri - dacas). A nyilvános kulccsal történő titkosítás és titkos kulccsal visszafejtés biztosítja azt, hogy a küldeményt csak a meghatározott címzett (a titkos kulcs tulajdonosa) képes visszafejteni.*

Szimmetrikus titkosítás - az adatok tulajdonképpeni átvitele: Miután az SSL kapcsolat létrejött, szimmetrikus titkosítást használ az adatok titkosítására, kevesebb CPU ciklust felhasználva (tehát kevésbé erőforrásigényes - a lektor). Szimmetrikus titkosításkor az adat ugyanazzal a kulccsal titkosítható és visszafejthető. A szimmetrikus titkosítás kulcsa a kapcsolat indításakor kerül átadásra, a nyilvános-titkos kulcspárral történő titkosítás alatt.

Üzenet ellenőrzés A szerver kivonatot készít az üzenetről valamilyen algoritmus szerint, mint például HMAC, SHA, MD5, majd ezek alapján ellenőrzi az adatok sértetlenségét.

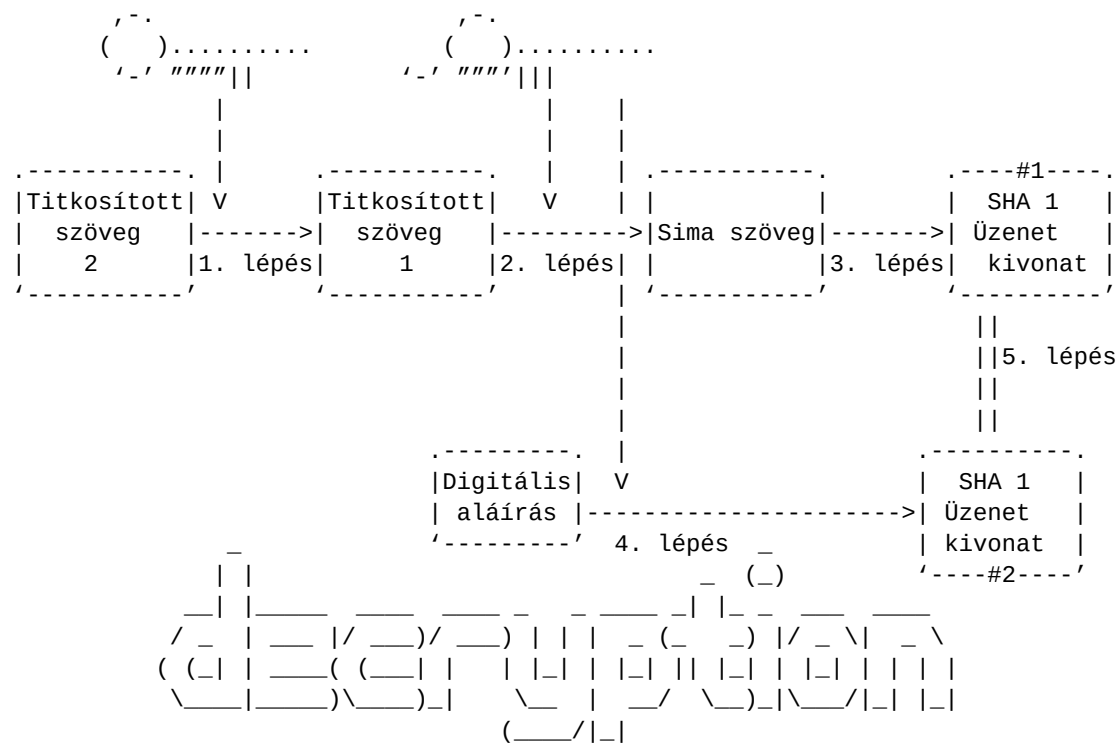
6.1.2. A hitelesség és sértetlenség ellenőrzése

Titkosítási folyamat



- 1. lépés: az eredeti "sima szöveg" titkosítása a feladó titkos kulcsának használatával, ennek eredménye a "titkosított szöveg 1". Ez biztosítja a feladó hitelességét.
- 2. lépés: a "titkosított szöveg 1" titkosítása a címzett nyilvános kulcsával, ennek eredménye a "titkosított szöveg 2". Ez biztosítja a címzett hitelességét (értsd: csak a címzett tudja visszafejteni a szöveget a saját titkos kulcsával).
- 3. lépés: az SHA1 üzenet kivonat (ellenőrző összeg - dacas) készítése a "sima szöveg" alapján.
- 4. lépés: SHA1 üzenet kivonat titkosítása a feladó titkos kulcsával, ennek eredménye a "sima szöveg" digitális aláírása. Ezt a digitális aláírást a címzett felhasználhatja az üzenet sértetlenségének és a feladó hitelességének ellenőrzésére.
- 5. lépés: a "digitális aláírás" és a "titkosított szöveg 2" elküldése a címzettnek.

