Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Кафедра информационных технологий автоматизированных систем

Отчет по лабораторной работе №6
по курсу «ИТОКБ»
на тему: «Знакомство с OpenSSL»

Вариант 4

Выполнил магистрант группы 025941: Колесников В.Г.

Проверил: Боброва Т.С.

МИНСК
2021

Задание 1: Установить OpenSSL, ознакомиться с возможностями библиотеки.

Результат выполнения команды «help» приведен на рисунке 1.1.

```
OpenSSL> help
Standard commands
asn1parse         ca                ciphers           cms
crl               crl2pkcs7         dgst              dhparam
dsa               dsaparam          ec                ecparam
enc               engine            errstr            gendsa
genpkey           genrsa            help              list
nseq              ocsp              passwd            pkcs12
pkcs7             pkcs8             pkey              pkeyparam
pkeyutl           prime             rand              rehash
req               rsa               rsautl            s_client
s_server          s_time            sess_id           smime
speed             spkac             srp               storeutl
ts                verify            version           x509

Message Digest commands (see the `dgst' command for more details)
blake2b512        blake2s256        gost              md4
md5               rmd160            sha1              sha224
sha256            sha3-224          sha3-256          sha3-384
sha3-512          sha384            sha512            sha512-224
sha512-256        shake128          shake256          sm3

Cipher commands (see the `enc' command for more details)
aes-128-cbc       aes-128-ecb       aes-192-cbc       aes-192-ecb
aes-256-cbc       aes-256-ecb       aria-128-cbc      aria-128-cfb
aria-128-cfb1     aria-128-cfb8     aria-128-ctr      aria-128-ecb
aria-128-ofb      aria-192-cbc      aria-192-cfb      aria-192-cfb1
aria-192-cfb8     aria-192-ctr      aria-192-ecb      aria-192-ofb
aria-256-cbc      aria-256-cfb      aria-256-cfb1     aria-256-cfb8
aria-256-ctr      aria-256-ecb      aria-256-ofb      base64
bf                bf-cbc            bf-cfb            bf-ecb
bf-ofb            camellia-128-cbc  camellia-128-ecb  camellia-192-cbc
camellia-192-ecb  camellia-256-cbc  camellia-256-ecb  cast
cast-cbc          cast5-cbc         cast5-cfb         cast5-ecb
cast5-ofb         des               des-cbc           des-cfb
des-ecb           des-ede           des-ede-cbc       des-ede-cfb
des-ede-ofb       des-ede3          des-ede3-cbc      des-ede3-cfb
des-ede3-ofb      des-ofb           des3              desx
rc2               rc2-40-cbc        rc2-64-cbc        rc2-cbc
rc2-cfb           rc2-ecb           rc2-ofb           rc4
rc4-40            seed              seed-cbc          seed-cfb
seed-ecb          seed-ofb          sm4-cbc           sm4-cfb
sm4-ctr           sm4-ecb           sm4-ofb
```

Рисунок 1.1 — Результат команды «help»

Задание 2: Выполнить тестирование скорости выполнения различных алгоритмов шифрования.

