Лабораторная работа №11

Использование алгоритмов хеширования для подтверждения неизменности файла с применением OpenSSL.

Тема: Использование алгоритмов шифрования для сокрытия содержимого файла с применением OpenSSL

Цель работы: получить навык применения программного продукта OpenSSL для применения алгоритмов шифрования, изучить алгоритмы DES и RSA.

Алгоритм шифрования DES

Одной из наиболее известных криптографических систем с закрытым ключом является DES – Data Encryption Standard. Эта система первой получила статус государственного стандарта в области шифрования данных. Она разработана специалистами фирмы IBM и вступила в действие в США 1977 году.

Алгоритм DES широко использовался при хранении и передаче данных между различными вычислительными системами; в почтовых системах, в электронных системах чертежей и при электронном обмене коммерческой информацией. Стандарт DES реализовывался как программно, так и аппаратно. Предприятиями разных стран был налажен массовый выпуск цифровых устройств, использующих DES для шифрования данных. Все устройства проходили обязательную сертификацию на соответствие стандарту.

Длина ключа в алгоритме DES составляет 56 бит. Именно с этим фактом связана основная полемика относительно способности DES противостоять различным атакам. Как известно, любой блочный шифр с закрытым ключом можно взломать, перебрав все возможные комбинации ключей. При длине ключа 56 бит возможны 256 разных ключей. Если компьютер перебирает за одну секунду 1 000 000 ключей (что примерно равно 220), то на перебор всех 256 ключей потребуется 236 секунд или чуть более двух тысяч лет, что, конечно, является неприемлемым для злоумышленников.

Алгоритм RSA

Алгоритм шифрования с открытым ключом RSA был предложен одним из первых в конце 70-х годов XX века. Его название составлено из первых букв фамилий авторов: Р.Ривеста (R.Rivest), А.Шамира (A.Shamir) и Л.Эйдельмана (L.Adleman). Алгоритм RSA является, наверно, наиболее популярным и широко применяемым асимметричным алгоритмом в криптографических

системах. Алгоритм основан на использовании того факта, что задача разложения большого числа на простые сомножители является трудной.

Криптографическая система RSA базируется на следующих двух фактах из теории чисел:

- задача проверки числа на простоту является сравнительно легкой;

- задача разложения чисел вида n = pq (p и q — простые числа); на множители является очень трудной, если мы знаем только n, а p и q — большие числа (это так называемая задача факторизации, подробнее о ней см. "Основные положения теории чисел, используемые в криптографии с открытым ключом").

Алгоритм RSA представляет собой блочный алгоритм шифрования, где зашифрованные и незашифрованные данные должны быть представлены в виде целых чисел между 0 и n -1 для некоторого n.

Шифрование с использованием симметричных алгоритмов

Для шифрования используются следующие команды:

$ openssl enc -<алгоритм> <параметры>

$ openssl <алгоритм> <параметры>

Если используется первый вариант команды, один дефис перед именем алгоритма обязателен. Во втором варианте дефисов перед именем алгоритма быть не должно.

В качестве параметров используются следующие ключи (дефисы перед ключами обязательны):

- **in <имя_файла>**- имя входного файла;

- **out <имя_файла>** - имя выходного файла;

- **pass <источник_пароля>** - указание на источник парольной фразы. Синтаксис одинаков и для последующих команд:

- **e** - шифровать (по умолчанию);

- **d** - расшифровать;

- **base64** - кодировать в base64 после шифрования или декодировать из base64 перед расшифровкой;

- pubin — указание на то, что передается открытый ключ;

- pubout — вывести открытый ключ;

- des, -des3, -idea — зашифровать выходной файл с ключом соответствующим алгоритмом (будет запрошена парольная фраза; возможно использование только формата PEM для выходного ключа);

- encrypt — зашифровать открытым ключом;

- decrypt — расшифровать закрытым ключом.

Практическая часть

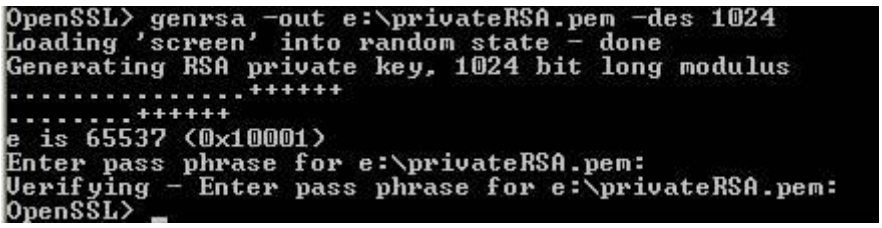
1. Изучите теоретическое руководство к лабораторной работе.

2. Подготовьте (создайте или выберите) текстовый файл с семантически понятным содержимым.

