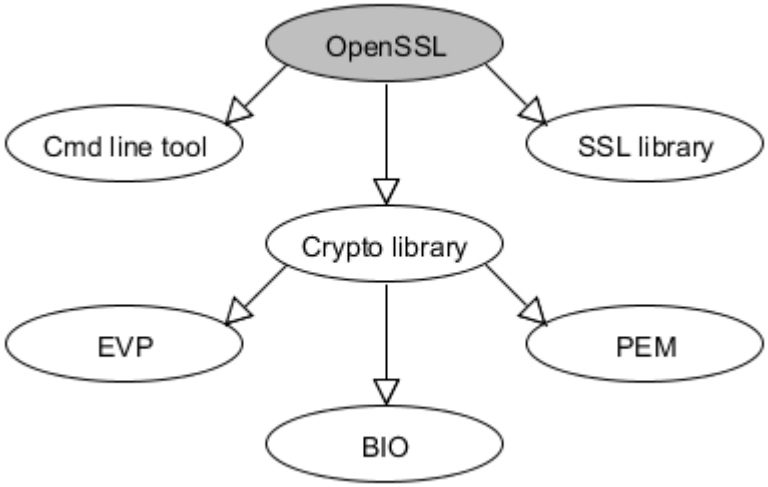
5. OpenSSL: Seznámení s knihovnou, hašovací funkce, proudové šifry

Obsah

Co je OpenSSL

OpenSSL (http://www.openssl.org) je knihovna pracující s SSL (Secure Socket Layer) a TLS (Transport Layer Security) protokoly. Tato knihovna je vyvíjena od roku 1998 a je založena na starší knihovně SSLeay. OpenSSL se skládá ze tří hlavních částí:



Command line tool

OpenSSL obsahuje nástroj pro příkazový řádek, pomocí kterého lze rychle a jednoduše šifrovat text (DES, 3DES, RC4 …), vytvářet certifikáty (X.509), počítat hashe (SHA1, SHA256, MD5 …) a další funkce z Crypto knihovny. Přiklad na vypočítání SHA1 pro zadaný text v příkazovém řádku:

```
echo -n "Hashovany text." | openssl sha1
```

SSL library

SSL library je knihovna implementující vrstvy síťové komunikace Secure Sockets Layer (SSL v2/v3) a Transport Layer Security (TLS v1). Zastřešuje použití hashovacích a šifrovacích funkcí pro navázání zabezpečené relace a implementuje také několik typů struktur, se kterými ssl funkce pracují, jako například SSL_CTX, SSL_METHOD a další.

Crypto library

Crypto knihovna poskytuje implementaci konkrétních kryptografických funkcí, datových struktur pro uložení klíčů a formátování zpráv a práci s certifikáty. Kromě nízkoúrovňové implementace jednotlivých algoritmů také poskytuje rozhraní vyšší úrovně, tzv. EVP (envelope) funkce.

Zadání úkolu

Task2_hash (3 body):

Napište program, který nalezne libovolnou zprávu, jejíž **hash (SHA-384) začíná** zleva na posloupnost nulových bitů:

- Počet nulových bitů je zadán celým číslem jako parametrem na příkazové řádce.
- Pořadí bitů je big-endian: Bajt 0 od MSB do LSB, Bajt 1 od MSB do LSB, …, poslední bajt od MSB do LSB.
- Nalezenou **zprávu vypište v hexadecimální podobě**.

Ukázky kódu

Příklad na hashovací funkce

Tento kód zahashuje `text` hashem uloženým v proměnné `hashFunction`.

```
#include <stdio.h>
#include <string.h>
#include <openssl/evp.h>

int main (int argc, char * argv[]) {
    char text[] = "Text pro hash.";
```

```
    char hashFunction[] = "sha1";  // zvolena hashovaci funkce ("sha1", "md5", ...)

    EVP_MD_CTX * ctx;  // struktura kontextu
    const EVP_MD * type; // typ pouzite hashovaci funkce
    unsigned char hash[EVP_MAX_MD_SIZE]; // char pole pro hash - 64 bytu (max pro sha 512)
    unsigned int length;  // vysledna delka hashe

    /* Inicializace OpenSSL hash funkci */
    OpenSSL_add_all_digests();
    /* Zjisteni, jaka hashovaci funkce ma byt pouzita */
    type = EVP_get_digestbyname(hashFunction);

    /* Pokud predchozi prirazeni vratilo -1, tak nebyla zadana spravne hashovaci funkce */
    if (!type) {
        printf("Hash %s neexistuje.\n", hashFunction);
        return 1;
    }

    ctx = EVP_MD_CTX_new(); // create context for hashing
    if (ctx == NULL)
        return 2;

    /* Hash the text */
    if (!EVP_DigestInit_ex(ctx, type, NULL)) // context setup for our hash type
        return 3;

    if (!EVP_DigestUpdate(ctx, text, strlen(text))) // feed the message in
        return 4;

    if (!EVP_DigestFinal_ex(ctx, hash, &length)) // get the hash
        return 5;

    EVP_MD_CTX_free(ctx); // destroy the context

    /* Vypsani vysledneho hashe */
    printf("Hash textu \"%s\" je: ", text);
    for (unsigned int i = 0; i < length; i++)
        printf("%02x", hash[i]);
    printf("\n");
    return 0;
}
```