Скорость выполнения алгоритмов шифрования можно проверить с помощью команды «speed». Результаты выполнения алгоритмов приведены на рисунках 2.1-2.4.

```
The 'numbers' are in 1000s of bytes per second processed.
type             16 bytes     64 bytes    256 bytes   1024 bytes   8192 bytes  16384 bytes
md2                  0.00         0.00         0.00         0.00         0.00         0.00
mdc2                 0.00         0.00         0.00         0.00         0.00         0.00
md4              60599.69k   185710.44k   437996.03k   713902.66k  1200832.13k  1189237.30k
md5             103145.67k   198886.38k   352809.18k   432833.50k   463289.11k   645824.51k
hmac(md5)        38266.74k   110294.72k   258983.34k   417901.91k   715199.96k   724915.54k
sha1            269480.47k   660711.79k  1250172.91k  1657138.86k  1824000.68k  1934600.87k
rmd160           45762.76k   114658.05k   138980.52k   174673.58k   189086.25k   190257.83k
rc4             302385.61k   328395.05k   308277.25k   332092.07k   446617.43k   440358.23k
des cbc          77074.76k    79864.77k    80821.33k    80409.34k    67174.40k    53024.09k
des ede3         19851.02k    21748.14k    19224.22k    19188.05k    19644.36k    19496.96k
idea cbc             0.00         0.00         0.00         0.00         0.00         0.00
seed cbc         95262.62k    91940.65k    64485.38k    65053.45k    65122.29k    65190.79k
rc2 cbc          33354.29k    34041.86k    34555.48k    34587.99k    33650.22k    34723.16k
rc5-32/12 cbc        0.00         0.00         0.00         0.00         0.00         0.00
blowfish cbc     86143.82k    96183.80k   135918.11k   141485.63k   142587.38k   146909.87k
cast cbc        117669.97k   134583.94k   135505.66k   138953.05k   133474.80k    87342.72k
aes-128 cbc     168493.37k   215821.50k   177767.17k   158843.29k   142800.21k   140446.98k
aes-192 cbc     126035.97k   124352.75k   130507.26k   130233.28k   131625.44k   133207.38k
aes-256 cbc     113767.89k   118127.60k   116222.96k   137360.52k   178295.18k   172530.16k
camellia-128 cbc   159867.89k   139666.16k   133517.40k   137666.35k   141650.37k   203134.20k
camellia-192 cbc   131104.38k   150417.88k   156411.96k   158993.29k   160023.46k   158639.86k
camellia-256 cbc   129906.72k    97093.48k   101207.67k   102852.95k   110439.08k   157122.56k
sha256          223543.25k   552090.71k  1026689.80k  1038815.57k  1169823.08k  1166344.19k
sha512           33089.57k   133234.33k   242077.53k   357742.11k   415440.90k   421462.02k
whirlpool        24680.35k    57579.39k   129282.65k   157142.02k   165303.64k   168711.51k
aes-128 ige     206549.59k   221994.55k   191526.06k   156981.93k   145636.17k   146173.77k
aes-192 ige     114932.20k   125623.23k   117940.06k   129497.99k   135009.09k   132055.04k
aes-256 ige     100110.43k   108571.95k   118155.61k   118852.27k   132416.26k   108544.00k
ghash           899486.19k  2402045.70k  5164476.50k  7679098.54k  7950011.05k  7184624.30k
rand              8477.37k    31291.57k   105333.00k   247956.62k   406320.44k   439797.31k
```

Рисунок 2.1 — Скорость выполнения алгоритмов

```
                  sign    verify    sign/s verify/s
rsa  512 bits 0.000050s 0.000003s  19923.9 346547.4
rsa 1024 bits 0.000117s 0.000010s   8561.5  98631.1
rsa 2048 bits 0.000955s 0.000031s   1047.4  31905.0
rsa 3072 bits 0.003770s 0.000077s    265.3  13000.1
rsa 4096 bits 0.008406s 0.000131s    119.0   7635.1
rsa 7680 bits 0.075682s 0.000400s     13.2   2499.1
rsa 15360 bits 0.263158s 0.001644s      3.8    608.2
                  sign    verify    sign/s verify/s
dsa  512 bits 0.000098s 0.000065s  10239.9  15441.6
dsa 1024 bits 0.000183s 0.000152s   5470.8   6580.2
dsa 2048 bits 0.000428s 0.000294s   2337.7   3397.1
```