Visszafejtési folyamat



- 1. lépés: a "titkosított szöveg 2" visszafejtése a címzett titkos kulcsának használatával, ennek eredménye a "titkosított szöveg 1".
- 2. lépés: a "titkosított szöveg 1" visszafejtése a feladó nyilvános kulcsának használatával, ennek eredménye a "sima szöveg".
- 3. lépés: SHA1 üzenet kivonat (ellenőrző összeg - dacas) elkészítése, az előző 2 lépés eredményeként kapott "sima szöveg" alapján.
- 4. lépés: a "digitális aláírás" visszafejtése a feladó nyilvános kulcsának használatával, ennek eredménye az "SHA1 üzenet kivonat".
- 5. lépés: az "SHA üzenet kivonat #1" és "SHA üzenet kivonat #2" összehasonlítása. Amennyiben a kettő egyezik, úgy az üzenet nem módosult az átvitel alatt, így az eredeti "sima szöveg" sértetlen.

6.2. Teszt tanúsítványok

Az Apache fordítása közben létrehoztunk egy teszt tanúsítványt. A mod-ssl csomagban lévő makefile programot használtuk az egyéni Tanúsítvány létrehozásához. Erre a

```
# make certificate TYPE=custom
```

parancsot használtuk.

Ezt a tanúsítványt tesztelési célokra használhatjuk.

6.3. Tanúsítványok "üzemi" használatra

"Üzemi" használathoz szükségünk lesz egy tanúsítványra valamely Certificate Authority-tól (tanúsítvány hatóság) (ezentúl CA). A CA-k a tanúsítványt áruba bocsátók, akik egy megbízható CA listán vannak a felhasználó böngésző kliensében. Mint azt az algoritmus titkosítás részben említettem, ha a CA nincs a megbízott hatóságok listáján, a felhasználó figyelmeztető üzenetet kap, amikor megpróbál kapcsolódni egy biztosított/biztonságos helyhez.

Hasonlóan a teszt tanúsítványokhoz, ez is küld egy figyelmeztető üzenetet a felhasználó böngészőjének.

6.4. Hogyan generálhatsz a CSR-t?

A CSR (TAK) vagy Certificate Signing Request-et (tanúsítvány aláírási kérelem) el kell küldeni egy megbízott CA-nak aláírásra. Ez a rész foglalkozik azzal, hogyan generálhatsz CSR-t és küldheted el egy általad kiválasztott CA-nak. Az **# openssl req** parancs használható erre, az alábbiak szerint:

```
# cd /usr/local/apache/conf/
# /usr/local/ssl/bin/openssl req -new -nodes -keyout private.key -out public.csr
Generating a 1024 bit RSA private key
............++++++
....++++++
writing new private key to 'private.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:California
Locality Name (eg, city) []:San Jose
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Seagate
Organizational Unit Name (eg, section) []:Global Client Server
Common Name (eg, YOUR name) []:xml.seagate.com
Email Address []:saqib@seagate.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:badpassword
An optional company name []:
```

> **"PRNG not seeded" üzenet:** Ha nincs `/dev/random` könyvtár a rendszerünkön, *"PRNG not seeded"* hibaüzenetet kapsz. Ebben az esetben add ki a következő parancsot:
>
> **`# /usr/local/ssl/bin/openssl req -rand some_file.ext -new -nodes -keyout private.key -out public.csr`**
>
> A "some_file.ext" részt cseréljük ki egy rendszerünkön létező fájl nevére. Bármilyen fájlt megadhatunk. Az Openssl ezt fogja véletlen szám generáláshoz használni.

> A Solaris 9 rendszer részeként adnak `/dev/random` fájlt. Amennyiben Solaris rendszert használsz, elképzelhető, hogy telepítened kell a 112438 (http://sunsolve.sun.com/pub-cgi/findPatch.pl?patchId=112438) foltot a /dev/random fájl használatához.

