key data. The PKA encrypt callable service encrypts a supplied clear key under an RSA public key. Using the PKA decrypt callable service makes it possible to unwrap the RSA-encrypted SSL seed and return it to the application in the clear. The application can then use this clear key to generate session encryption keys.

11.6.3 Encryption exercises

Implementing Crypto-card support with Apache and OpenSSL

Now we need to create a new directory to hold the Apache, OpenSSL and Mod_SSL source trees and untar the sources. Everything for this excercise is found under the ../labstuff/crypto directory.

```
# cd /usr/src/
# mkdir webserver
# cd webserver/
# tar xfzBp /mnt/labstuff/crypto/apache_1.3.26.tar.gz
# tar xfzBp /mnt/labstuff/crypto/mod_ssl-2.8.10-1.3.26.tar.gz
# tar xfzBp /mnt/labstuff/crypto/openssl-engine-0.9.6b.tar.gz
# ls
.  ..  apache_1.3.26  mod_ssl-2.8.10-1.3.26  openssl-engine-0.9.6b
```

Change to the Openssl [engine] directory and configure, compile, test and install openssl. In order to do this there are a couple of prerequisites. You will need to apply two patches to the openssl source and change a couple of Makefiles to add a library to the compile. The patches are found in the ../labstuff/crypto directory.

```
# ls /mnt/labstuff/crypto/*.patch
/mnt/labstuff/crypto/Configure-390.patch  /mnt/labstuff/crypto/ibmca.patch
Now let's apply the patches and configure, compile, test and install openssl.
# cd openssl-engine-0.9.6b/
# patch -sp0 < /mnt/labstuff/crypto/Configure-390.patch
# cd crypto/
# patch -sp5 < /mnt/labstuff/crypto/ibmca.patch
missing header for context diff at line 4 of patch
missing header for context diff at line 28 of patch
missing header for context diff at line 52 of patch
missing header for context diff at line 70 of patch
missing header for context diff at line 105 of patch
missing header for context diff at line 122 of patch
missing header for context diff at line 138 of patch
missing header for context diff at line 839 of patch
# cd ../
# ./config -ldl
```

You should see at the end of the config command a message that says:

```
Configured for linux-s390.
```

Openssl is configure now for linux-s390. Let's **make** (compile), **make test** and **make install**:

```
# make
# make test
# make install
```

One last thing for openssl, we require a symbolic link to satisfy the libica rpm that will be install later:

```
 # ln -s /usr/lib/libcrypto.so.0.9.6  /usr/lib/libcrypto.so.2
```

Now it is time to configure mod_ssl for apache.

```
# cd ../mod_ssl-2.8.10-1.3.26/
```


The following statement is a single line:

```
# ./configure --with-apache=../apache_1.3.26 --with-ssl=../openssl-engine-0.9.6b
--enable-module=ssl --enable-module=so --enable-rule=SSL_EXPERIMENTAL
```

We are done with openssl and mod_ssl. Lets go to Apache and compile, create certificates and install Apache using ssl.

```
# cd ../apache_1.3.26/
# make
# make certificate TYPE=custom
```

Accept all the defaults and make sure that you remember the password phrase, you will need it. Answer the questions and be consistent. After you create the two certificates it is time to do a make install so that we move everything to the correct location where it will be accessible to the Linux system.

```
# make install
```

Lets look back and see what have we done:

1. Downloaded tar files for apache, modssl and openssl.
2. Created the source trees for each of the products by unravelling the tar files
3. Configured, compiled, tested (in the case of openssl) and installed the three products.
4. Created two certificates, one for the client and one for the server.

We have setup the stage for using the crypto card by implementing the software prerequisites as stated in the Device Drivers and Installation Commands. We are missing several steps to complete our setup. First, we need to install the Generalized Interface library for the IBM eServer Cryptographic Accelerator Device Driver and the openCryptoki PKCS#11 api libraries in support of all IBM crypto cards.

Note, that the Generalized Interface library (libica) is a low level api for the Specified adapter, it is not intended to be an interface which is written to by applications. Applications should use the openCryptoki PKCS#11 api for interfacing to the token.

Second, we need to install the z90crypt module and define a device. Third, we need to configure Apache to define the crypto device .

