Procedures to build crypto libraries in Minix

Note: this docment is fully tested only in Minix3.1.2a.

In this document, we give step-by-step instructions on how to create a crypto library (*libcrypt.a*), and compile/link/build/run applications using the newly-built library in Minix.

Step 1: Get the files needed:

1. Download the *libcrypt.tar* file to your host machine from http://www.cis.syr.edu/~wedu/seed/Labs/Files/libcrypt.tar
2. Upload the *libcrypt.tar* file to your Minix machine, and put it in the directory of */usr/tmp*. You can use *ftp* to upload the *libcrypt.tar* file.
3. There is an alternative way to get the file into Minix: Use 'packman' to install a tool called 'wget'in Minix, then use wget to download file by issuing 'wget http://www.cis.syr.edu/~wedu/seed/Labs/Files/libcrypt.tar'
4. Login to your Minix machine, and do the following:

   ```
   # cd /usr/tmp
   # tar xvf libcrypt.tar
   ```

   Now, in this directory (*/usr/tmp*), there should be two directories: *libcrypt*, and *demo*, and one file: *README*

   In the default directory:

README:	explanation of the contents of this package

   In *libcrypt/* directory:

md5.h:	header file for the md5 algorithm
md5.c:	function implementation of the md5 algorithm
aes.h:	header file for the aes algorithm
aes.c:	function implementation of aes algorithm
sha256.h:	header file for the sha256 algorithm
sha256.c:	function implementation of sha256 algorithm
hmac_md5.c:	function implementation of hmac_md5 algorithm
Makefile:	the makefile used to build the library (this file is useless in Minx3.1.2a)

   In *demo/* directory:

hmc_md5_demo.c:	the program to demonstrate the usage of hmac_md5
aes_demo.c:	the program to demonstrate to use of aes algorithm

In the following steps, we assume our current directory is */usr/tmp*.

Step 2: Create the crypto library of our own:

We name the crypto library that we will create as libcrypt.a. Follow the procedures below:

1. Copy the header files to the */usr/src/include* directory, and install the header files to */usr/include/* using the following command:
   ```
   # cp /usr/tmp/libcrypt/*.h /usr/src/include/ && cd /usr/src/include && make install
   ```
2. Create a sub-directory under */usr/src/lib* called *crypt:*
   ```
   # mkdir /usr/src/lib/crypt
   ```
3. Copy the function implementation files, to */usr/src/lib/crypt* directory:
   ```
   # cp /usr/tmp/libcrypt/*.c /usr/src/lib/crypt
   ```
4. We need to create a file named *Makefile.in* in the directory of */usr/src/lib/crypt.* The content of the file is shown below:
   ```
   # Makefile.in for lib/crypt.

   CFLAGS="-O -D_MINIX -D_POSIX_SOURCE"

   LIBRARIES=libcrypt

   libcrypt_FILES=" \
           aes.c \
           hmac_md5.c \
           md5.c \
           sha256.c"

   TYPE=both
   ```

   Also, we need to modify the *Makefile.in* in the directory of */usr/src/lib* as below:
   Find `SUBDIRS="ansi \` , and insert a line as an entry: `crypt \` . So it look like this:
   ```
   SUBDIRS="ansi \
   … //omit severl entries
   ip \
   crypt \
   math \
   other \
   … //omit severl entries
   gnu"
   ```

   Save and exit.
5. Make sure there is no *Makefile* in */usr/src/lib/crypt*. Then build and install the library (*libcrypt.a*) using the following commands:
   ```
   #cd /usr/src/lib
   # make
   # make install
   ```
   // After this step, you should be able to find *libcrypt.a* in */usr/lib*

Step 3: Compile and link the demo programs

1. Compile the *aes_demo.c* and *hmc_md5_demo.c* programs

```
# cd /usr/tmp/demo
# cc aes_demo.c –o aes_demo –l crypt
# cc hmc_md5_demo.c –o hmc_md5_demo –l crypt
```

2. Run the aes_demo and hmac_md5_demo program:

```
# ./aes_demo
# ./hmc_md5_demo
```