

Лабораторная работа №7. Реализация шифра *RSA*

Цель работы: исследование структуры алгоритма и методики практической реализации криптосистемы шифрования *RSA*.

Теоретическая часть

Как известно, алгоритмы симметричного шифрования используют ключи относительно небольшой длины и поэтому могут быстро шифровать большие объёмы данных.

При использовании алгоритма симметричного шифрования отправитель и получатель применяют для шифрования и расшифрования данных один и тот же секретный ключ. Таким образом, алгоритмы симметричного шифрования основываются на предположении о том, что зашифрованное сообщение не сможет прочитать никто, кроме того кто обладает ключом для его расшифрования. При этом если ключ не скомпрометирован, то при расшифровании автоматически выполняется аутентификация отправителя, т.к. только он имеет ключ, с помощью которого можно зашифровать сообщение. Таким образом, для симметричных криптосистем актуальна проблема безопасного распределения симметричных секретных ключей. В связи с этим без эффективной организации защищённого распределения ключей использование обычной системы симметричного шифрования в вычислительных сетях практически невозможно.

Решением данной проблемы является использование асимметричных алгоритмов шифрования, называемых криптосистемами с открытым ключом. В них для зашифрования данных используется один ключ, называемый «открытым» а для расшифрования – другой называемый «закрытым или секретным». Следует иметь в виду, что ключ расшифрования не может быть определён из ключа зашифрования.

В асимметричных криптосистемах открытый ключ и криптограмма могут быть отправлены по незащищённым каналам. Концепция таких систем основана на применении однонаправленных функций.

В качестве примера однонаправленной функции может служить целочисленное умножение. Прямая задача – вычисление произведения двух больших целых чисел p и q , $n = p * q$. Это относительно несложная задача для ЭВМ.

Обратная задача – факторизация или разложение на множители большого целого числа практически неразрешима при достаточно больших значениях n .

Например, если $p \approx q$, а их произведение $n \approx 2664$, то для разложения этого числа на множители потребуется 223 операций, что практически невозможно выполнить за приемлемое время на современных ЭВМ.

Другим примером однонаправленной функции является модульная экспонента с фиксированным основанием и модулем.

Например, если $y = ax$, то естественно можно записать, что $x = \log_a(y)$.

Задача дискретного логарифмирования формулируется следующим образом. Для известных целых a, n, y следует найти такое число x , при котором $ax \pmod n = y$. Например, если $a = 2664$ и $n=2664$ нахождение показателя степени x для известного y потребует около 1026 операций, что также невозможно выполнить на современных ЭВМ.

В связи с тем, что в настоящее время не удалось доказать, что не существует эффективного алгоритма вычисления дискретного логарифма за приемлемое время, то модульная экспонента также условно отнесена к однонаправленным функциям.

Другим важным классом функций, используемых при построении криптосистем с открытым ключом являются, так называемые, однонаправленные функции с секретом. Функция относится к данному классу при условии, что она является однонаправленной и, кроме того, возможно эффективное вычисление обратной функции, если известен секрет.

В данной лабораторной работе исследуется криптосистема *RSA*, использующая модульную экспоненту с фиксированным модулем и показателем степени (т.е. однонаправленную функцию с секретом).

Задание к работе

Порядок выполнения работы соответствует, приведённой ниже, криптосистеме шифрования данных по схеме *RSA*.

Схема алгоритма шифрования данных *RSA*

3.1. Определение открытого «*e*» и секретного «*d*» ключей

3.1.1. Выбор двух взаимно простых больших чисел *p* и *q*

3.1.2. Определение их произведения: $n = p * q$

3.1.3. Определение функции Эйлера: $\phi(n) = (p-1)(q-1)$

3.1.4. Выбор открытого ключа *e* с учётом условий:

$$1 < e \leq \phi(n), \text{НОД}(e, \phi(n)) = 1$$

3.1.5. Определение секретного ключа *d*, удовлетворяющего условию $e * d \equiv 1 \pmod{\phi(n)}$, где $d < n$

3.2. Алгоритм шифрования сообщения *M* (действия отправителя)

3.2.1. Разбивает исходный текст сообщения на блоки *M1, M2,..., Mn* ($M_i = 0, 1, 2, \dots, n$)

3.2.2. Шифрует текст сообщения в виде последовательности блоков:

$$C_i = M_i^e \pmod{n}$$

3.2.3. Отправляет получателю криптограмму : *C1, C2,... Cn*

3.2.3. Получатель расшифровывает криптограмму с помощью секретного ключа *d* по формуле: $M_i = C_i^d \pmod{n}$

3.3. Процедуру шифрования данных рассмотрим на следующем примере (для простоты и удобства расчётов в данном примере использованы числа малой разрядности):

3.3.1. Выбираем два простых числа *p* и *q*, $p = 3, q = 11$;

3.3.2. Определяем их произведение (модуль) $n = p * q = 33$;

3.3.3. Вычисляем значение функции Эйлера $\phi(n) = (p-1)(q-1)$

$$\phi(n) = 2 * 10 = 20$$

3.3.4. Выбираем случайным образом открытый ключ с учётом выполнения

условий $1 < e \leq \phi(n)$ и $\text{НОД}(e, \phi(n)) = 1$, $e = 7$;

3.3.5. Вычисляем значение секретного ключа d , удовлетворяющего условию

$$e * d \equiv 1 \pmod{\phi(n)}, 7 * d \equiv 1 \pmod{20}; d = 3;$$

3.3.6. Отправляем получателю пару чисел ($n = 33$, $e = 7$);

Представляем шифруемое сообщение M как последовательность целых чисел **312**.

3.3.7. Разбиваем исходное сообщение на блоки $M1 = 3$, $M2 = 1$, $M3 = 2$;

3.3.8. Шифруем текст сообщения, представленный в виде последовательности блоков: $Ci = Mi^e \pmod{n}$

$$C1 = 3^7 \pmod{33} = 2187 \pmod{33} = 9,$$

$$C2 = 1^7 \pmod{33} = 1 \pmod{33} = 1,$$

$$C3 = 2^7 \pmod{33} = 128 \pmod{33} = 29.$$

3.3.9. Отправляем криптограмму $C1 = 9$, $C2 = 1$, $C3 = 29$.

3.3.10. Получатель расшифровывает криптограмму с помощью секретного ключа d по формуле: $Mi = Ci^d \pmod{n}$

$$M1 = 9^3 \pmod{33} = 729 \pmod{33} = 3$$

$$M2 = 1^3 \pmod{33} = 1 \pmod{33} = 1$$

$$M3 = 29^3 \pmod{33} = 24389 \pmod{33} = 2.$$

Полученная последовательность чисел **312** представляет собой исходное сообщение M .

Составить блок-схему и программу алгоритма шифрования RSA . Представить листинг программы шифрования заданного сообщения M с использованием алгоритма RSA .