

Лабораторная работа № 4

Однонаправленные хэш-функции. Электронная цифровая подпись.

Цель работы: Изучить различные алгоритмы однонаправленного хэширования данных, основанные на симметричных блочных алгоритмах шифрования. Ознакомиться со схемами цифровой подписи и получить навыки создания и проверки подлинности электронной цифровой подписи.

Однонаправленные хэш-функции

Однонаправленная функция $H(M)$ применяется к сообщению произвольной длины и возвращает значение фиксированной длины h :

$$h = H(M),$$

где h имеет длину m .

Многие функции позволяют вычислять значение фиксированной длины по входным данным произвольной длины, но у однонаправленных хэш-функций есть дополнительные свойства, делающие их однонаправленными:

- зная M , легко вычислить h . Зная h , трудно определить M , для которого $H(M) = h$;
- зная M , трудно определить другое сообщение M' , для которого $H(M) = H(M')$.

Смысл однонаправленных хэш-функций состоит в обеспечении для уникального идентификатора («отпечатка пальца»).

В некоторых приложениях однонаправленности недостаточно, необходимо, чтобы выполнялось другое требование, называемое **устойчивостью к столкновению**: должно быть трудно найти два случайных сообщения M и M' , для которых $H(M) = H(M')$.

Это возможно сделать методом дня рождения. Он основан не на поиске другого сообщения M' , для которого $H(M) = H(M')$, а на поиске двух случайных сообщений M и M' , для которых $H(M) = H(M')$.

Следующий протокол, впервые описанный Гидеоном Ювалом, показывает, как, если требование устойчивости к столкновению не выполняется, Алиса может использовать вскрытие методом дня рождения для обмана Боба:

1. Алиса готовит две версии контракта: одну, выгодную для Боба, и другую, приводящую его к банкротству.
2. Алиса вносит несколько незначительных изменений в каждый документ и вычисляет хэш-функции. (Этими изменениями могут быть действия, подобные следующим: замена «пробела» комбинацией «пробел» – «забой» – «пробел», вставка одного-двух «пробелов» перед возвратом каретки и т.д. Делая или не делая по одному изменению в каждой из 32 строк, Алиса может легко получить 2^{32} различных документа).
3. Алиса сравнивает хэш-значения для каждого изменения в каждом из двух документов, разыскивая пару, для которой эти значения совпадают. (Если выходом хэш-функции является всего лишь 64-разрядное значение, Алиса, как правило, сможет найти совпадающую пару, сравнив 2^{32} версий каждого документа). Она восстанавливает два документа, дающих одинаковое хэш-значение.
4. Алиса получает подписанную Бобом выгодную для него версию контракта, используя протокол, которым он подписывает только хэш-значения.
5. Спустя некоторое время, Алиса подменяет контракт, подписанный Бобом, другим, который он не подписывал. Теперь она может убедить арбитра в том, что Боб подписал другой контракт.

Длина однонаправленных хэш-функций

64-битные хэш-функции слишком малы, чтобы противостоять вскрытию методом дня рождения. Более практичны однонаправленные хэш-функции, выдающие 128-битные хэш-значения. При этом, чтобы найти два документа с одинаковыми хэш-значениями для вскрытия методом дня

рождения, придётся хэшировать 2^{64} случайных документа, что недостаточно, если нужна длительная безопасность.

Для удлинения хэш-значений, выдаваемых конкретной хэш-функцией, был предложен следующий метод:

1. Для сообщения с помощью одной из однонаправленных хэш-функций генерируется хэш-значение.
2. Хэш-значение добавляется к сообщению.
3. Генерируется хэш-значение объединения сообщения и хэш-значения этапа 1.
4. Создаётся большее хэш-значение, состоящее из объединения хэш-значения этапа 1 и хэш-значения этапа 3.
5. Этапы 1–4 повторяются нужное количество раз для обеспечения требуемой длины хэш-значения.