Рисунок 2.2 — Скорость выполнения RSA и DSA

```
                              sign    verify    sign/s verify/s
160 bits ecdsa (secp160r1)   0.0002s   0.0002s   4532.2   4444.6
192 bits ecdsa (nistp192)    0.0004s   0.0003s   2475.1   3103.5
224 bits ecdsa (nistp224)    0.0001s   0.0002s  14895.8   5138.0
256 bits ecdsa (nistp256)    0.0000s   0.0001s  30802.7   9373.3
384 bits ecdsa (nistp384)    0.0017s   0.0012s    600.3    829.5
521 bits ecdsa (nistp521)    0.0005s   0.0007s   2015.8   1405.7
163 bits ecdsa (nistk163)    0.0004s   0.0007s   2810.5   1512.4
233 bits ecdsa (nistk233)    0.0005s   0.0010s   2116.6    998.9
283 bits ecdsa (nistk283)    0.0007s   0.0013s   1477.9    747.4
409 bits ecdsa (nistk409)    0.0015s   0.0027s    683.1    365.1
571 bits ecdsa (nistk571)    0.0019s   0.0041s    516.0    244.5
163 bits ecdsa (nistb163)    0.0004s   0.0008s   2770.8   1306.7
233 bits ecdsa (nistb233)    0.0005s   0.0010s   1936.4   1002.6
283 bits ecdsa (nistb283)    0.0007s   0.0011s   1445.3    901.0
409 bits ecdsa (nistb409)    0.0010s   0.0020s   1002.0    492.4
571 bits ecdsa (nistb571)    0.0021s   0.0042s    466.5    240.9
256 bits ecdsa (brainpoolP256r1)   0.0005s   0.0006s   1939.6   1687.7
256 bits ecdsa (brainpoolP256t1)   0.0007s   0.0006s   1510.3   1795.9
384 bits ecdsa (brainpoolP384r1)   0.0011s   0.0008s    891.1   1184.4
384 bits ecdsa (brainpoolP384t1)   0.0014s   0.0009s    721.2   1058.9
512 bits ecdsa (brainpoolP512r1)   0.0017s   0.0013s    596.3    773.3
512 bits ecdsa (brainpoolP512t1)   0.0017s   0.0017s    588.5    573.6
```

Рисунок 2.3 — Скорость выполнения ECDSA

```
                               op      op/s
160 bits ecdh (secp160r1)   0.0003s   3222.2
192 bits ecdh (nistp192)    0.0003s   2988.0
224 bits ecdh (nistp224)    0.0001s   8597.4
256 bits ecdh (nistp256)    0.0001s  16782.5
384 bits ecdh (nistp384)    0.0010s    987.0
521 bits ecdh (nistp521)    0.0006s   1697.9
163 bits ecdh (nistk163)    0.0003s   3151.8
233 bits ecdh (nistk233)    0.0005s   2158.7
283 bits ecdh (nistk283)    0.0008s   1296.9
409 bits ecdh (nistk409)    0.0014s    726.9
571 bits ecdh (nistk571)    0.0023s    439.7
163 bits ecdh (nistb163)    0.0002s   4230.7
233 bits ecdh (nistb233)    0.0005s   2121.9
283 bits ecdh (nistb283)    0.0008s   1213.5
409 bits ecdh (nistb409)    0.0014s    736.6
571 bits ecdh (nistb571)    0.0029s    342.1
256 bits ecdh (brainpoolP256r1)   0.0006s   1590.6
256 bits ecdh (brainpoolP256t1)   0.0006s   1728.8
384 bits ecdh (brainpoolP384r1)   0.0016s    641.4
384 bits ecdh (brainpoolP384t1)   0.0011s    893.5
512 bits ecdh (brainpoolP512r1)   0.0023s    439.7
512 bits ecdh (brainpoolP512t1)   0.0023s    437.8
253 bits ecdh (X25519)   0.0000s  20241.9
448 bits ecdh (X448)   0.0006s   1622.4
                               sign    verify    sign/s verify/s
253 bits EdDSA (Ed25519)   0.0001s   0.0002s  14481.2   5451.9
456 bits EdDSA (Ed448)   0.0004s   0.0007s   2295.4   1411.7
```

Рисунок 2.4 — Скорость выполнения ECDH и EdDSA

Задание 3: Создать криптографические ключи. Выбрать несколько произвольных файлов и выполнить:

