AC 2007-2819: DEVELOPING BASIC CRYPTOGRAPHY LAB MODULES WITH OPEN SSL

Ed Crowley, University of Houston

Developing Basic Cryptography Lab Modules with OpenSSL

Perceived topic: Innovative Teaching

Keywords: Security, Cryptography, Open Source, Lab Development, Networking

While there has been a recent proliferation of quality cryptography texts, there remains a shortage of quality applied laboratory exercises and related support materials. In part, this is due to the cost and availability of commercial cryptographic software. In part, this is due to the time and resource commitment required to develop laboratory modules.

Utilization of free open source software can help offset related monetary costs. While there are many open source cryptographic tools, not all tools are appropriate for "hands-on" learning activities. For example, our lab modules are designed to make cryptographic processes visible to students; while, tools like PGP and GPG are designed to make cryptographic processes transparent to users.

This paper presents the author's experiences developing and utilizing applied cryptographic lab modules. The primary cryptographic software utilized, in these modules, is OpenSSL. This is an Open Source tool kit available for both Linux and Windows. In addition to OpenSSL, the modules utilize a variety of other open source tools including Ethereal (WireShark), net cat, Firefox, and Apache.

Because the objectives are curriculum dependent, it would be helpful to review the curriculum within which our cryptography course evolved.

Our Applied Cryptography Class

Our College is a small college within a large university. In addition to other goals, our applied security program is designed to prepare students to provide enterprise security assessment and evaluation. Expected job titles for our graduates include security investigator, manager, and auditor.

The scope of the applied cryptography class includes cryptographic services required to securely store and transmit confidential information. It also includes related cryptographic services such as those that provide integrity, authentication, and nonrepudiation. Specific topic areas include: symmetric and asymmetric cryptography, digital fingerprints, message authentication codes, digital signatures, certificates, public key infrastructure, and virtual private networks (VPNs). Laboratory activities that enable students to apply related cryptographic principles augment conventional class activities.

In a classical cryptography course, learning goals would include the mathematical analysis of related cryptographic number theory. While we introduce our students to the basic mathematical foundations of specific algorithms, an in-depth examination of number theory wouldn't necessarily contribute to our program's applied goals.

Rather than researching new cryptographic algorithms, our students focus on evaluating cryptographic implementations in the context of particular enterprise security goal(s). In these evaluations, our students draw on existing standards and standard organizations such as the Security Requirements for Cryptographic Modules (FIPS-140) [2] and the National Institute for Standards and Technology (NIST). A look at sample learning goals for our laboratory module's will help illustrate this.

Sample Module Goals
At the end of this class, students will be able to:

1. Apply cryptographic services to:
   a. Conceal information within a file (encryption).
   b. Verify a file's integrity
   c. Authenticate a file's origin
2. Provide evidence of non-repudiation by creating and utilizing an RSA digital signature
3. Employ asymmetric technology to encrypt a symmetric key
4. Utilize symmetric, asymmetric, and hybrid cryptographic technologies.
5. Define, explain, and demonstrate relevant cryptographic services including confidentiality, integrity, authentication, and non-repudiation.
6. Demonstrate and explain relevant cryptographic mechanisms including encryption, hashing, message digests, message authentication codes (MACs), and digital signatures.
7. Explain how symmetric and asymmetric technologies combine in a hybrid cryptosystem.
8. Create an encrypted communications channel that provides confidentiality.
9. Compare and contrast symmetric and asymmetric cryptography
10. List and discuss cryptographic vulnerabilities including key management, randomness, and speed
11. Generate symmetric keys and asymmetric key pairs
12. Create an unsecured communications channel between two computers.
13. Communicate securely through an unsecured communications channel.

When presented to the students, these module goals are accompanied by an overview. Let's look at the overview.

Cryptography Overview
As cryptography evolved, it's definition also evolved. Originally, cryptography's definition was derived from its literal meaning, that is, from the original Greek, as "secret writing".