Обзор однонаправленных хэш-функций

Нелегко построить функцию, вход которой имеет произвольный размер, а тем более сделать её однонаправленной. В реальном мире однонаправленные хэш-функции строятся на идее функции сжатия. Такая однонаправленная функция выдаёт хэш-значение длины n при заданных входных данных большей длины m . Входами функции сжатия являются блок сообщения и выход предыдущего блока текста (см. рисунок 1). Выход представляет собой хэш-значение всех блоков до этого момента, т.е. хэш-значение блока M_i равно

$$h_i = f(M_i, h_{i-1}).$$

Это хэш-значение вместе со следующим блоком сообщения становится следующим входом функции сжатия. Хэш-значением всего сообщения будет хэш-значение последнего блока.

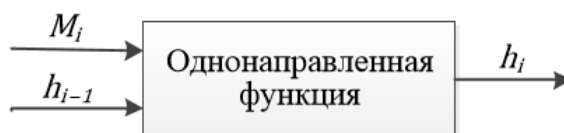


Рис 1 – Однонаправленная функция

Хэшируемый вход должен каким-то способом содержать бинарное представление длины всего сообщения. Таким образом, преодолевается потенциальная проблема, вызванная тем, что сообщения различной длины могут давать одно и то же хэш-значение. Иногда такой метод называется **MD-усилением**.

В качестве однонаправленных хэш-функций можно использовать симметричные блочные алгоритмы шифрования. Самый очевидный способ – это шифрование сообщения в режиме CBC и CFB с помощью фиксированного ключа и IV, хэш-значением будет последний блок шифротекста.

Более хороший способ использует в качестве ключа блок сообщения, в качестве входа – предыдущее хэш-значение, а выходом служит текущее хэш-значение.

Действительные хэш-функции ещё сложнее. Размер блока обычно совпадает с длиной ключа, и размером хэш-значения будет длина блока. Т.к. большинство блочных алгоритмов 64-битные, то спроектирован ряд схем, дающих хэш-значение, в два раза большее длины блока.

При условии, что хэш-функция правильна, безопасность этой схемы основана на безопасности используемой блочной функции. Однако есть и исключения. Дифференциальный криптоанализ лучше работает против блочных функций в хэш-значениях, чем против блочных функций, используемых для шифрования: ключ известен, поэтому можно использовать различные приёмы. Для успеха нужна только одна правильная пара, и можно генерировать столько выбранного открытого текста, сколько нужно.

Полезной мерой для хэш-функций, основанных на блочных шифрах, является скорость хэширования или количество n -битовых блоков сообщения (n – это размер блока алгоритма), обрабатываемых при шифровании. Чем выше скорость хэширования, тем быстрее алгоритм.

Схемы, в которых длина хэш-значения равна длине блока

Общая схема:

$H_0 = l_H$, где l_H – случайное начальное значение, задаваемое пользователем, или, например, длина сообщения.

$$H_i = E_A(B) \oplus C$$

где A , B и C могут быть либо M_i , H_{i-1} , $(M_i \oplus H_{i-1})$, либо константы (возможно, равные 0). H_0 – это некоторое случайное начальное число l_H . Сообщение разбивается на обрабатываемые отдельно части в соответствии с размером блока M_i :



Рис 2 – Общая схема

Три различные переменные (A , B , C) могут принимать одно из четырёх возможных значений, поэтому всего существует 64 варианта схем этого типа.

Далее приведены четыре схемы безопасных хэш-функций:

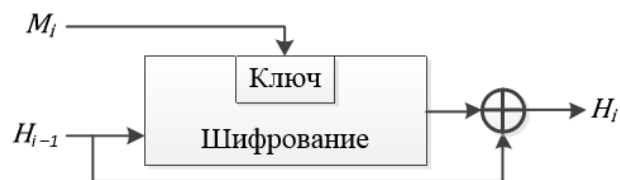


Рис 3 – Схема №1

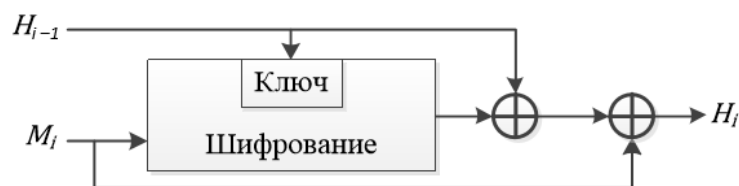


Рис 4 – Схема №2

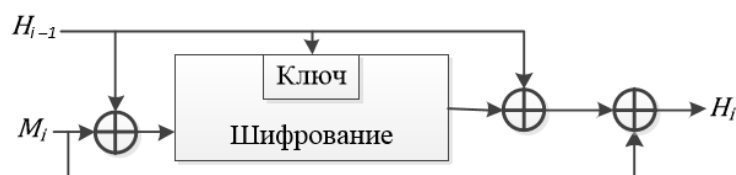


Рис 5 – Схема №3

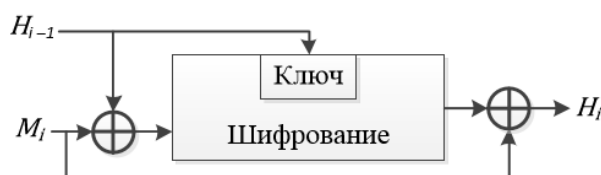


Рис 6 – Схема №4

Схемы, в которых длина хэш-значения равна удвоенной длине блока

Схема Preneel-Bosselaers-Govaerts-Vandewalle

При 64-битном блочном алгоритме схема выдаёт два 64-битных хэш-значения G_i и H_i , объединение которых даёт 128-битное хэш-значение. У большинства блочных алгоритмов длина блока равна 64 битам. Два соседних блока L_i и R_i , (размер каждого равен размеру блока) хэшируются вместе.

$$G_0 = L_G, H_0 = I_H$$

$$G_i = E_{L_i \oplus H_{i-1}}(R_i \oplus G_{i-1}) \oplus R_i \oplus G_{i-1} \oplus H_{i-1}$$

$$H_i = E_{L_i \oplus R_i}(H_{i-1} \oplus G_{i-1}) \oplus L_i \oplus G_{i-1} \oplus H_{i-1}$$

где I_G и I_H – два случайных начальных значения.

Схема Quisquater-Girault

Эта схема генерирует хэш-значение, в два раза большее длины блока. Она использует два хэш-значения G_i и H_i , и хэширует вместе два блока – L_i и R_i .

$$G_0 = L_G, H_0 = I_H$$

$$W_i = E_{L_i}(G_{i-1} \oplus R_i) \oplus R_i \oplus H_{i-1}$$

$$G_i = E_{R_i}(W_i \oplus L_i) \oplus G_{i-1} \oplus H_{i-1} \oplus L_i$$

$$H_i = W_i \oplus G_{i-1}$$

где I_G и I_H – два случайных начальных значения.

Электронная цифровая подпись

На протяжении многих веков при ведении деловой переписки, заключении контрактов и оформлении любых других важных бумаг подпись ответственного лица или исполнителя была неременным условием

признания его статуса или неоспоримым свидетельством его важности.

Подобный акт преследовал две цели:

- гарантирование истинности письма посредством сличения подписи с имеющимся образцом;
- гарантирование авторства документа (с юридической точки зрения).

Выполнение данных требований основывается на следующих свойствах подписи:

- подпись аутентична, т.е. с её помощью получателю документа можно доказать, что она принадлежит подписывающему;
- подпись служит доказательством, что только тот человек, чей автограф стоит на документе, мог подписать данный документ, и никто другой не смог бы этого сделать;
- подпись непереносима, т.е. она является частью документа, и поэтому перенести её на другой документ невозможно;
- документ с подписью является неизменяемым, т.е. после подписания его невозможно изменить, оставив данный факт незамеченным;
- подпись неоспорима, т.е. человек, подписавший документ, в случае признания экспертизой, что именно он засвидетельствовал данный документ, не может оспорить факт подписания;
- любое лицо, имеющее образец подписи, может удостовериться в том, что данный документ подписан владельцем подписи.

С переходом к безбумажным способам передачи и хранения данных, а также с развитием систем электронного перевода денежных средств, в основе которых – электронный аналог бумажного платёжного поручения, проблема виртуального подтверждения аутентичности документа приобрела особую остроту. Развитие любых подобных систем теперь немыслимо без существования электронных подписей под электронными документами. Однако применение и широкое распространение **электронно-цифровых**

подписей (ЭЦП) повлекло целый ряд правовых проблем. Так, ЭЦП может применяться на основе договоренностей внутри какой-либо группы пользователей системы передачи данных, и в соответствии с договоренностью внутри данной группы ЭЦП должно иметь юридическую силу. Но будет ли электронная подпись иметь доказательную силу в суде, например, при оспаривании факта передачи платежного поручения?