Step 1

```
# rpm -ivh /mnt/labstuff/crypto/libica-1.1-3.s390.rpm --nodeps
libica                  ##################################################
# rpm -ivh /mnt/labstuff/crypto/openCryptoki-1.4-0.s390.rpm --nodeps
openCryptoki            ##################################################
```

Step 2

There are a couple of scripts in the Device Drivers and Installation Commands manual to load and unload the z90crypt module. These scripts also take care of creating the device nodes. They are found in pages 217-219. For our purpose we will do the commands manually instead of typing the commands into a script file and running the script.

```
# insmod z90crypt
Using /lib/modules/2.4.17-SuSE/kernel/drivers/s390/misc/z90crypt.o
```

If everything went well you should be able to enter the following two commands and see a similar output.

```
# cat /proc/driver/z90crypt

z90crypt version: 1.1.1
Cryptographic domain: 7       <======== This may be different
Total device count: 1
```

```
PCICA count: 1
PCICC count: 0
requestq count: 0
pendingq count: 0
Total open handles: 0    <======== Once you bring up Apache with SSL, this will be 1


Mask of online devices: 01 means PCICA, 02 means PCICC
    0000000000000000 0001000000000000 0000000000000000 0000000000000000
    0000000000000000 0000000000000000 0000000000000000 0000000000000000


Mask of waiting work element counts
    0000000000000000 0000000000000000 0000000000000000 0000000000000000
    0000000000000000 0000000000000000 0000000000000000 0000000000000000
# cat /proc/devices
Character devices:
  1 mem
  2 pty/m%d
  3 pty/s%d
  4 ttyS
  5 ptmx
 10 misc
 43 ctc/ctctty%d
128 ptm
136 pts/%d
162 raw
227 tty3270
228 fs3270
254 z90crypt

Block devices:
  1 ramdisk
  7 loop
 94 dasd
# mknod /dev/z90crypt c 254 0     <===== 254 is the major node where z90rypt was registered
# ls /dev/z90crypt -l
crw-rw-rw-    1 root     root     254,   0 Aug 28 16:19 /dev/z90crypt
```

Step 3

Edit the apache configuration file and add the configuration for the SSLCryptoDevice following mod_ssl.c definition as shown below:

```
# cd /usr/local/apache/conf/
# vi httpd.conf
   <IfModule mod_ssl.c>
   SSLCryptoDevice ibmca
   ...
   something else
   ...
   </IfModule>
```

Lets test!!!

```
# cd ../bin/
# ./apachectl startssl
Apache/1.3.26 mod_ssl/2.8.10 (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 pbc99212.pok.ibm.com:443 (RSA)
Enter pass phrase:

Ok: Pass Phrase Dialog successful.
./apachectl startssl: httpd started
```

If everything is working you should see several processes running in Linux that look like:

```
/usr/local/apache/bin/httpd -DSSL
```

Now use the browser from your PC and point to the IP address of your host (9.117.99.24<**x**>) and test. If you use **https** you will be ask to accept the certificate comming from the server, please answer only for this session.

11.7 Accounting and logging

11.7.1 Accounting

Accounting was developed for the purpose of keeping track of resource compsumption. Although the original intention was not security, the account tools can help audit or investigate intrussions to your system. In order to be able to use the accounting tools your Linux kernel must be configure for BSD acounting: CONFIG_BSD_PROCESS_ACCT=y . This is found under the Genral Setup label if you are using make menuconfig.

In general the accounting commands are divided into two:

11.7.1.1 Connection accounting

Connection accountng tracks user sessions and user logins. There are two facilities in Linux which helps gathering the records: utmp and wtmp.

UTMP is used for active user sessions. The utmp file allows one to discover information about who is currently using the system. There may be more users currently using the system, because not all programs use utmp logging.

WTMP file records all logins and logouts. Its format is exactly like utmp except that a null user name indicates a logout on the associated terminal. Furthermore, the terminal name "~" with user name "shutdown" or "reboot" indicates a system shutdown or reboot and the pair of terminal names "|"/"}" logs the old/new system time when date(1) changes it. wtmp is maintained by login(1), and init(1) and some versions of getty(1). Neither of these programs creates the file, so if it is removed record-keeping is turned off.

Command	Description
dump-utmp	Converts the raw data from /var/run/utmp or /var/run/wtmp into parsable format to custom scripts
ac	Prints out a report of connect time (in hours) based on the logins/logouts in the current wtmp file. A total is also printed out.
last	Last searches back through the file /var/log/wtmp (or the file designated by the -f flag) and displays a list of all users logged in (and out) since that file was created.