Over time, as cryptography continued to evolve, the definition broadened. For example, , the National Institute for Standard and Technology (NIST) now defines cryptography as "… a branch of mathematics that is based on the transformation of data and can be used to provide several security services: confidentiality, data integrity, authentication, authorization and non-repudiation."

In various forms, cryptography has been employed for over 4,000 years. Until the mid 1970s, cryptography's primary security service was that of confidentiality i.e. protecting secrets. During that time, the predominant cryptographic technology was symmetric or single key, cryptology. Historically, key management problems associated with symmetric cryptography, limited cryptographic applications primarily to large, well funded, governmental organizations.

However the, relatively, recent discovery of asymmetric cryptography has dramatically impacted the way that cryptography can be applied. Specifically, asymmetric key cryptography eliminates the key management problem. Further, asymmetric cryptography is not limited to providing confidentiality services.

In addition to confidentiality, asymmetric technology can provide integrity, authentication, and non repudiation security services. Asymmetric cryptography has also lowered the costs associated with enterprise level cryptographic implementations. The argument could be made that without asymmetric cryptography, ecommerce would not be feasible.

Because of these advantages, asymmetric cryptography is now widely employed. However, for bulk encryption asymmetric technology is much slower than symmetric technology. Consequently, asymmetric cryptography is most often utilized in a hybrid system that combines the strengths of both symmetric and asymmetric cryptographic technologies.

For example, in a hybrid system symmetric cryptography is used for bulk encryption while asymmetric cryptography is used for key exchange and temporary session key encryption. Understanding both asymmetric and symmetric technologies as well as how they combine into a hybrid system can provide a foundation for a variety of security technologies and protocols including Secure Session Layer (SSL), Public Key Infrastructure (PKI), and Virtual Private Networks (VPNs). As Figure 1-1 shows, symmetric, asymmetric, and one way functions provide a foundation from which learning modules that include Basic Protocol Building Blocks, Applied Protocols, and Secure Protocol Applications can evolve [1].

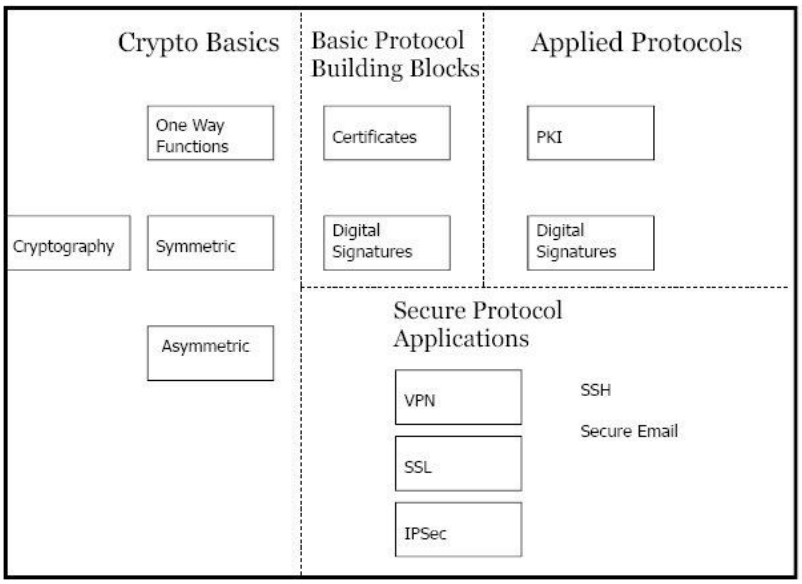

Figure 1-1 Crypto Basics

Our lab modules provide students with "Hands-on" experience with cryptographic services. In these modules, students apply symmetric, asymmetric, and hybrid technologies as well as one way functions. Lab activities focus on four cryptographic security services and their related mechanisms. Table 1-1 shows these services and mechanism.