Рассмотрим существующие схемы электронной цифровой подписи.

Схема 1

Данная схема предполагает шифрование электронного документа на основе симметричных алгоритмов и предусматривает наличие в системе третьего лица (арбитра), пользующегося доверием участников обмена. Взаимодействие пользователей данной системой производится по следующей схеме (см. рисунок 7):

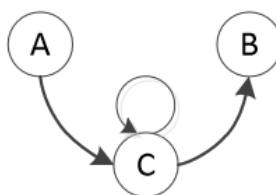


Рис 7 – Основные методы построения ЭЦП. Схема 1

Участник А зашифровывает сообщение своим секретным ключом K_A , знание которого разделено с арбитром (С на рисунке 7), затем зашифрованное сообщение передаётся арбитру с указанием адресата данного сообщения (информация, идентифицирующая адресата, передаётся также в зашифрованном виде).

Арбитр расшифровывает полученное сообщение ключом K_A , производит необходимые проверки и затем зашифровывает его секретным ключом участника В (K_B). Далее зашифрованное сообщение посылается участнику В вместе с информацией, что оно пришло от участника А.

Участник В расшифровывает данное сообщение и убеждается в том, что отправителем является участник А.

Авторизацией документа в данной схеме считается сам факт шифрования электронного документа секретным ключом и передачи

зашифрованного электронного документа арбитру. Основным преимуществом этой схемы является наличие третьей стороны, исключающей какие-либо спорные вопросы между участниками информационного обмена, т.е. в данном случае не требуется дополнительной системы арбитража ЭЦП. Недостатком схемы являются необходимость участия в обмене информацией третьей стороны и использование симметричных алгоритмов шифрования. На практике эта схема не получила широкого распространения.

Схема 2

Фактом подписания документа в данной схеме (см. рисунок 8) служит шифрование документа секретным ключом его отправителя. Здесь используются асимметричные алгоритмы шифрования.

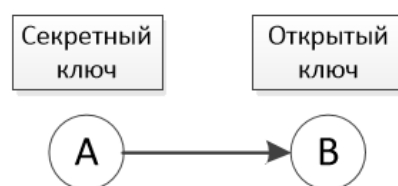


Рис 8 – Основные методы построения ЭЦП. Схема 2

Вторая схема используется довольно редко, поскольку длина электронного документа может оказаться очень большой (шифрование асимметричным алгоритмом может оказаться неэффективным по времени). Но в этом случае в принципе не требуется наличие третьей стороны, хотя она и может выступать в роли сертификационного органа открытых ключей пользователя.

Схема 3

Наиболее распространённая схема ЭЦП использует шифрование окончательного результата обработки электронного документа хэш-функцией при помощи асимметричного алгоритма. Структурная схема такого варианта построения ЭЦП представлена на рисунке 9:

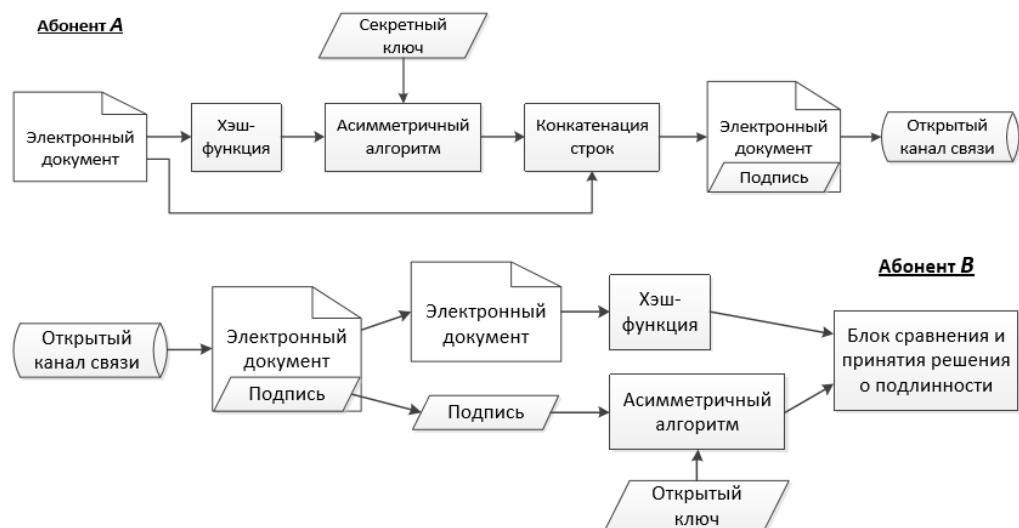


Рис 9 – Основные методы построения ЭЦП. Схема 3

Процесс генерации ЭЦП происходит следующим образом.

