FREE eBook

LEARNING openssl

Free unaffiliated eBook created from
Stack Overflow contributors.

#openssl

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: openssl

It is an unofficial and free openssl ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official openssl.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with openssl

Remarks

This section provides an overview of what openssl is, and why a developer might want to use it.

It should also mention any large subjects within openssl, and link out to the related topics. Since the Documentation for openssl is new, you may need to create initial versions of those related topics.

Versions

Release	Date
1.1.0e	2017-02-16
1.1.0d	2017-01-26
1.1.0c	2016-11-10
1.1.0b	2016-09-26
1.1.0a	2016-09-22
1.1.0	2016-08-25
1.0.2k	2017-01-26
1.0.2j	2016-09-26
1.0.2i	2016-09-22
1.0.2h	2016-05-03
1.0.2g	2016-03-01

Examples

Installation or Setup

OpenSSL is an open source project that provides a robust, commercial-grade, and full-featured toolkit for the Transport Layer Security (TLS) and Secure Sockets Layer (SSL) protocols. It is also a general-purpose cryptography library.

The OpenSSL toolkit is licensed under an Apache-style license, which basically means that you are free to get and use it for commercial and non-commercial purposes subject to some simple

license conditions.

Build and Install openssl on Linux/Unix Systems

Overview

These instructions are for acquiring, building, and installing openssl from source. Openssl is usually included in package managers as well.

Resources

https://github.com/openssl/openssl

Dependencies

- make
- perl 5
- gcc/clang
- git

Dependencies can be installed through a package manager such as apt, dnf, or brew.

Steps

```
$ cd ~/path/to/projects
$ git clone https://github.com/openssl/openssl.git
$ cd openssl
$ ./config
$ make
$ make test
$ sudo make install
```

By default, openssl will be installed to /usr/local.

Verify

```
$ openssl --version
```

You now have a default build of openssl installed to your machine.

(De-)Initialization of openssl library

Overview

Openssl consists of 2 libraries: `libcrypto` and `libssl`. Before openssl API can be used in an

application, mandatory initialization procedures are expected to be performed. Once application is done with openssl related work, it is expected to cleanup allocated resources.

Code below does complete initialization, however, developer is free to initialize only openssl stuff he is interested in.

Initialize libcrypto

```
ERR_load_crypto_strings();
OpenSSL_add_all_algorithms();
OPENSSL_config(NULL); // Load default configuration (e.g. openssl.conf)
```

Initialize libssl

```
OPENSSL_init_ssl(0, NULL);
```

Deinitialize

```
CONF_modules_unload(1);
EVP_cleanup();
CRYPTO_cleanup_all_ex_data();
ERR_remove_state();
ERR_free_strings();
```

Run OpenSSL on Windows without Installing

This workaround helped us so much at my job (Tech Support), we made a simple batch file we could run from anywhere (We didnt have the permissions to install the actual exe). This workaround will run OpenSSL and open up the bin folder for you (cause this is where any files you create or modify will be saved).

How to Set Up:

1. Download the OpenSSL binaries [here][1]. (Note that this is confirmed to work with version 0.9.8h.)

2. Copy this code to a file named StartOpenSSL.bat. Save this to a location of your choice. It can be run from anywhere.

```
@echo off
title OpenSSL

cd\openssl\bin

if exist "C:\openssl\share\openssl.cnf" (

set OPENSSL_CONF=c:/openssl/share/openssl.cnf
```

```
start explorer.exe c:\openssl\bin

echo Welcome to OpenSSL

openssl

) else (

echo Error: openssl.cnf was not found
echo File openssl.cnf needs to be present in c:\openssl\share
pause

)

exit
```

3. Once you have downloaded the OpenSSL binaries, extract them to your C drive in a folder titled OpenSSL. (The path needs to be C:\OpenSSL). Do not move any of the folders contents around, just extract them to the folder.

4. You are ready to use OpenSSL. This is a great workaround for Windows users who dont have the privileges to install it as it requires no permissions. Just run the bat file from earlier by double clicking it. [1]: http://gnuwin32.sourceforge.net/packages/openssl.htm

OpenSSL commands examples

Inspect ssl certificate

```
openssl x509 -in server.crt -noout -text
```

Generate server key

```
openssl genrsa -out server.key 2048
```

Generate csr

```
openssl req -out server.csr -key server.key -new
```

Read Getting started with openssl online: https://riptutorial.com/openssl/topic/2695/getting-started-with-openssl

