Теорка лаба 3

1. **Алгоритм шифрования RSA**  
   Начнем с того, что это *асиметрическое шифрование*!!!!!!!!!!!!!!!!!!!  
     
   Шаг первый. Подготовка ключей.   
   Нужно сделать предварительные действия: сгенерировать публичный и приватный ключ.   
   - выбираем два простых числа, например p = 3, q = 7.   
   - вычисляем модуль – произведение наших чисел: n = p \* q = 3 \* 7 = 21  
   - вычисляем функцию Эйлера: ф = (p – 1) \* (q – 1) = 2 \* 6 = 12  
   - выбираем число е, которое должно соответствовать критериям:   
    1. Оно должно быть простое   
    2. Оно должно быть меньше ф. Имеем варианты: 3, 5, 7, 11  
    3. Оно должно быть взаимно простое с ф.   
   таким образом у нас остаются варианты: 5, 7, 11. Выбираем е = 5, это так называемая открытая экспонента.   
     
   Теперь пара чисел {e, n} = {5, 21} – наш открытый ключ. Он отправляется собеседнику, чтоб зашифровать сообщение, но это еще не все, нужно получить закрытый ключ.   
     
   Мне нужно получить число d, обратное e по модулю ф. Тоесть остаток от деления по модулю ф произведения d \* e, должен быть равен единице. Исходя из этого, d может быть равен 5, но чтоб не путать с e, возьмем d = 17.   
   Пара {d, n} = {17, 21} – наш секретный ключ, который мы не передаем никому. Он нужен для расшифровки зашифрованного открытым ключом сообщения.   
     
   Шаг второй. Шифрование.   
   Допустим сообщение – число 19. У нас имеется открытый ключ {e, n} = {5, 21}. Шифрование выполняется по следующему алгоритму:   
   - возводите ваше вообщение в степень e и выполняете деление по модулю n. Тоесть, возводим 19 в степень 5 = 2476099 и берем остаток от деления на 21, получаем 10.   
   10 – наше зашифрованное сообщение.   
   *Важное уточнение!!!!!!! Число в сообщении не должно быть больше n.*   
     
   Использование на практике:   
   Допустим есть слово КРОТ. Нужно перевести его в числовое представление, например код букв в системе ASCII. Для каждой буквы пишем число, и шифруем эти числа с помощью алгоритма, приведенного выше.   
   *Важно!!! Числа для генерации ключа должны быть достаточно большими.*
2. **Алгоритм дешифрования RSA**  
   Мы получили зашифрованные данные (10) и у нас есть приватный ключ {d, n} = {17, 21}.   
   Стоит обратить внимание на то, что открытый ключ не может расшифровать сообщение. А закрытый ключ мы никому не говорили, в этом и прелесть асиметрического шифрования.   
   Начинаем раскодировать:   
   - делаем операцию, очень похожую на кодирование, но вместо e используем d. Возводим сообщение в степень d: 10 в 17 степень, вычисляем остаток от деления на 21 и получаем число 19. Число 19 и есть нашим оригинальным сообщением.
3. **Расширенный алгоритм Эвклида**  
   1. Выберем начальную единичную матрицу E = {{1, 0}, {0, 1}}.   
   2. Вычислим остаток от деления r = a mod b.   
   3. Если r = 0, то решение – второй столбец матрицы E – решение задачи (вектор {x, y}).   
   4. С уравнения a = b\*q + r выведем q = (a - r) / b  
   5. Заменим E = E \* {{0, 1}, {1, -q}}. Переходим к пункту 2.
4. **Криптостойкость RSA**  
   До сих пор математически не доказано, что для восстановления m по e и n обязательно нужно раскладывать число n на множители. Но если будет открыт метод криптоанализа RSA, позволяющий получить d, то этот метод можно будет также использовать для разложения на простые множители числа n. Такие алгоритмы в настоящее время не найдены. Существует несколько достаточно эффективных способов взлома, но они работают только при выборе определенных параметров n, e и d, а также при неосторожном обращении с секретными ключами. Грамотным выбором параметров и четкой системой использования секретных ключей эффективность подобного рода атак сводится к нулю. Таким образом, считается, что стойкость рассматриваемого алгоритма полностью зависит от трудоемкости разложения на множители больших чисел.
5. **Использование RSA**  
   Используют для цифровой подписи или шифровки другого более простого ключа, поскольку достаточно большие вычислительные объемы.

Теорка лаба 4

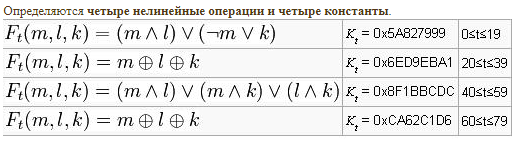
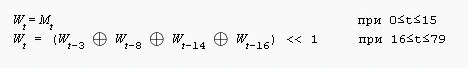
1. **Для чего используется криптографическое хеширование? Какие алгоритмы для него используются?**   
   Хеширование генерирует из сообщения произвольной длины хеш-значение заданной длины. Поскольку одинаковое сообщение генерирует одинаковый хеш и необратимо, то хеширование необходимо для проверки целостности/подлинности или для секьюрного хранения данных.   
   Используемые алгоритмы:   
   SHA-1/SHA-256/SHA-512.
2. **Описать принципы алгоритма SHA-1, его преимущества и недостатки.**  
   Secure Hash Algorithm 1 — алгоритм криптографического хеширования. Для входного сообщения произвольной длины алгоритм генерирует 160-битное хеш-значение, называемое также дайджестом сообщения. Используется во многих криптографических приложениях и протоколах. Также рекомендован в качестве основного для государственных учреждений в США.