Service	Mechanism	Definition
confidentiality	encryption/ decryption	The property that sensitive information is not disclosed to unauthorized individuals, entities, or processes. (FIPS 140-2)
integrity	Hash/Digest MAC Digital Signatures	The property that sensitive data has not been modified or deleted in an unauthorized and undetected manner. (FIPS 140-2)
authenticity	Hash MAC Digital Signature Certificates PKI	An assurance of authenticity is provided using authentication controls, which protect a communication system against acceptance of a fraudulent transmission or simulation by establishing the validity of the information content and the originator. (NIST 800-21)
non-repudiation	MAC Digital Signature Certificates PKI	Non-repudiation services provide assurance of the origin of data to both the receiver and a third party. The objective is to provide evidence to counter denials that the sender participated in a specified transaction. (NIST 800-21)

Table 1-1 Security services and mechanisms.

Specific security services are obtained through the application of specific cryptographic mechanisms. These mechanisms may employ symmetric or asymmetric technologies. They may also combine into a hybrid system. As the modules progress, students gain

experience with specific security services and mechanisms. Students employ these technologies and mechanisms to provide confidentiality, integrity, authenticity, and non-repudiation security services.

Lab Modules
Related laboratory activities were developed in modules. Each module has its own objectives and procedures. Figure 1-1 presents a context of how additional modules, after Crypto Basics, might evolve.

In addition to extensibility, modules facilitate use of the activities in multiple courses. Since each module demonstrates specific learning objectives, they can be mixed and matched to separate course level objectives. Since each module is complete, they may be also be presented in different course without concern for undocumented dependencies. Though, they are dependent upon the skill level of the student within each class. Along with these attributes, module learning objectives are designed to be easily measured.

The next section presents several modules that we have developed and successfully utilized.

Module Examples
Part Zero – Introduction

Objectives
At the end of this module, you will be able to:
- Access and use OpenSSL from the command line.
- Access OpenSSL help (man) pages from the command line.
- Locate and access online sources of OpenSSL documentation.
- Download related cryptographic standards from the National Institute for Standards and Technology (NIST).

Procedure
This module introduces you to the OpenSSL Cryptographic Toolkit. Here, you will access relevant help (man) pages and learn to work with OpenSSL from the command line. You will also learn to identify relevant Internet based information sources. At the end of the lab, you will download, and read, the "Security Requirements for Cryptographic Modules" standard from the National Institute for Standards and Technology (NIST) [2].

First, open a console window.

At the prompt, type:
```
man openssl
```

After you have read the general man pages exit from them. You may exit from the Man Pages by either pressing the q key or pressing both the ctrl and the z keys.

In later activities, you will be using the enc command. Now, is a good time to become familiar with this command. To learn more about the encrypt command type:

```
man enc
```

Now, lets look a little closer at OpenSSL. While you have the manual pages showing in one console window, open another console window. At the prompt in the new console window, type the following commands: (Note: be sure to press return at the end of each command.)

```
openssl  version

openssl list-standard-commands

openssl list-message-digest-commands

openssl list-cipher-commands

openssl ciphers -v  -ssl3
```

Open a navigator window and browse to the OpenSSL web site. In the browser's URL text box, type:

```
www.OpenSSL.org
```

Check the available documentation. Also check for other sources of OpenSSL information such as forums and mailing lists.

Now, browse to the National Institute of Standards and Technology at:

```
http://csrc.nist.gov/publications/fips/
```

Download and save FIPS 140-2, the "Requirements for Cryptographic Modules". If you don't have permanent storage available at your workstation, use a web based service to email the pdf file to yourself.

Before proceeding to Part One, you should check the end of lab questions to make sure that you can answer them. You will also want to save relevant screen shots for your online lab narrative.

Part One –Symmetric Cryptography

Objectives At the end of this module, you will be able to: • Use OpenSSL to generate a pseudo random number

- Generate DES keys.
- Utilize DES to encrypt and decrypt documents.
- Download and encrypt an RFC.
- Use the cat command to type a file to the console.
- Create file hashes that demonstrate file integrity.