- шифрование (зашифрование и расшифрование) посредством различных симметричных алгоритмов;
- шифрование (зашифрование и расшифрование) посредством различных асимметричных алгоритмов;
- хэширование различных файлов различными алгоритмами (обязательно md5 и sha1)

Процесс и результат выполнения команд создания криптографических ключей через «genrsa» приведен на рисунке 3.1.

```
OpenSSL> genrsa -out ./rsa_private_des_1024.prem -des -rand /dev/random 1024
Generating RSA private key, 1024 bit long modulus (2 primes)
.............+++++
....+++++
e is 65537 (0x010001)
Enter pass phrase for ./rsa_private_des_1024.prem:
Verifying - Enter pass phrase for ./rsa_private_des_1024.prem:
OpenSSL> genrsa -out ./rsa_private_des3_2048.pem -des3 -rand /dev/random 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
.................+++++
.+++++
e is 65537 (0x010001)
Enter pass phrase for ./rsa_private_des3_2048.pem:
Verifying - Enter pass phrase for ./rsa_private_des3_2048.pem:
OpenSSL> genrsa -out ./rsa_private_aes_256_4096.pem -aes256 -rand /dev/random 4096
Generating RSA private key, 4096 bit long modulus (2 primes)
.........................++++
.....................................................................................++++
e is 65537 (0x010001)
Enter pass phrase for ./rsa_private_aes_256_4096.pem:
Verifying - Enter pass phrase for ./rsa_private_aes_256_4096.pem:
OpenSSL> genrsa -out ./rsa_private_camellia256_8192.pem -camellia256 -rand /dev/random 8192
Generating RSA private key, 8192 bit long modulus (2 primes)
.....................................................................................................
.....................................+++
....+++
e is 65537 (0x010001)
Enter pass phrase for ./rsa_private_camellia256_8192.pem:
Verifying - Enter pass phrase for ./rsa_private_camellia256_8192.pem:
```

Рисунок 3.1 — Процесс и результат создания криптографических ключей через «genrsa»

Для установки параметров шифрования DSA создаются файлы параметров на 1024, 2048, 4086 и 8192 бит. Процесс и результат создания файла параметров через «dsaparm» приведен на рисунке 3.2.

```
OpenSSL> dsaparam -rand /dev/random -out dsa_params_1024 1024
Generating DSA parameters, 1024 bit long prime
This could take some time
..+.....................+.....................+..+..+............+...+.....................+............
....+.........++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*
.....+.....+....+..+..........+.........................+.............+.............................
..+..+........+........+.+...........+.............+.+........+.....................+....+.............
....+.................+...........+...+..+..++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++*
```

Рисунок 3.2 — Процесс и результат создания файла параметров через «dsaparm»

Процесс и результат выполнения команд создания криптографических ключей через «gendsa» приведен на рисунке 3.3.

```
OpenSSL> gendsa -out dsa_private_des_1024.pem -des ./dsa_params_1024
Generating DSA key, 1024 bits
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
OpenSSL> gendsa -out dsa_private_des3_2048.pem -des3 ./dsa_params_2048
Generating DSA key, 2048 bits
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
OpenSSL> gendsa -out dsa_private_aes256_4096.pem -aes256 ./dsa_params_4096
Generating DSA key, 4096 bits
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
OpenSSL> gendsa -out dsa_private_camellia256_8192.pem -camellia256 ./dsa_params_8192
Generating DSA key, 8192 bits
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
```

Рисунок 3.3 — Процесс и результат создания криптографических ключей через «gendsa»

Процесс шифрования посредством различных симметричных алгоритмов представлен на рисунке 3.4. На рисунке 3.5 показаны файлы, используемые при шифровании. На рисунках 3.6 и 3.7 представлено сравнение исходных и расшифрованных файлов.

```
OpenSSL> enc -seed -pbkdf2 -in 1 -out 1_enc
enter seed-cbc encryption password:
Verifying - enter seed-cbc encryption password:
OpenSSL> enc -seed -d -pbkdf2 -in 1_enc -out 1_dec
enter seed-cbc decryption password:
OpenSSL> enc -aria256 -a -pbkdf2 -in 2 -out 2_enc
enter aria-256-cbc encryption password:
Verifying - enter aria-256-cbc encryption password:
OpenSSL> enc -aria256 -a -d -pbkdf2 -in 2_enc -out 2_dec
enter aria-256-cbc decryption password:
```