Исходное сообщение разбивается на блоки по 512 бит в каждом. Последний блок дополняется до длины, кратной 512 бит. Сначала добавляется 1, а потом нули, чтобы длина блока стала равной 512 — 64 бит. В оставшиеся 64 бита записывается длина исходного сообщения в битах. Дополнение последнего блока осуществляется всегда, даже если сообщение уже имеет нужную длину. Таким образом, число добавляемых битов находится в диапазоне от 1 до 512. Инициализируются пять 32-битовых переменных   
A = a = 0x67452301

B = b = 0xEFCDAB89

C = c = 0x98BADCFE

D = d = 0x10325476

E = e = 0xC3D2E1F0  
  
Главный цикл итеративно обрабатывает каждый 512-битный блок. Итерация состоит из четырех этапов по двадцать операций в каждом. Блок сообщения преобразуется из 16 32-битовых слов Mi в 80 32-битовых слов Wj по следующему правилу:  


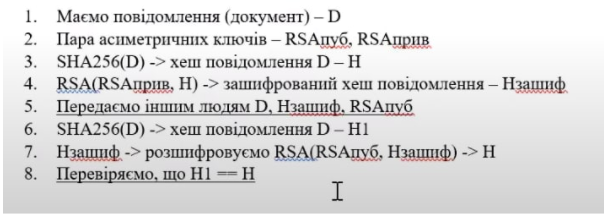
1. **С какой целью используется электронная цифровая подпись (ЭЦП)?**   
   Электронная цифровая подпись (ЭЦП) – мощное средство контроля подлинности информации в электронном виде, обеспечения целостности электронных данных, подтверждения их авторства и актуальности. Электронная цифровая подпись – это информационный объект, создаваемый для подписываемых данных, позволяющий удостовериться в целостности и аутентичности этих данных.
2. **Описать принцип работы ЭЦП**  
   Широко применяемая в настоящее время технология электронной подписи основана на асимметричном шифровании с открытым ключом и опирается на следующие принципы:

Можно сгенерировать пару очень больших чисел (открытый ключ и закрытый ключ) так, чтобы, зная открытый ключ, нельзя было вычислить закрытый ключ за разумный срок. Механизм генерации ключей строго определён и является общеизвестным. При этом каждому открытому ключу соответствует определённый закрытый ключ. Если, например, Иван Иванов публикует свой открытый ключ, то можно быть уверенным, что соответствующий закрытый ключ есть только у него.

Имеются надёжные методы шифрования, позволяющие зашифровать сообщение закрытым ключом так, чтобы расшифровать его можно было только открытым ключом. Механизм шифрования является общеизвестным.

Если электронный документ поддается расшифровке с помощью открытого ключа, то можно быть уверенным, что он был зашифрован с помощью уникального закрытого ключа. Если документ расшифрован с помощью открытого ключа Ивана Иванова, то это подтверждает его авторство: зашифровать данный документ мог только Иванов, т.к. он является единственным обладателем закрытого ключа.

Однако шифровать весь документ было бы неудобно, поэтому шифруется только его хеш — небольшой объём данных, жёстко привязанный к документу с помощью математических преобразований и идентифицирующий его. Шифрованный хеш и является электронной подписью.



1. **Какие существуют стандарты ЭЦП? Чем они отличаются? Где целесообразнее их использовать?**Американские стандарты электронной цифровой подписи: DSA, ECDSA

Российские стандарты электронной цифровой подписи: ГОСТ Р 34.10-94 (в настоящее время не действует), ГОСТ Р 34.10-2001

Украинский стандарт электронной цифровой подписи: ДСТУ 4145-2002

Стандарт PKCS#1 описывает, в частности, схему электронной цифровой подписи на основе алгоритма RSA

схема Шнорра

ElGamal

Вероятностная схема подписи Рабина

Отличаются они используемыми алгоритмами.

1. **На основе какой особенности хеширования может быть осуществлена атака на данные, защищенные с помощью ЭЦП? Какое название носит парадокс, на котором эта атака базируется? В чем его математическая подоплека? Як повысить защищенность ЭЦП от такой атаки?**   
   Атака “дней рождения” полагается на высокую вероятность нахождения коллизий между случайными попытками и порядком перестановок

Коллизии:

1 — Для N почти невозможно подобрать такое сообщение М, для которого H(N) = H(M)

2 — почти невозможно найти пару сообщений, для которых H(m1) = H(m2)

Отличия в том, что идут две разных атаки