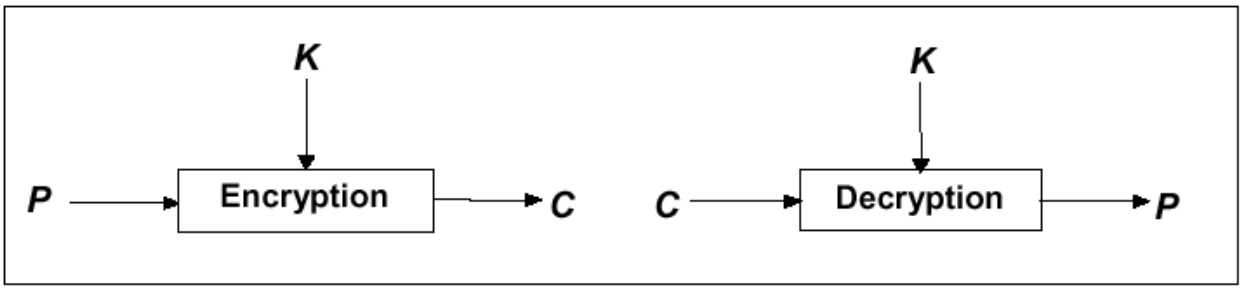

Figure P 1-1 Encryption and Decryption

During this step, you will generate a 56 bit DES key. Then, you will use that DES key to encrypt and decrypt an ASCII text file. After that, you will employ an MD5 hash to demonstrate that the unencrypted file is identical to the original text file.

For this lab module, we will use the IETF's RFC3766 as our text file. To obtain the file, launch your browser and go to the rfc editor site at the below listed URL. Once you are at the web site, locate rfc3766.

ftp://ftp.rfc-editor.org/in-notes/

To download this document, right click on the rfc3766.txtlink.

Select the "Save link as" option.

Download the file as RFC3766XX.txt where XX are your initials.

Note One
Accept the default download directory. This will be your home directory. Since you are user Knoppix, your home directory is also named "Knoppix".
Note Two
Be sure to change the download file name to rfc3766XX where XX are your initials.
Note Three
Accept the default file type (rfc3766XX.txt). Note that while Linux does not require file types, we will use them here to distinguish among several versions of this file.

Now, open a console window. Automatically, you will be placed in your default home directory. Since, by default, you are user Knoppix, this will be the Knoppix directory. When you list the files in your home directory, you should see the RFC that you previously downloaded.

```
ls –al
```

While you are in your home directory, create a file containing a pseudo random number 56 bits long. Later, you will use that number as your DES key. In an open console window, enter:

```
openssl rand –out des_keyXX 56
cat des_keyXX
```

Notes

1. In the above command lines, XX represents your initials.
2. Make sure that your initials are different than your partners.
3. As Linux is case sensitive, the case you choose for your initials is significant.

Now, encrypt RFC3766.

```
openssl des –e –a –kfile des_keyXX –in rfc3766XX.txt –
out rfc3766XX.enc
```

(Note, enter the above command listing on a single line.)

To make sure that the encryption operation worked, list the encrypted file.

```
cat rfc3766XX.enc
```

Now to provide assurance that the process works both ways, you should decrypt the .enc file that you just created. After you decript the file, go ahead and list it on your console.

```
openssl des –d –a –kfile des_keyXX –in rfc3766XX.enc –out
rfc3766XX.dec
cat rfc3766XX.dec
```

You would expect the file rfc3766XX.dec to be identical to the file that you downloaded rfc3766XX.txt. Since identical files will have identical message digests, you can prove that the files are identical by creating and comparing each file's message digest. Enter the following command lines. Then, compare the generated digests (hashes).

```
openssl md5 rfc3766XX.txt
openssl md5 rfc3766XX.dec
```

Part One A-- Symmetric Key and File Exchange, Symmetric Decryption

Objectives

At the end of this module, you will be able to:

- Use Apache to distribute an encrypted document.
- Use the Net Cat Utility to distribute your DES Key.
- Use the Ethereal (WireShark) Packet Analyzer to capture packets.
- Demonstrate file integrity through the creation of file hashes.

