OpenSSL Background and Hints

29 September 2009

1 Introduction

OpenSSL's manpages are very detailed, but the high-level documentation is somewhat lacking. This handout aims to fill in some of the gaps. It also provides some crucial details about using C libraries that you may be missing because earlier courses use C++ exclusively.

2 A word about manpages

Unix manpages (with which you should be familiar from EECS 280) are generally referred to by giving the name of the page followed by the section of the manual. For example, "see `ls(1)`" refers the reader to the manpage for ls in Section 1, which is general user commands. The section number is important because sometimes multiple sections contain manpages with the same name (e.g., `rsa(1)` and `rsa(3)`.

You do not need to memorize all the sections, but it is helpful to know that Section 2 documents system calls and Section 3 documents the C standard library. Some libraries, such as OpenSSL, define their own subsections of Section 3, such as 3ssl.

You should also be aware that manpages don't necessarily have a "categorical" name. Instead, they are usually titled after some subset of the functions they document. For example, the manpage about allocating and freeing OpenSSL bignums has the titles `BN_new(3ssl)`, `BN_init(3ssl)`, `BN_free(3ssl)`, and so forth.

3 Coping with C

This section describes some of the features of OpenSSL that may seem strange to people coming from a 280/281 background.

3.1 Using C libraries from C++

The OpenSSL library is intended to be used from C, but we are using it from C++. The biggest difference that you might not be aware of is that none of the OpenSSL structures have constructors. Instead, most of them have initialization functions that you need to call before using them. Similarly, they may have cleanup functions you must call when you are done using them. Make sure you at least skim the entire manpage for the structures you are working with to be sure you're not missing any of these! Do not try to skip the initialization by simply allocating OpenSSL structures with `new`; it won't work.

3.2 C file I/O

Some of the OpenSSL functions you need to use in Part 3 take pointers to `FILE` objects. The C standard library's file I/O functions all return `FILE *`s. They are well documented in the manpages; `http://www.cppreference.com/wiki/c/io/start` gives an overview. To get you started, `fopen(3)` is the function you should use to open files.

4 The EVP interface

The OpenSSL EVP (EVP is from Digital EnVeloPe) functions provide a high-level interface to many of the cryptographic functions. In general, the OpenSSL documentation recommends that the EVP interface (and the high-level interfaces in general) should be preferred to low-level interfaces that are specific to the cryptographic function in use. Conveniently, the initialization functions for encryption and hashing accept objects that specify the underlying algorithm to use. For example, there is a `EVP_aes_128_ofb` function that can be passed to `EVP_EncryptInit_ex` to tell the `EVP_CIPHER_CTX` being initialized to use AES-128 in OFB mode.

One "gotcha" is that the manpages may not always be up to date on the latest algorithms included. In particular, `EVP_DigestInit(3ssl)` lists `EVP_sha` and `EVP_sha1`, but not `EVP_sha256`. However, if you examine OpenSSL's evp.h (in /usr/include/openssl/evp.h on CAEN machines), you'll find that `EVP_sha256` does exist. Quickly understanding code that has no or poor documentation will be an important skill if you go on to do security research, so it might be useful to practice making educated guesses about what might be available in OpenSSL and checking them against its headers.

`hmac(3ssl)` is probably the simplest OpenSSL interface you need to use for this project, so if you're having trouble dealing with OpenSSL, be sure to start with HMAC.

5 Bignums

"bignums" are unlimited-precision integers. In other words, unlike the built-in types, there is no limit to the sizes of numbers that bignums can represent. You will need to use bignums to represent the Diffie-Hellmann parameters in part 2 of project 1. OpenSSL's bignum functions all start with `BN_` and are outlined in `bn(3ssl)`. The bignum library is a reasonable example of how encapsulation is implemented in C, which does not have classes like C++. It may be to your advantage to familiarize yourself with this library. Check the "see also" section at the bottom of `bn(3)` for more details on specific functions.

5.1 Choosing Random Bignums

You can use `BN_rand(3ssl)` to choose random bignums. You can safely ignore the warning in the manpage to seed the PRNG before calling this function on Linux because `/dev/urandom` (a device file providing pseudorandom data) is automatically used to provide the seed.