Chapter 2: Keys

Syntax

- EVP_PKEY *EVP_PKEY_new(void);
- RSA * RSA_new(void);
- int RSA_generate_key_ex(RSA *rsa, int bits, BIGNUM *e, BN_GENCB *cb);
- int EVP_PKEY_assign_RSA(EVP_PKEY *pkey, RSA *key);
- int PEM_write_PrivateKey(FILE *fp, EVP_PKEY *x, const EVP_CIPHER *enc, unsigned char *kstr, int klen, pem_password_cb *cb, void *u);
- int PEM_write_bio_PrivateKey(BIO *bp, EVP_PKEY *x, const EVP_CIPHER *enc, unsigned char *kstr, int klen, pem_password_cb *cb, void *u);
- EVP_PKEY *PEM_read_PrivateKey(FILE *fp, EVP_PKEY **x, pem_password_cb *cb, void *u);
- EVP_PKEY *PEM_read_bio_PrivateKey(BIO *bp, EVP_PKEY **x, pem_password_cb *cb, void *u);
- void EVP_PKEY_free(EVP_PKEY *key);

Examples

Generate RSA Key

In order to generate an RSA key, an `EVP_PKEY` must first be allocated with `EVP_PKEY_new`:

```
EVP_PKEY *pkey;
pkey = EVP_PKEY_new();
```

An exponent for the key is also needed, which will require allocating a `BIGNUM` with `BN_new` and then assigning with `BN_set_word`:

```
BIGNUM *bn;
bn = BN_new();
BN_set_word(bn, RSA_F4);
```

To generate the key, create a new `RSA` with `RSA_new` and call `RSA_generate_key_ex`:

```
RSA *rsa;
rsa = RSA_new();
RSA_generate_key_ex(
    rsa,  /* pointer to the RSA structure */
    2048, /* number of bits for the key - 2048 is a good value */
    bn,   /* exponent allocated earlier */
    NULL, /* callback - can be NULL if progress isn't needed */
);
```

To assign the newly generated key to the `EVP_PKEY` structure, call `EVP_PKEY_assign_RSA`:

```
EVP_PKEY_assign_RSA(pkey, rsa);
```

The `RSA` structure will be automatically freed when the `EVP_PKEY` structure is freed. This is done with `EVP_PKEY_free`:

```
EVP_PKEY_free(pkey);
```

Save Private Key

An `EVP_PKEY` can be saved directly to disk in a several formats. `PEM_write_PrivateKey` is used to save `EVP_PKEY` in a PEM format:

```
FILE *f;
f = fopen("key.pem", "wb");
PEM_write_PrivateKey(
    f,                  /* use the FILE* that was opened */
    pkey,               /* EVP_PKEY structure */
    EVP_des_ede3_cbc(), /* default cipher for encrypting the key on disk */
    "replace_me",       /* passphrase required for decrypting the key on disk */
    10,                 /* length of the passphrase string */
    NULL,               /* callback for requesting a password */
    NULL                /* data to pass to the callback */
);
```

To save a private key to a `BIO`, use `PEM_write_bio_PrivateKey`:

```
BIO *bio;
bio = BIO_new(BIO_s_mem());
PEM_write_bio_PrivateKey(
    bio,                /* BIO to write the private key to */
    pkey,               /* EVP_PKEY structure */
    EVP_des_ede3_cbc(), /* default cipher for encrypting the key on disk */
    "replace_me",       /* passphrase required for decrypting the key on disk */
    10,                 /* length of the passphrase string */
    NULL,               /* callback for requesting a password */
    NULL                /* data to pass to the callback */
);
```

Load Private Key

To load a private key directly from disk, use the `PEM_read_PrivateKey` function:

```
FILE *f;
EVP_PKEY *pkey;
f = fopen("key.pem", "rb");
PEM_read_PrivateKey(
    f,     /* use the FILE* that was opened */
    &pkey, /* pointer to EVP_PKEY structure */
    NULL,  /* password callback - can be NULL */
    NULL   /* parameter passed to callback or password if callback is NULL */
);
```

To load a private key from a `BIO`, use `PEM_read_bio_PrivateKey`:

```
BIO *bio;
bio = BIO_new_mem_buf((void *)input, input_len);
PEM_read_bio_PrivateKey(
    bio,   /* BIO to read the private key from */
    &pkey, /* pointer to EVP_PKEY structure */
    NULL,  /* password callback - can be NULL */
    NULL   /* parameter passed to callback or password if callback is NULL */
);
```

Read Keys online: https://riptutorial.com/openssl/topic/4760/keys

Credits

S. No	Chapters	Contributors
1	Getting started with openssl	Camille G., Community, Josip Ivic, Michael, Rex Linder, Roland Bär, tysonite, user, vaibhav magar
2	Keys	Nathan Osman, tysonite