Рисунок 3.4 — Шифрование с использованием симметричных алгоритмов

Рисунок 3.5 — Файлы, используемые при шифровании симметричными алгоритмами

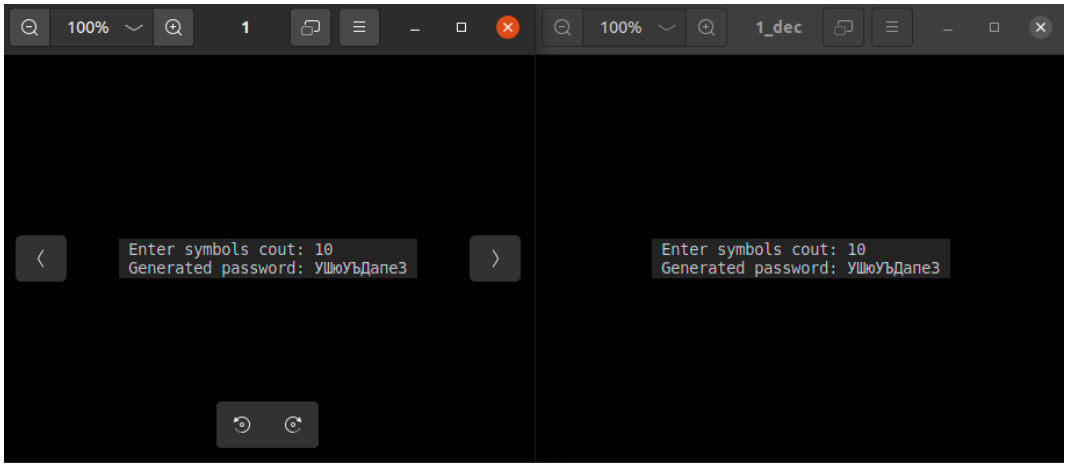

Рисунок 3.6 — Сравнение исходного и расшифрованного файла «1»