Ezen a ponton pár kérdést tesz fel a szerver helyéről, hogy generálhassa a Certificate Signing Request-et.

Megjegyzés: A közönséges neved (Common Name) a teljes DNS neve (Fully Qualified DNS) a webszerverednek, például dav.server.com. Ha mást írsz oda, akkor NEM fog működni. Jegyezd meg a használt jelszót, a jövőbeli használat érdekében.

Mihelyst befejeződött a folyamat, lesz egy `private.key` és egy `public.csr` fájlod. Szükséged lesz a `public.csr` fájlt bemutatnunk a CA-nak. Ekkor a public.key fájl még nem titkosított. A titkosításhoz használd az

```
 # mv private.key private.key.unecrpyted
# /usr/local/ssl/bin/openssl rsa -in private.key.unecrpyted -des3 -out private.key
```

parancsokat.

6.5. A szerver titkos kulcsának és tanúsítványának telepítése

Miután a CA feldolgozta a kérésed, visszaküldenek egy kódolt tanúsítványt. A Digitális Tanúsítvány formátumát az X.509 v3 szabvány határozza meg. A következőkben látható egy tipikus, X509 v3 szabvány szerinti Digitális Tanúsítvány felépítése:

- Certificate
  - Version
  - Serial Number
  - Algorithm ID
  - Issuer
  - Validity
  -
    - Not Before
    - Not After
  - Subject
  - Subject Public Key Info
  -
    - Public Key Algorithm
    - RSA Public Key

  - Extensions
- Certificate Signature Algorithm
- Certificate Signature

6.5.1. Egy Digitális Tanúsítvány ellenőrzése

Egy X.509 Tanúsítvány ellenőrzésére használd a következő parancsot:

```
# openssl verify server.crt
 server.crt: OK
```

Ahol a `server.crt` a Digitális Tanúsítványt tartalmazó fájl neve.

6.5.2. Egy Digitális Tanúsítvány tartalmának megtekintése

Egy Digitális Tanúsítvány tartalma megtekinthető a **# openssl x509** parancs használatával, az alábbiak szerint:

```
# openssl x509 -text -in server.crt
 Certificate:
     Data:
         Version: 3 (0x2)
         Serial Number: 312312312 (0x0)
         Signature Algorithm: md5WithRSAEncryption
  Issuer: C=US, O=GTE Corporation, CN=GTE CyberTrust Root
         Validity
             Not Before: Feb  8 03:25:50 2000 GMT
             Not After : Feb  8 03:25:50 2001 GMT
      Subject: C=US, ST=New York, L=Pelham, O=xml-dev, OU=web, CN=www.xml-dev.com/Email=saqib@xml-de
         Subject Public Key Info:
             Public Key Algorithm: rsaEncryption
             RSA Public Key: (1024 bit)
                 Modulus (1024 bit):
  ............
  ............
                 Exponent: 65537 (0x10001)
     Signature Algorithm: md5WithRSAEncryption
      ............
  ............
```

6.5.3. A httpd.conf fájl módosítása a tanúsítványok telepítéséhez

Ezt kell elhelyezned a szerveren, és beállítanod az Apache-ban ennek helyét.

Például a titkos kulcsot az `/usr/local/apache2/conf/ssl.key/` könyvtárba, a tanúsítványt pedig az `/usr/local/apache2/conf/ssl.crt/` könyvtárba.

Másold le a tanúsítványt egy `server.crt` nevű fájlba, az `/usr/local/apache2/conf/ssl.crt/` könyvtárba.

Az előző lépésben generált private.key fájlt helyezd az `/usr/local/apache2/conf/ssl.key/` könyvtárba

Ezután módosítsd az `/usr/local/apache2/conf/ssl.conf` fájlt, hogy a megfelelő titkos kulcsra és tanúsítványra mutasson:

```
#   Server Certificate:
#   Point SSLCertificateFile at a PEM encoded certificate.  If
#   the certificate is encrypted, then you will be prompted for a
#   pass phrase.  Note that a kill -HUP will prompt again.  Keep
#   in mind that if you have both an RSA and a DSA certificate you
#   can configure both in parallel (to also allow the use of DSA
#   ciphers, etc.)
SSLCertificateFile /usr/local/apache2/conf/ssl.crt/server.crt
#SSLCertificateFile /usr/local/apache2/conf/ssl.crt/server-dsa.crt

#   Server Private Key:
#   If the key is not combined with the certificate, use this
#   directive to point at the key file.  Keep in mind that if
#   you've both a RSA and a DSA private key you can configure
#   both in parallel (to also allow the use of DSA ciphers, etc.)
SSLCertificateKeyFile /usr/local/apache2/conf/ssl.key/private.key
#SSLCertificateKeyFile /usr/local/apache2/conf/ssl.key/server-dsa.key
```