For this lab module, you will exchange encrypted files and DES keys with your lab partner. Your encrypted file will be distributed through port 80 on the web. Your DES key will be distributed through Net Cat using whatever port your select.

After the exchange, you will have two different encrypted files, yours and your partners. You will also have two different DES keys, yours and your partners. Consequently, you will need to be very careful to keep the names of the files and keys organized. In order to succeed in this module, you will also need to have good communications with your lab partner.

To ascertain the security of each communications channel, you will employ the Ethereal (WireShark) packet analyzer. With this tool, you will capture each TCP stream. After you have captured the appropriate packets, you will emulate an intruder and attempt to reassemble the original files from the captured packets.

Prior to beginning the lab, you must exchange IP addresses with your lab partner. You can determine your IP address by opening a console window and typing:

```
ifconfig
```

Since you will be using your web server to exchange the encrypted file with your lab partner, you now need to move your encrypted file from your home directory to your web server's root directory.

To do this first, from the KDE menu, select KNOPPIX, from there, select root shell. This will open a shell with root privileges. Once that shell is open, copy the encrypted file from your home directory to your web server's default directory.

```
cp rfc3766XX.enc /var/www/rfc3766XX.enc
```

Now, you are ready to start your web server.

```
su
apache start
```

To make sure that your web server is running, start Netscape Navigator. Place 127.0.0.1 in the browser's address box and press the Return Key. At this point, you should see the default Apache homepage in your Navigator Window.

You will now need to edit the web server's default home page. For this operation, use the Kwrite editor. You will need superuser privileges to edit the default home page. Open a console window, type:

```
su
kwrite
```

… use the File Open Menu selection to open the “index.html” file. You will find this file in the /var/www directory.

Now, in the open kwrite session, edit the index.html. First, delete everything in the file after the first <body> tag and prior to the list of links (<ul> tag> at the page bottom. Then, edit the first link in the list to point to the encrypted file.

```
<A HREF="rfc3766XX.enc">RFC3766</A>
```

After you have deleted the unneeded portions of the original “index.html” file and added the link to your encrypted file, save the index file. Use the default file name. Use the default directory. (Note, if you are using super user privileges and still have problems saving the index.html file simply save the file under a different name

You should further customize the default home page. At a minimum, you should add your name in a level 1 header at the top of the page. This will provide you lab partner with assurance when they go to your page to download your encrypted file. You should also add other relevant information.

```
<h1>Uno Kitty</h1>
```

Now, have your partner navigate to your web server with their browser. They can do that by entering your IP address in the address box of their web browser. Once there, they should download the rfc that you encrypted to their machine.

Now, you will use NetCat to create a connection between your and your lab partner’s computers. This connection will facilitate your key exchange.

During this exchange, the partner that is furnishing the key will be the server. The partner that is receiving the key will be the client. Prior to starting this step, begin capturing packets with Ethereal. You will find it useful to employ a display or a capture filter that limits the display or capture to your partner’s IP address. You will also find it useful to have Ethereal update the list of packets in real time.

Open a console window, and type:

```
su
ethereal
```

After the Ethereal Interface opens, first choose “Captures” pull down menu. Then, choose the Options choice. Select the appropriate check box to select the “Display packets in real time” option. On a busy network, you may also want to set the display filter to only display packets from your lab partner.

Now, each lab partner needs to set up their NetCat Server. Part of the set up is the selection of an upper level port with which to open communications. You will need to select a unique port and to communicate that port number with your lab partner. It is

important to communicate the appropriate port number to your lab partner. Before proceeding, please coordinate with your lab partner and fill out the following table.