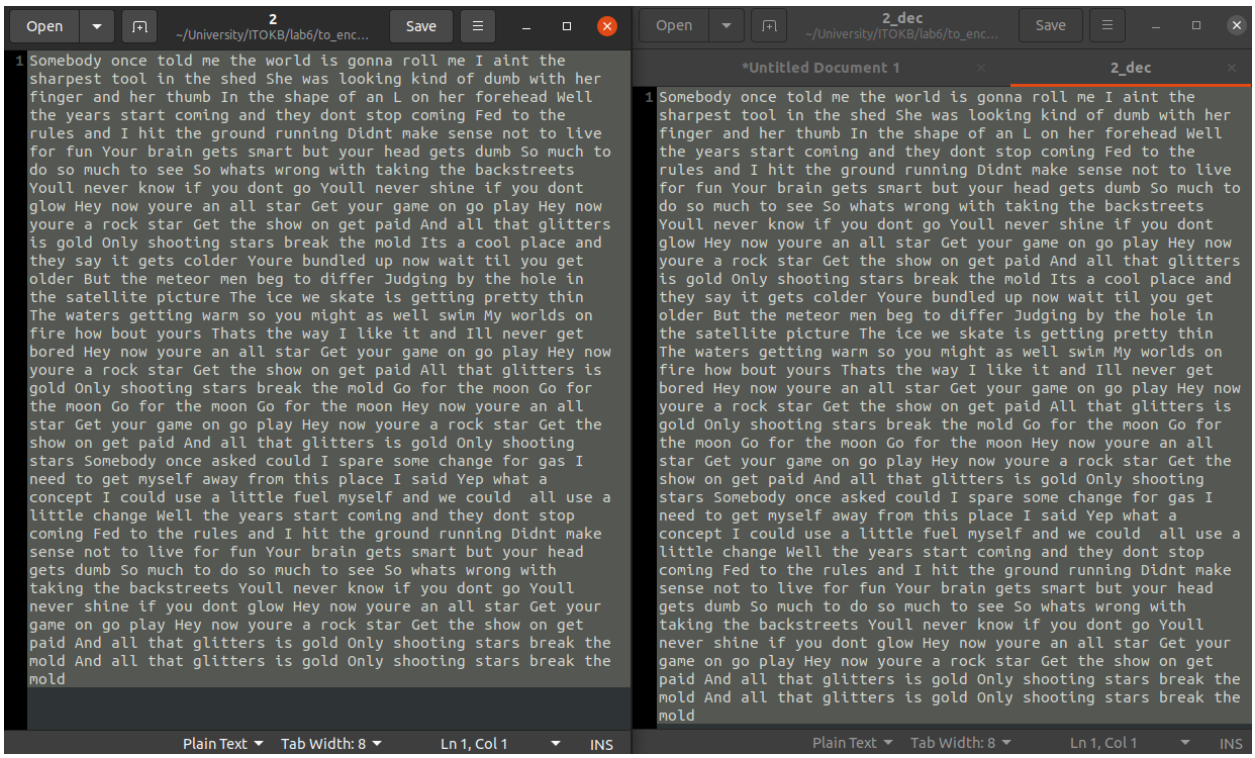

Рисунок 3.7 — Сравнение исходного и расшифрованного файла «2»

Для шифрования асимметричными алгоритмами необходим публичный ключ. Он создается с помощью флага «-pubout» и указания файлов секретного («-in») и выходного публичного («-out») ключей. Процесс создания публичных ключей показан на рисунке 3.8.

```
OpenSSL> rsa -in rsa_private_des_1024.pem -out rsa_pub_des_1024.pem -pubout
Enter pass phrase for rsa_private_des_1024.pem:
writing RSA key
OpenSSL> rsa -in rsa_private_des3_2048.pem -out rsa_pub_des3_2048.pem -pubout
Enter pass phrase for rsa_private_des3_2048.pem:
writing RSA key
OpenSSL> rsa -in rsa_private_aes256_4096.pem -out rsa_pub_aes256_4096.pem -pubout
Enter pass phrase for rsa_private_aes256_4096.pem:
writing RSA key
OpenSSL> rsa -in rsa_private_camellia256_8192.pem -out rsa_pub_camellia256_8192.pem -pubout
Enter pass phrase for rsa_private_camellia256_8192.pem:
writing RSA key
OpenSSL> dsa -in dsa_private_des_1024.pem -out dsa_pub_des_1024.pem -pubout
read DSA key
Enter pass phrase for dsa_private_des_1024.pem:
writing DSA key
OpenSSL> dsa -in dsa_private_des3_2048.pem -out dsa_pub_des3_2048.pem -pubout
read DSA key
Enter pass phrase for dsa_private_des3_2048.pem:
writing DSA key
OpenSSL> dsa -in dsa_private_aes256_4096.pem -out dsa_pub_aes256_4096.pem -pubout
read DSA key
Enter pass phrase for dsa_private_aes256_4096.pem:
writing DSA key
OpenSSL> dsa -in dsa_private_camellia256_8192.pem -out dsa_pub_camellia256_8192.pem -pubout
read DSA key
Enter pass phrase for dsa_private_camellia256_8192.pem:
writing DSA key
```