6.6. A jelmondat (passphrase) eltávolítása az RSA titkos kulcsból

A webszerveren tárolt RSA titkos kulcs általában titkosított, ezért szükséged van egy jelmondatra a használatához. Ezért kér jelmondatot, mikor az Apache-ot modssl-el indítod:

```
# apachectl startssl
Apache/1.3.23 mod_ssl/2.8.6 (Pass Phrase Dialog)
Some of your private key files are encrypted for security reasons.
In order to read them you have to provide us with the pass phrases.
Server your.server.dom:443 (RSA)
Enter pass phrase:
```

Az RSA titkos kulcs titkosítása nagyon fontos. Ha valaki megkaparintja a "titkosítatlan RSA titkos kulcsot", akkor könnyen eltulajdoníthatja a webszervert. Ha a kulcs titkosított, az illető nem tud semmit tenni a jelmondat nélkül, hacsak "nyers erővel" (brute force) fel nem töri. Használj erős (értsd: hosszú és értelmetlen) jelmondatot erre a célra.

A kulcs titkosítása néha kellemetlenség forrása is lehet, mivel a webszerver minden indításakor kéri a jelmondatot. Különösen ha rc szkripteket használunk, a webszerver rendszerindításkor történő betöltéséhez. A jelmondat bekérése problémát okozhat, mivel megállítja a folyamatot, bemenetre vár.

Könnyen megszabadulhatsz a jelmondattól, ha visszafejted (decrypt) a kulcsot. Bizonyosodj meg arról, hogy senki se szerezheti meg a kulcsot. Vedd figyelembe a biztonsági és védelmi ajánlásokat, mielőtt visszafejted a kulcsot a webszerveren.

A kulcs visszafejtésének módja:

Először készíts másolatot a titkosított kulcsról

```
# cp server.key server.key.cryp
```

aztán írd újra a kulcsot titkosítással. Kérni fogja tőled az eredeti titkosított kulcs jelmondatát:

```
# /usr/local/ssl/bin/openssl rsa -in server.key.cryp -out server.key
read RSA key
Enter PEM pass phrase:
writing RSA key
```

Íme egy módja annak, miként biztosíthatod a visszafejtett titkos kulcsot. Így csak a root felhasználó olvashatja:

```
# chmod 400 server.key
```

6.7. SSL teljesítménybeállítás

6.7.1. Munkafolyamatok közötti SSL részfolyamat-gyorstár (Inter Process SSL Session Cache)

Az Apache többfolyamatos modellt használ, amelyben NEM ugyanaz a munkafolyamat foglalkozik az összes kéréssel. Ennek eredményeként az SSL részfolyamat adatai (Session Information) elvesznek, mikor a kliens többszörös kéréssel fordul a szerverhez. A többszörös kapcsolódás nagy többletterhelést jelent a webszervernek és a kliensnek. Ennek elkerülésére az SSL részfolyamatok adatai egy munkafolyamatok közötti részfolyamat-tárban tárolódnak, ez lehetővé teszi a munkafolyamatok számára a kapcsolódási adatokhoz való hozzáférést. Az SSLSessionCache kapcsoló az `/usr/local/apache2/conf/ssl.conf` fájlban van, itt határozhatod meg az SSL részfolyamat-gyorstár helyét:

```
SSLSessionCache        shmht:logs/ssl_scache(512000)
#SSLSessionCache        shmcb:logs/ssl_scache(512000)
#SSLSessionCache         dbm:logs/ssl_scache
SSLSessionCacheTimeout  300
```

A dbm használata: a logs/ssl_scache DBF hash-fájlt készít gyorstárként a helyi lemezeden.

A shmht használata: a logs/ssl_scache(512000) a gyorstárat a megosztott memóriában hozza létre.

> **shmht vs shmcb:** shmht: egy hash táblát használ az SSL kapcsolódási adatok gyorstárazására a megosztott memóriában.
>
> shmht: egy ciklikus buffert használ az SSL kapcsolódási adatok gyorstárazására a megosztott memóriában.

