БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра интеллектуальных информационных технологий

ОТЧЁТ
по лабораторной работе №7
по дисциплине

«СРЕДСТВА И МЕТОДЫ ЗАЩИТЫ ИНФОРАМЦИИ В ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМАХ»

Выполнил студент гр. 121701

Мулярчик Д.С.

Проверил

Сальников Д. А.

Минск 2023

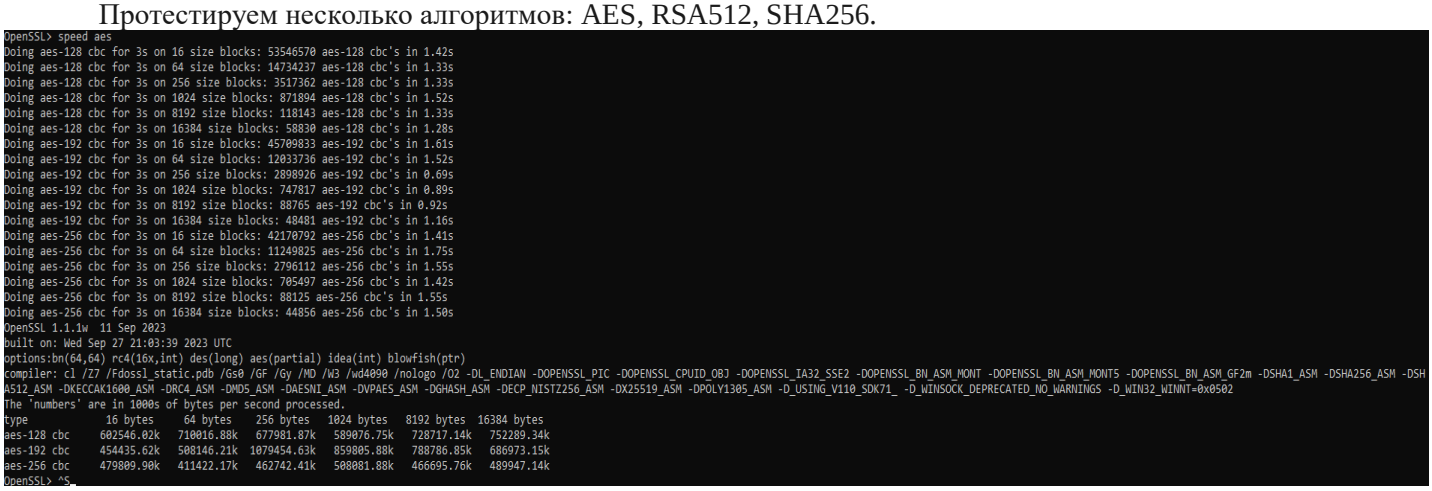
Тема: установка, использование и анализ специализированных средств криптографического пакета OpenSSL

Цель: установить пакет OpenSSL, сравнить скорость выполнения шифрования различных алгоритмов и детально разобраться в компонентах сертификата X.509

Задание:

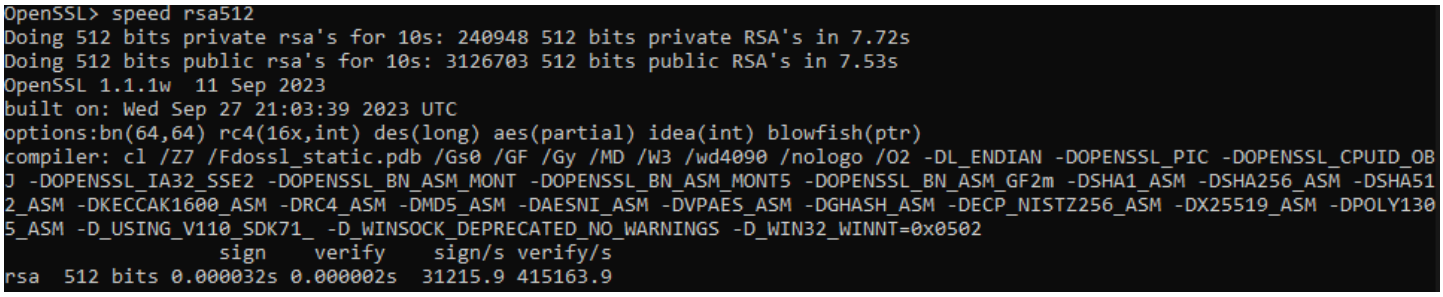
1. Установить OpenSSL на виртуальную машину (или рабочую версию ОС Windows 7/8/10 пользователя) и ознакомиться с возможностями библиотеки (команда «?»).
2. Выполнить тестирование скорости выполнения различных алгоритмов шифрования.
3. Создать криптографические ключи. Выбрать несколько произвольных файлов и выполнить:
    a. шифрование (зашифрование и расшифрование) посредством различных симметричных алгоритмов;
    b. шифрование (зашифрование и расшифрование) посредством различных асимметричных алгоритмов;
    c. хэширование различных файлов различными алгоритмами (обязательно md5 и sha1).
4. Создать самоподписанный сертификат X509. Изучить состав сертификата и назначение его компонентов.
5. Оформить отчет. В отчет поместить:
    a. результаты тестирования производительности;
    b. времена шифрования (выполнить сравнительную оценку скорости шифрования DES и AES, AES и RSA, объяснить полученные результаты);
    c. полученные хэш значения;
    d. сертификат с описанием его компонентов.

Протестируем несколько алгоритмов: AES, RSA512, SHA256.



Вывод разделен на разделы для разных алгоритмов шифрования (**aes-128-cbc**, **aes-192-cbc** и **aes-256-cbc**) и различных размеров блоков (16 байт, 64 байта, 256 байт, 1024 байта, 8192 байта и 16384 байта). Числа под каждым разделом указывают, сколько килобайт данных может быть обработано в секунду для
данного алгоритма и размера блока.

Тип	16 байт	64 байта	256 байт	1024 байта	8192 байта	16384 байта
Aes-128-cbc	602546	710016	677981	589076	728717	752289
Aes-192-cbc	454435	508146	1079454	859805	788786	686973
Aes-256-cbc	479809	411422	462742	508081	466695	489947



Сначала мы выполняем операции с **закрытыми** ключами длиной 512 бит в течение 10 секунд. За это время было выполнено 240948 операций с **закрытыми** ключами длиной 512 бит, и это заняло 7.72 секунд. Далее мы выполняем операции с **открытыми** ключами длиной 512 бит в течение 10 секунд. За это время было выполнено 3126703 операции с **открытыми** ключами длиной 2048 бит, и это заняло 7.53 секунд. В последней строке указаны результаты теста для операций RSA с ключами длиной 512 бит:

1) **sign** - Время, затраченное на операцию подписи (закрытый ключ), которое составляет приблизительно 0.000032 секунд.
2) **verify** - Время, затраченное на операцию проверки (открытый ключ), которое составляет приблизительно 0.000002 секунд.
3) **sign/s** - Скорость операций подписи, которая составляет приблизительно 31215.9 операций в секунду.
4) **verify/s** - Скорость операций проверки, которая составляет приблизительно 415163.9 операций в секунду

```
OpenSSL> speed sha64
speed: Unknown algorithm sha64
error in speed
OpenSSL> speed sha256
Doing sha256 for 3s on 16 size blocks: 17524302 sha256's in 2.06s
Doing sha256 for 3s on 64 size blocks: 9935997 sha256's in 2.23s
Doing sha256 for 3s on 256 size blocks: 4712443 sha256's in 1.92s
Doing sha256 for 3s on 1024 size blocks: 1478335 sha256's in 2.14s
Doing sha256 for 3s on 8192 size blocks: 199015 sha256's in 2.25s
Doing sha256 for 3s on 16384 size blocks: 99186 sha256's in 2.17s
OpenSSL 1.1.1w  11 Sep 2023
built on: Wed Sep 27 21:03:39 2023 UTC
options:bn(64,64) rc4(16x,int) des(long) aes(partial) idea(int) blowfish(ptr)
compiler: cl /Z7 /Fdossl_static.pdb /Gs0 /GF /Gy /MD /W3 /wd4090 /nologo /O2 -DL_ENDIAN -DOPENSSL_PIC -DOPENSSL_CPUID_OB
J -DOPENSSL_IA32_SSE2 -DOPENSSL_BN_ASM_MONT -DOPENSSL_BN_ASM_MONT5 -DOPENSSL_BN_ASM_GF2m -DSHA1_ASM -DSHA256_ASM -DSHA51
2_ASM -DKECCAK1600_ASM -DRC4_ASM -DMD5_ASM -DAESNI_ASM -DVPAES_ASM -DGHASH_ASM -DECP_NISTZ256_ASM -DX25519_ASM -DPOLY130
5_ASM -D_USING_V110_SDK71_ -D_WINSOCK_DEPRECATED_NO_WARNINGS -D_WIN32_WINNT=0x0502
The 'numbers' are in 1000s of bytes per second processed.
type             16 bytes     64 bytes    256 bytes   1024 bytes   8192 bytes  16384 bytes
sha256          135946.10k   284600.31k   627712.73k   707183.67k   724591.50k   748230.64k
```