Рисунок 3.8 — Процесс создания публичных ключей RSA и DSA

Процесс шифрования посредством различных асимметричных алгоритмов представлен на рисунке 3.9. На рисунке 3.10 показаны файлы, используемые при шифровании. На рисунках 3.11 и 3.12 представлено сравнение исходных и расшифрованных файлов.

```
OpenSSL> rsautl -in 3 -out 3_enc -inkey rsa_pub_aes256_4096.pem -pubin -encrypt
OpenSSL> rsautl -in 3_enc -out 3_dec -inkey rsa_private_aes256_4096.pem -decrypt
Enter pass phrase for rsa_private_aes256_4096.pem:
OpenSSL> rsautl -in 4 -out 4_enc -inkey rsa_pub_camellia256_8192.pem -pubin -encrypt
OpenSSL> rsautl -in 4_enc -out 4_dec -inkey rsa_private_camellia256_8192.pem -decrypt
Enter pass phrase for rsa_private_camellia256_8192.pem:
```

Рисунок 3.9 — Шифрование с использованием асимметричных алгоритмов

Рисунок 3.10 — Файлы, используемые при шифровании симметричными алгоритмами

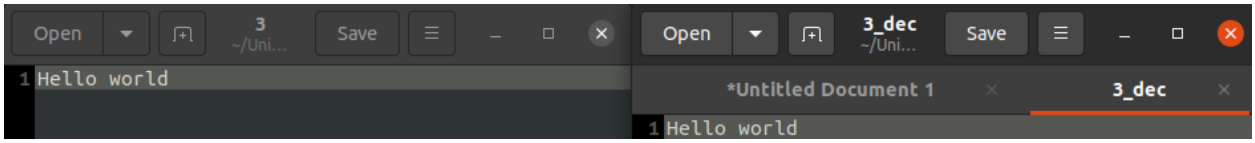

Рисунок 3.11 — Сравнение исходного и расшифрованного файла «3»

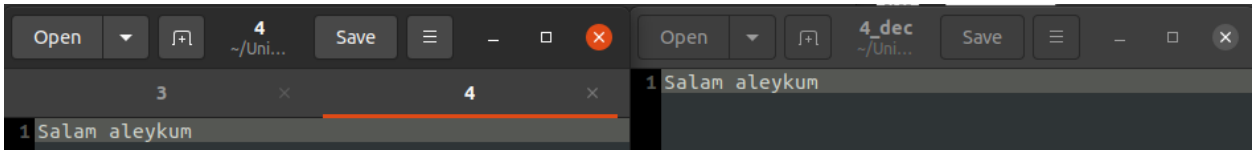

Рисунок 3.12 — Сравнение исходного и расшифрованного файла «4»

Для вычисления хэшей используется функция «dgst» с названием хэша в виде параметра «-*», где * - название хэша. Вычисленные хэш-суммы файлов представлены на рисунке 3.13.

```
OpenSSL> dgst -sha1 -c 5
SHA1(5)= d2:c6:47:61:ba:09:d3:5f:7e:b0:4f:9a:53:07:4d:bf:30:ec:d3:ac
OpenSSL> dgst -md5 6
MD5(6)= 8b6a344e7dbeab66d6053368bb1158e6
OpenSSL> dgst -sha256 -c 7
SHA256(7)= a9:97:82:a6:c2:c3:f7:74:a8:74:ee:14:9d:04:af:3f:e6:99:a6:b6:be:ee:83:
75:33:5a:e7:87:15:06:86:61
OpenSSL> dgst -blake2b512 8
BLAKE2b512(8)= 219a897eefba5bdd5f6df019c69abbd197cfbd31a72e42045d81e8c882082cd94
79e68d3ad8a4decd44a2e0daf75f52514cec6a7ad0d17dc9ed03c150f3ebb43
```

Рисунок 3.13 — Вычисление хэш-сумм различными алгоритмами

Задание 4: Создать самоподписанный сертификат X509. Изучить состав сертификата и назначение его компонентов.