3. С помощью OpenSSL сгенерируйте ключ шифрования для алгоритма DES. Для этого введите команду: **genrsa [-out file] [-des] [-rand file] [bits]**

Openssl> genrsa –out e:\privateRSA.pem –des 1024

(После ввода команды система запросит от пользователя ввести пароль шифрования, а зетам повторить еще раз введенный пароль с целью избежать случайной ошибки со стороны пользователя при вводе пароля.)



В результате выполнения этой команды на диске будет создан файл **private.pem** со следующим содержимым:

-----BEGIN RSA PRIVATE KEY-----

Proc-Type: 4,ENCRYPTED

DEK-Info: DES-CBC,05D80AE4689B4866

…….

X4m5A+VKQVFW6YrP1TewxckDS4Iv1AvNxdcdJpK6mKBPdLRkyygjdw==

-----END RSA PRIVATE KEY-----

4. С помощью OpenSSL примените сгенерированный ключ шифрования и алгоритм DES к текстовому файлу. Измерьте время шифрования и запомните (запишите) его.

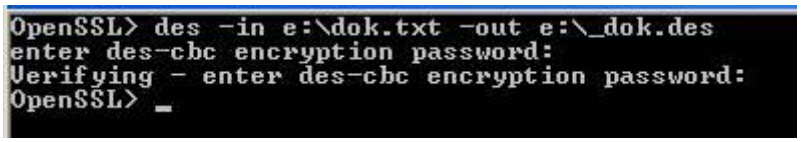
1) Для создания открытого ключа на основе созданного секретного ключа используйте команду rsa:

rsa –in filename [-out file] [-des] [-check] [-pubout]

(Параметр – **pubout** указывает на необходимость создания открытого ключа (**public key**) в файле, указанном в параметре –**out**. Параметр – **in** указывает на файл с секретным ключом.)

2) Зашифровать файл с использованием алгоритма DES:

openssl>des –in file –out file



5. Откройте текстовый файл, убедитесь, что содержимое не является семантически понятным.

6. Убедитесь, что при передаче неправильного ключа текст не расшифровывается.

7. Убедитесь, что при передаче правильного ключа текст расшифровывается корректно (расшифрованный текст совпадает с исходным).

Для дешифрования данных используется ключ –**d**.:

8. Выполните шифрование по алгоритмам DES-EDE и 3DES, используя только функцию DES, сравните свои результаты с результатами реализаций в OpenSSL, убедитесь в их совпадении (например, шифруйте вручную, а расшифровывайте OpenSSL), каждый раз при шифровании запоминайте время.

1) Пример шифрования файла симметричным алгоритмом des3:

Openssl> enc -des3 -e -in plain.txt -out enc.txt

enter des-ede3-cbc encryption password:

Verifying - enter des-ede3-cbc encryption password:

2) Расшифровать файл можно, заменив ключ '**- e**' ключом '**-d**':

Openssl> enc -des3 -d -in enc.txt -out newfile.txt

enter des-ede3-cbc decryption password:

9. Повторите действия 3-7 для алгоритма RSA.

10. Повторите действия 3-7 для алгоритма des3 и blowfish:

1) Шифрование файла по алгоритму des3

- зашифруем файл, используя алгоритм des3: **openssl des3 -in file -out file.des3**

- расшифруем полученный файл: **openssl des3 -d -in file.des3 -out file**

2) Шифрование файла по алгоритму **blowfish**

- зашифруем файл, используя алгоритм blowfish(bf), и закодируем base64: **openssl bf -a -in file -out file.bf64**

- теперь расшифруем его и обработаем сразу же base64: **openssl bf -a -d -in file.bf64 -out file**

3) Шифрование файла по алгоритму **blowfish (bf)** в режиме CBC (в режиме CBC ключ постоянно меняется, что затрудняет атаки)

- шифруем файл infile.txt по алгоритму **blowfish** в режиме CBC и записываем результат в файл **outfile.bf_cbc: openssl> enc -e -in infile.txt -out outfile.bf_cbc -bf-cbc**

- расшифровываем файл: **Openssl> enc -d -in outfile.bf_cbc -out dec_file.txt -bf-cbc**

- зашифруем файл, используя алгоритм blowfish(bf), и закодируем base64: **openssl> bf -a -in file -out file.bf64**

- расшифруем файл и обработаем сразу же base64: **openssl bf -a -d -in file.bf64 -out file**

11. Шифрование файла, используя алгоритм **AES256**

- создаем текстовый файл с расширением txt и шифруем его: **openssl> enc -e -aes-256-cbc -in file.txt -out file.encoded**

После ввода команды запрашивается пароль.

- можно указать пароль уже внутри команды: **Openssl> enc -e -aes-256-cbc -k password -in file -out file.encoded** (Этот метод может быть не надежным с точки зрения видимости пароля на экране для стоящего за спиной)

- выполняем дешифрование полученного файла **file.encoded: openssl> enc -d -aes-256-cbc -in file.encoded -out file_decrypto**

12. Сравните время шифрования с применением алгоритмов DES, DES-EDE, DES3, RSA.