Zdrojový kód zkompilujete pomocí následujícího příkazu. Doporučujeme napsat si makefile.

```
gcc -Wall -pedantic nazevSouboru.c -o nazevVystupu -lcrypto
```

Příklad na proudovou šifru

Tento kód zašifruje pomocí proudové šifry `rc4` otevřený text `ot` za použití klíče `key` a inicializačního vektoru `iv`.

```
#include <stdlib.h>
#include <string.h>
#include <openssl/evp.h>

int main (void) {
    unsigned char ot[1024] = "Text pro rc4.";  // open text
    unsigned char st[1024];  // sifrovany text
    unsigned char key[EVP_MAX_KEY_LENGTH] = "Muj klic";  // klic pro sifrovani
    unsigned char iv[EVP_MAX_IV_LENGTH] = "inicial. vektor";  // inicializacni vektor
    const char cipherName[] = "RC4";
    const EVP_CIPHER * cipher;

    OpenSSL_add_all_ciphers();
    /* sifry i hashe by se nahraly pomoci OpenSSL_add_all_algorithms() */
    cipher = EVP_get_cipherbyname(cipherName);
    if (!cipher) {
        printf("Sifra %s neexistuje.\n", cipherName);
        return 1;
    }

    int otLength = strlen((const char *) ot), stLength = 0, tmpLength = 0;

    EVP_CIPHER_CTX * ctx; // context structure
    ctx = EVP_CIPHER_CTX_new();
    if (ctx == NULL)
        return 2;

    printf("OT: %s\n", ot);

    /* Sifrovani */
    if (!EVP_EncryptInit_ex(ctx, cipher, NULL, key, iv)) // context init - set cipher, key, init vector
        return 3;

    if (!EVP_EncryptUpdate(ctx,  st, &tmpLength, ot, otLength)) // encryption of pt
        return 4;
```

```
    stLength += tmpLength;

    if (!EVP_EncryptFinal_ex(ctx, st + stLength, &tmpLength)) // get the remaining ct
        return 5;

    stLength += tmpLength;

    printf ("Zasifrovano %d znaku.\n", stLength);

    /* Desifrovani */
    if (!EVP_DecryptInit_ex(ctx, cipher, NULL, key, iv)) // nastaveni kontextu pro desifrovani
        return 6;

    if (!EVP_DecryptUpdate(ctx, ot, &tmpLength,  st, stLength)) // desifrovani st
        return 7;

    otLength += tmpLength;

    if (!EVP_DecryptFinal_ex(ctx, ot + otLength, &tmpLength)) // dokonceni (ziskani zbytku z kontextu)
        return 8;

    otLength += tmpLength;

    /* Clean up */
    EVP_CIPHER_CTX_free(ctx);

    /* Vypsani zasifrovaneho a rozsifrovaneho textu. */
    printf("ST: %s\nDT: %s\n", st, ot);
    return 0;
}
```

Zdrojový kód zkompilujete pomocí následujícího příkazu. Avšak doporučujeme napsat si makefile.

```
gcc -Wall -pedantic nazevSouboru.c -o nazevVystupu -lcrypto
```

Tipy pro vytvoření makefile

Viz například návod z MIT (http://web.mit.edu/gnu/doc/html/make_2.html), nebo si zopakujte informace z BI-PA2 (https://courses.fit.cvut.cz/BI-PA2/tutorials/texts/makefile.html#_kompilace-modul%C5%AF-a-makefile).

Všechny příkazy jsou odsazeny znakem TAB.

```
CC = gcc
CFLAGS = -g
LDFLAGS = -lcrypto

all: hash
run:
	#run your tests here...
hash: hash.o
	$(CC) -o $@ $< $(LDFLAGS)
%.o: %.c
	$(CC) -c -o $@ $< $(CFLAGS)
```

Tipy pro použití CLion (pokud používáte pro vývoj)

V CLion otevřete `CMakeLists.txt`, přidejte následující řádek, a pak znovu načtěte CMake projekt:

```
target_link_libraries(${PROJECT_NAME} crypto)
```

Zdroje

- Oficiální stránky OpenSSL: http://www.openssl.org/
- Secure programming with the OpenSSL API: http://ibm.co/ZYsHLO
- John Viega, Matt Messier, Pravir Chandra: Network Security with OpenSSL, Odkaz (https://books.google.cz/books?id=IIqwAy4qEl0C)
- OpenSSL Wiki http://wiki.openssl.org/index.php/Main_Page