В начале производятся операции SHA-256 с блоками размером 16, 64, 256, 1024, 8192, 16384 байт. Например, за 2.17 секунды было обработано 99186 хэшей SHA-256 с блоком 16384 байт. Затем следуют сведения о версии OpenSSL, настройках компиляции и информация о вашем процессоре. В конце вывода приведены результаты производительности операции SHA-256 для разных размеров блоков, измеряемые в тысячах байт в секунду (KB/s):

1. Для блоков размером 16 байт скорость составляет примерно 135946.1 KB/s.
2. Для блоков размером 64 байта скорость составляет примерно 284600.31 KB/s.
3. Для блоков размером 256 байт скорость составляет примерно 627712.73 KB/s.
4. Для блоков размером 1024 байта скорость составляет примерно 707183.67 KB/s.
5. Для блоков размером 8192 байта скорость составляет примерно 724591.5 KB/s.
6. Для блоков размером 16384 байта скорость составляет примерно 748230.64 KB/s

Из этих данных можно сделать общий вывод: если вам нужно шифрование данных, AES может быть более быстрым вариантом, особенно при использовании AES-192-CBC. Если требуется хэширование данных, SHA-256 предоставляет приемлемую скорость и является стандартным выбором для хэширования. RSA-512, как асимметричный алгоритм, будет медленнее, чем симметричное шифрование (AES) и хэширование (SHA-256). RSA обычно используется для подписи и проверки цифровых подписей, а не для шифрования больших объемов данных.

Сравнительная оценка скорости шифрования DES, AES, RSA:
AES обеспечивает хороший баланс между безопасностью и производительностью. Скорость зависит от длины ключа, и более длинные ключи требуют больше времени для обработки. DES уступает по скорости и безопасности по сравнению с AES, поэтому не рекомендуется для использования. RSA медленнее симметричных алгоритмов, и его скорость зависит от длины ключа.

Создание криптографических ключей:

Создадим файл, который мы хотим зашифровать:

```
PS C:\users\asus\Desktop> echo "skoro sessiya(" >> simsisya7.txt
PS C:\users\asus\Desktop>
```