> **Megjegyzés::** Nem minden platform/operációs rendszer támogatja hash tábla létrehozását a megosztott memóriában. Ekkor a "dbm:logs/ssl_scache"-t kell használnod helyette.

6.7.2. Az SSLSession gyorstár ellenőrzése

Az SSLSessionCache megfelelő működésének ellenőrzésére az **openssl** segédprogramot használhatod a **-reconnect** kapcsolóval, mint azt a következőkben láthatod:

```
# openssl s_client -connect your.server.dom:443 -state  -reconnect

CONNECTED(00000003)
.......
.......
Reused, TLSv1/SSLv3, Cipher is EDH-RSA-DES-CBC3-SHA
SSL-Session:
.....
Reused, TLSv1/SSLv3, Cipher is EDH-RSA-DES-CBC3-SHA
SSL-Session:
.....
Reused, TLSv1/SSLv3, Cipher is EDH-RSA-DES-CBC3-SHA
SSL-Session:
.....
Reused, TLSv1/SSLv3, Cipher is EDH-RSA-DES-CBC3-SHA
SSL-Session:
.....
Reused, TLSv1/SSLv3, Cipher is EDH-RSA-DES-CBC3-SHA
SSL-Session:
.....
```

A **-reconnect** kapcsoló kényszeríti az "s_client"-et arra, hogy ötször ugyanazzal a SSL munkafolyamat-azonosítóval (SSL session ID) kapcsolódjon a szerverhez. Ötször ugyanannak az SSL munkafolyamat-azonosítónak az újrahasználatát kell látnod, mint a fenti példában.

A. HTTP/HTTPS teljesítménybeállító programok

Íme egy lista, a Web szerverekhez használható, nyílt forrású teljesítménybeállító programokról:

i. SSLswamp (http://distcache.sourceforge.net/) - egy SSL-t használó szerverhez csatlakozni képes terhelhetőségi teszt/teljesítménymérő program

ii. HTTPERF (http://www.hpl.hp.com/personal/David_Mosberger/httperf.html) - Egy eszköz a Web szerver teljesítményének beméréséhez

iii. ab (http://httpd.apache.org/docs-2.1/en/programs/ab.html) - Apache HTTP szerver teljesítménybeállító program

B. Hardveres SSL titkosítási megoldások

A következő hardveres SSL titkosítási megoldások érhetők el:

i. CHIL (Cryptographic Hardware Interface Library; titkosító hardverek csatolófelületeinek programkönyvtára) (http://www.ncipher.com) az nCipher-től

ii. ab (http://httpd.apache.org/docs-2.1/en/programs/ab.html) - Apache HTTP szerver teljesítménybeállító program

C. Megbízott tanúsítvány hatóságok (Trusted Certificate Authorities)

Íme a tanúsítvány hatóságok (Certificate Authorities) listája, amelyeket a különböző böngészők megbízhatónak minősítenek:

i. Baltimore (http://www.baltimore.com/)

ii. Entrust (http://www.entrust.com/)

iii. GeoTrust (http://www.globalsign.net/)

iv. Thawte (http://www.thawte.com)

v. TrustCenter (http://www.trustcenter.de/)

A nyilvános kulcsú titkosítással kapcsolatos szavak gyűjteménye

A

Asymmetric Cryptography (aszimmetrikus titkosítás)

Ez a titkosítás egy kulcspárt - titkos (Private) és nyilvános (Public) kulcsot használ. A titkos kulcsot (Private Key) biztonságos helyen kell tartani, a nyilvános kulcsot (Public Key) pedig széles körben terjeszteni.

C

Certificate (tanúsítvány)

Egy adatjegyzék, amely az X.509 Format-ban szereplő információkat tartalmazza.

Certificate Authority (CA) (tanúsítvány hatóság; TH)

A digitális tanúsítvány (Digital Certificate) kibocsátója. Azon végfelhasználó (End-Entity) azonosságát is hitelesíti, amelynek birtokában van a digitális tanúsítvány.

Certificate Signing Request (CSR) (tanúsítvány aláírási kérelem; TAK)

A tanúsítvány aláírási kérelem (Certificate Signing Request; CSR) az, ami elküldésre kerül a tanúsítvány hatóságnak (Certifiate Authority; CA) bejegyzésre. Az aláírási kérelem tartalmazza a végfelhasználó (End-Entity) nyílvános kulcsát (Public Key), amelyre a digitális tanúsítványt kérelmezik.