	IP Address	Server Port	Client Port
You			
Your Partner			

Table 1-2 NetCat Planning Table

Open your NetCat server connection by typing:

```
nc –vvn –l –p XXXX < des_keyXX
```

XXXX is the number of the port that you will be opening for your server. Be sure to choose a higher level (over 1024) port. And be sure to let your lab partner know which port you have chosen.

To establish your client connection to your lab partner type:

```
nc aaa.bbb.ccc.ddd YYYY > des_keyPP
```

Where aaa.bbb.ccc.ddd is the ip address of your partner's server and YYYY are the upper level ports that your lab partner has opened on their NetCat server.

You should now exchange keys with your partner. That is, your partner should send their key to you and you should send your key to your partner. If you have chosen Ethereal's "View packets in real time" option, you should see the TCP connection in Ethereal's capture packet window. After the key exchange is complete, press Ctrl-C to break the net cat connection.

At this point, you should stop your packet capture and save the file as part 1A. This packet capture will contain the passwords. At the end of the lab, you should see if you can recover the key from the packets. Hint, you may have to choose binary file type to save the key. What does this experience tell you about the security of conventional network connections?

Now, make sure that you have captured your partner's key. Note, if you see a file with the appropriate name but a file size of zero, then your key exchange failed and you need to repeat the process.

```
ls –al
```

Now, that you have evidence that the file transfer was successful, use your lab partner's key to decrypt the encrypted rfc that you downloaded from your lab partner's web site.

```
openssl des –d –a –kfile des_keyPP –in rfc3766PP.enc –out
rfc3766PP.dec
```

Now, let's create evidence that will prove that the decrypted rfc is identical to the original unencrypted rfc. That is, you will create a hash of booth files. Compare it to the original file's hash.

```
openssl md5 rfc3766XX.txt
openssl md5 rfc3766PP.dec
```

Conclusion

To understand cryptographic basics, a student needs to have a first hand knowledge of basic cryptographic activities and services. These activities include generating random numbers as well as generating and applying symmetric keys and asymmetric key pairs. Basic services include confidentiality and integrity. Services covered by other modules include authentication and nonrepudiation.

Antidotal student response to the lab modules has been enthusiastic. The fact that the labs are open source and LiveCD based means that the students can repeat and verify their lab work at their home or at their work. It also means that the students can freely distribute the software utilized within the modules. An unintended side effect is that other professors have adopted existing lab modules into other classes.

There is however much further work to be done. One task would be to create a custom LiveCd that would contain lab modules and associated class materials. Another task would be to create more modules that would extend the lab's scope. At present, we have lab modules for all of the Basic Topics in Figure 1-1. We also have modules that contain a subset of the Protocol Building Blocks.

In our class, the students also create their own projects. Several students have extended the class activities to demonstrate other software such as TrueCrypt. As the class evolves, our anticipation is that it the utilization of Open Source Software will contribute to the class becoming more project orientated.

Bibliographic Information

1. Dark, M, Morales, L, Justice, C, A Methodology for Developing and Disseminating Curriculum Resource material in Information Security, CISSE 2005.

2. FIPS PUB 140-2: Security Requirements for Cryptographic Modules, National Institute of Standards and Technology, May 25 2001

3. Kar, D, Teaching Cryptography in an Applied Computing Program, Journal of Computing Sciences in Colleges, 2006.

4. Mel, H, Baker, D, Cryptography, Decrypted, Addison-Wesley, 2001.

5. OpenSSL, http://www.openssl.org, last visited on Jan 14, 2007.

6. Stieber, A, OpenSSL Hacks, Linux Journal, July, 2006.

7. Tjaden, B, Fundamentals of Secure Computer Systems, Franklin, Beedle & Associates, Inc., 2004.

8. Viega,J,Messier,M, Chandra, P,Network Security with OpenSSL Cryptography for Secure Communications, O'Reilly, 2002.