Для AES создадим симметричный ключ, зашифруем и расшифруем файл:

```
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Установите последнюю версию PowerShell для новых функций и улучшения! https://aka.ms/PSWindows

PS C:\WINDOWS\system32> cd ../..
PS C:\> cd users
PS C:\users> cd asus
PS C:\users\asus> cd .\Desktop\
PS C:\users\asus\Desktop> echo "sessiya blizko" >> simsisya7.txt
PS C:\users\asus\Desktop> openssl rand -base64 32 >> sym_key.txt
PS C:\users\asus\Desktop> openssl enc -aes-256-cbc -salt -in simsisya7.txt -out "shifr.bin" -pass file:sym_key.txt
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
PS C:\users\asus\Desktop> type sym_key.txt
gqAGUp3mG56n527S3nIrOxb8JIctTuQ4ipqYyJCIfvg=
PS C:\users\asus\Desktop> type shifr.txt
type : Не удается найти путь "C:\users\asus\Desktop\shifr.txt", так как он не существует.
строка:1 знак:1
+ type shifr.txt
+ ~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (C:\users\asus\Desktop\shifr.txt:String) [Get-Content], ItemNotFoundExce
   ption
    + FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.GetContentCommand

PS C:\users\asus\Desktop> openssl enc -d -aes-256-cbc -in "shifr.bin" -out "rasshifr.txt" -pass file:sym_key.txt
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
PS C:\users\asus\Desktop> type rasshifr.txt
sessiya blizko
```

Для RSA создадим публичный и закрытый ключи, зашифруем и расшифруем файл:

```
PS C:\users\asus\Desktop> openssl genpkey -algorithm RSA -out key.pem
.....................................................................+++++
.........+++++
PS C:\users\asus\Desktop> openssl rsa -pubout -in key.pem -out public_key.pem
writing RSA key
PS C:\users\asus\Desktop> openssl rsautl -encrypt -pubin -inkey public_key.pem -in "simsisya7.txt" -out "shifr_rsa.bin"
PS C:\users\asus\Desktop> openssl rsautl -decrypt -inkey key.pem -in "shifr_rsa.bin" -out "rasshifr_rsa.txt"
PS C:\users\asus\Desktop> type rasshifr_rsa.txt
sessiya blizko
PS C:\users\asus\Desktop>
```

Захешируем файл при помощи MD5 и SHA256:

```
PS C:\users\asus\Desktop> openssl dgst -md5 -out md5hash.txt simsisya7.txt
PS C:\users\asus\Desktop> openssl dgst -sha256 -out sha256hash.txt simsisya7.txt
PS C:\users\asus\Desktop> type md5hash.txt
MD5(simsisya7.txt)= 6b24fbf466f4de1263536c5646e16dd8
PS C:\users\asus\Desktop> type sha256hash.txt
SHA256(simsisya7.txt)= b5b99141c2875a0fb628d0edce40f451907acb4e0555d87921a8736220b534e5
PS C:\users\asus\Desktop>
```

Создание самоподписанного сертификата X.509:

```
PS C:\users\asus\Desktop> openssl genpkey -algorithm RSA -out key.pem
...................+++++
.....................................................................................+++++
PS C:\users\asus\Desktop> openssl req -new -x509 -key key.pem -out cert.pem -days 30
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]:EN
State or Province Name (full name) [Some-State]:London
Locality Name (eg, city) []:West End
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Sdat Labu Corporation
Organizational Unit Name (eg, section) []:pzhpzh
Common Name (e.g. server FQDN or YOUR name) []:Kirill
Email Address []:assdas@asdasd.ru
PS C:\users\asus\Desktop>
```