Common Name (CN) (közönséges név)

A közönséges név (Common Name) a végfelhasználó (End-Entity) neve, például Saqib Ali. Ha a végfelhasználó egy webszerver, akkor ez a webszerver "teljesen képzett domain neve" (Fully Qualified Domain Name; FQDN).

D

Digital Certificate (digitális tanúsítvány)

Egy tanúsítvány kapcsolja a nyilvános kulcsot (Public Key) egy személyhez (Subject; end-entity). Ez a tanúsítvány az X.509 Format-ban meghatározott egyéb azonosító információkat is tartalmaz a személyről. A kibocsátó CA (tanúsítvány hatóság; TH) aláírásával van ellátva, annak titkos kulcsát használva ehhez a művelethez. Íme egy digitális tanúsítvány.

Digital Signature (digitális aláírás)

Egy digitális aláírás (Digital Signature) az üzenetkivonat (Message Digest) titkos kulccsal történő aláírásával készíthető el. Biztosít a küldő (Sender) (személy)azonosságáról (Identity) és az adat sértetlenségéről (Integrity of the Data).

E

End-Entity (végfelhasználó)

Egy felhasználó, aki részt vesz a nyilvános kulcsú titkosításban. Általában szerver, szolgáltatás (Service), útválasztó (Router) vagy személy. A tanúsítvány hatóság (Certificate Authority; CA) nem végfelhasználó.

H

Hash (a titkosítás eredménye)

A titkosítás eredménye egy hexadecimális szám, amely egy szöveg karakterláncából lett generálva, ezért két különböző karakterlánc nem képes ugyanazt a titkosított eredményt produkálni.

HMAC: Keyed Hashing for Message Authentication (kulcsos titkosítás üzenethitelesítéshez)

A HMAC egy megvalósítása az üzenethitelesítő-kód algoritmusnak (Message Authentication Code Algorithm).

M

Message Authentication Code (üzenethitelesítő-kód)

Hasonló az üzenetkivonathoz (Message Digest; Hash/Fingerprint), azzal a különbséggel, hogy a titkosított eredmény kiszámításához a megosztott rejtett kulcs (Shared Secret Key) lett felhasználva. Mivel a rejtett kulcs lett felhasználva, ezért egy támadó nem tudja megváltoztatni az üzenetkivonatot. Mindezek mellett a rejtett kulcsok kell először közölni a partnerekkel, ellentétben a digitális aláírással, amelynél az üzenetkivonat a titkos kulccsal (Private Key) van aláírva. A HMAC egy példája az üzenethitelesítő-kód algoritmusnak.

Message Digest 5 - MD5 (üzenetkivonat 5)

Az üzenetkivonat 5 (Message Digest 5; MD5) egy 128 bites egyirányú titkosító függvény.

P

Private Key (titkos kulcs)

A titkos kulcsot - az aszimmetrikus titkosítási rendszerben - a tulajdonosa (End-Entity) biztonságos helyen tartja. Titkosításra és visszafejtésre használható.

Public Key (nyilvános kulcs)

Az aszimmetrikus titkosítás nyilvános kulcsát széles körben terjesztik. Titkosításra és visszafejtésre használható.

Public Key Infrastructure (PKI) (nyilvános kulcsos rendszer; NYKR)

Nyilvános kulcsos rendszer.

S

SHA-1: Secure Hash Algorithm (biztonságos titkosító algoritmus)

A biztonságos titkosító algoritmus (Secure Hash Algorithm; SHA-1) egy 160 bites egyirányú titkosító függvény. Az üzenet maximális hossza 2^64 bit.

Secure Socket Layer (SSL) (Biztonsági Alréteg)

Az SSL egy biztonsági protokoll, amely hitelességet (digitális aláírás), megbízhatóságot (titkosítás) és adatsértetlenséget (üzenet ellenőrzés - MD5, SHA, stb.) biztosít

Symmetric Cryptography (Szimmetrikus Titkosítás)

Ebben a rendszerben az üzenet ugyanazzal a kulccsal titkosítható és visszafejthető. (((n^2-n))/2) számú kulcsra van szüksége n felhasználónak, ha ezen módszer szerint akarnak titkosítani.