Для создания самоподписанного сертификата X509 используется команда «req» с аргументом «-x509». Процесс создания сертификата приведен на рисунке 3.14.

```
OpenSSL> req -x509 -newkey rsa:4096 -keyout x509_key.pem -out x509_cert.cer -days 365
Generating a RSA private key
.....................................................................................................
.............++++
.....................................................................++++
writing new private key to 'x509_key.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:BY
State or Province Name (full name) [Some-State]:Minsk
Locality Name (eg, city) []:Minsk
Organization Name (eg, company) [Internet Widgits Pty Ltd]:BSUIR
Organizational Unit Name (eg, section) []:Faculty
Common Name (e.g. server FQDN or YOUR name) []:BSUIR
Email Address []:leksilonchikk@gmail.com
```

Рисунок 3.14 — Создание самоподписанного сертификата

В результате выполнения команды создается два файла:
1) x509_key.pem — зашифрованный секретный ключ
2) x509_cert.cer — файл сертификата
На рисунке 3.15 показаны созданные файлы сертификата и секретного ключа.

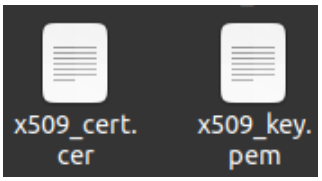

Рисунок 3.15 — Файлы секретного ключа и сертификата

Состав сертификата приведен на рисунках 3.16 и 3.17.

```
OpenSSL> x509 -in x509_cert.cer -noout -text
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            09:c8:f0:40:a0:b4:92:b8:b9:cb:36:af:ee:0d:2b:07:7c:5a:6d:ba
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = BY, ST = Minsk, L = Minsk, O = BSUIR, OU = Faculty, CN = BSUIR, emailAddress = le
ksilonchikk@gmail.com
        Validity
            Not Before: May  7 20:35:55 2021 GMT
            Not After : May  7 20:35:55 2022 GMT
        Subject: C = BY, ST = Minsk, L = Minsk, O = BSUIR, OU = Faculty, CN = BSUIR, emailAddress = l
eksilonchikk@gmail.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (4096 bit)
                Modulus:
                    00:b2:bb:99:7f:9a:67:ed:9d:7d:34:3b:59:25:3b:
```

Рисунок 3.16 — Состав сертификата, часть 1

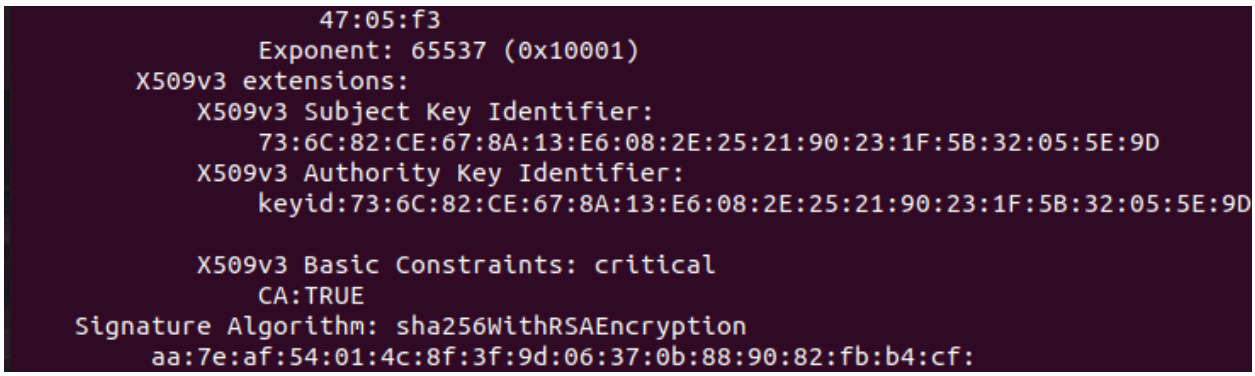

Рисунок 3.17 — Состав сертификата, часть 2

Структура сертификата:
- Данные:
  - Версия
  - Серийный номер
  - Идентификатор алгоритма подписи
  - Информация об издателе
  - Период действия:
    - Не ранее
    - Не позднее
  - Информация о субъекте
  - Информация об открытом ключе субъекта:
    - Алгоритм открытого ключа:
      - Открытый ключ субъекта
  - Дополнения
- Алгоритм подписи сертификата
  - Подпись сертификата