Просмотрим сертификат:

```
PS C:\users\asus\Desktop> openssl x509 -in cert.pem -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            60:17:db:bb:10:c8:e2:c7:50:98:b1:14:0e:07:b1:00:2a:f8:e8:d5
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = EN, ST = London, L = West End, O = Sdat Labu Corporation, OU = pzhpzh, CN = Kirill, emailAddress = assdas@asdasd.ru
        Validity
            Not Before: Nov 23 14:02:57 2023 GMT
            Not After : Dec 23 14:02:57 2023 GMT
        Subject: C = EN, ST = London, L = West End, O = Sdat Labu Corporation, OU = pzhpzh, CN = Kirill, emailAddress = assdas@asdasd.ru
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:c8:00:ba:90:48:84:1b:a7:77:e0:e8:b8:8c:b0:
                    98:02:9b:65:a4:f4:cf:63:b6:10:0f:75:34:19:6d:
                    56:31:41:ae:54:dc:9e:99:42:1a:79:e0:35:31:84:
                    17:60:a5:e9:e7:5a:9f:35:03:05:ec:e2:8b:f8:a8:
                    cc:2c:a2:69:17:d9:fe:dc:9f:65:3d:cc:4d:7f:14:
                    81:aa:53:11:f3:25:4c:4c:9c:e9:1e:c5:bf:f3:84:
                    61:2f:57:d5:b8:ca:a1:1b:fd:44:5c:30:62:77:f7:
                    bf:d5:c2:0e:86:a0:a5:e5:a4:5b:50:e4:65:ff:4b:
                    e6:b0:58:50:12:86:e9:5a:3a:42:b7:44:07:54:9b:
                    6d:e6:88:ff:13:43:4b:60:74:05:5d:b5:76:db:2d:
                    a4:1c:79:69:2e:c2:b1:8b:59:77:a2:27:12:8d:26:
                    42:9d:94:73:7d:63:89:d6:b4:2b:94:43:7f:19:06:
                    36:a5:da:55:fc:5f:7f:e8:9a:49:3e:71:ec:46:09:
                    df:4a:5f:d0:5c:1a:f3:67:16:dd:f9:49:98:4b:b1:
                    ee:44:b2:f9:86:0b:e6:06:54:fe:22:a7:8d:45:05:
                    89:7d:10:ae:8b:aa:32:c7:69:b5:b4:91:61:51:9d:
                    b6:2b:2f:9f:49:fe:37:e9:82:c5:e1:ed:4d:8b:52:
                    15:d3
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                D8:C7:19:D3:90:0C:E5:D1:2B:F3:82:DA:23:8D:B9:62:4C:93:68:69
            X509v3 Authority Key Identifier:
                keyid:D8:C7:19:D3:90:0C:E5:D1:2B:F3:82:DA:23:8D:B9:62:4C:93:68:69

            X509v3 Basic Constraints: critical
                CA:TRUE
    Signature Algorithm: sha256WithRSAEncryption
         55:2c:87:1a:f3:61:15:3c:fc:57:a2:18:d0:ef:d8:1a:14:7a:
         86:f8:9c:a0:7b:54:33:bf:a8:45:dc:a7:35:71:a5:52:07:7e:
         c2:c6:e0:c4:82:69:ac:3f:3e:06:57:17:2e:6c:1a:8a:a3:d1:
         61:33:fb:f6:48:48:c1:7f:67:56:9f:86:28:56:1f:f8:f6:3b:
         57:a5:54:ef:25:df:a4:da:25:1e:ac:69:68:50:e0:9c:b9:37:
         c9:33:99:ad:a1:3d:e1:c4:b5:20:8e:d5:9f:21:85:9e:31:26:
         e2:af:e0:9c:06:1c:e0:ee:88:59:35:b2:f3:47:db:a0:53:65:
         ec:a8:d9:62:9b:21:de:e9:7a:bb:2e:83:69:cd:fc:e3:10:3c:
         03:81:e8:fb:36:7b:d7:57:b3:ee:f5:f1:b3:89:18:62:84:3c:
         a2:50:80:ec:bd:b5:2f:ac:01:7c:2c:08:ef:90:d1:d4:66:fc:
         3c:d7:6e:ec:5a:ce:ae:87:72:26:c8:99:fa:a7:0b:c2:3b:58:
         75:38:81:1d:51:fc:1e:91:b7:1e:b1:ce:98:d8:8a:ac:76:2e:
         05:a9:59:e0:1b:df:aa:1a:cc:50:93:5f:7a:b0:94:82:5a:4d:
         bc:a9:e0:a7:e1:71:04:f7:a9:a4:fd:6e:74:b1:6b:1c:ac:24:
         60:a5:5c:a2
```

Вывод:
Мы изучили криптографическую библиотеку OpenSSL: узнали какие алгоритмы шифрования она поддерживает, оценили скорость выполнения нескольких алгоритмов и выполнили сравнительную

оценку различных алгоритмов шифрования. Также мы самостоятельно выполнили зашифрование и дешифрование данных с помощью OpenSSL библиотеки, а также самостоятельно подписали сертификат X.509 и изучили его компоненты.