Участник А вычисляет хэш-код от электронного документа. Полученный хэш-код проходит процедуру преобразования с использованием секретного ключа участника А. После этого полученное значение (которое и является ЭЦП) вместе с электронным документом отправляется участнику В.

Участник В должен получить электронный документ с ЭЦП и сертифицированный открытый ключ участника А, а затем произвести расшифрование на нём ЭЦП. Электронный документ подвергается операции хэширования, после чего результаты сравниваются, и если они совпадают, то ЭЦП признается истинной, в противном случае – ложной.

В настоящее время применяется несколько алгоритмов цифровой подписи:

- RSA (наиболее популярен);
- Digital Signature Algorithm, DSA (алгоритм цифровой подписи американского правительства, который применяют в стандарте цифровой подписи (Digital Signature Standard, DSS), также используется часто);
- алгоритм Эль-Гамала (иногда можно встретить);
- алгоритм, который применяют в стандарте ГОСТ Р34.10-94 (в основе лежит DSA и является вариацией подписи Эль-Гамала);

- так же существуют алгоритмы подписей, в основе которых лежит криптография эллиптических кривых; они похожи на все прочие, но в некоторых ситуациях работают эффективнее.

Электронная подпись RSA

Для осуществления подписи сообщения $M = M_1 M_2 M_3 \dots M_n$ необходимо вычислить хэшфункцию $m = h(M_1 M_2 M_3 \dots M_n)$, которая ставит в соответствие сообщению M число m . На следующем шаге достаточно снабдить подписью только число m , и эта подпись будет относиться ко всему сообщению M [1, 2, 4].

Далее по алгоритму RSA вычисляются ключи (e, n) и (d, n) .

Затем вычисляется $s = m^d \bmod n$ (d – секретная степень).

Число s – это и есть цифровая подпись. Она просто добавляется к сообщению и получается подписанное сообщение $\langle M, s \rangle$.

Теперь каждый, кто знает параметры подписавшего сообщение (т.е. числа e и n), может проверить подлинность подписи.

Для этого необходимо проверить выполнение равенства $h(M) = s^e \bmod n$.

Алгоритм Эль-Гамала

Для генерации пары ключей сначала выбирается большое простое число p , один из его первообразных корней g и случайное число x ($g < p$, $x < p$). Затем вычисляется $y = g^x \bmod p$.

Открытым ключом являются y , g и p . Закрытым ключом является x .

Чтобы подписать m , являющееся хэш-значением некоторого сообщения M , сначала выбирается секретное случайное число k , взаимно простое с $p-1$. Затем вычисляется $a = g^k \bmod p$.

Из соотношения $m = (x \cdot a + k \cdot b) \bmod (p-1)$ определяется b . Выполнив преобразования, получим $b = k^{-1} \cdot (m - x \cdot a) \bmod (p-1)$, где k^{-1} определяется из соотношения $k^{-1} \cdot k \equiv 1 \pmod{p-1}$.

В результате подписью будет пара (a, b) . Для проверки подписи нужно убедиться, что $y^a \cdot a^b \bmod p = g^m \bmod p$.

Пример.

Пусть $p = 11$, $g = 2$, $x = 8$. Тогда $y = 2^8 \bmod 11 = 3$, $m = 5$.

Выбираем $k = 9$. Тогда $a = 2^9 \bmod 11 = 6$.

Из соотношения $9 \cdot k^{-1} \equiv 1 \pmod{10}$ находим обратный элемент k^{-1} , применяя расширенный алгоритм Евклида (см. далее): $k^{-1} = 9$.

$$b = k^{-1} \cdot (m - x \cdot a) \bmod (p - 1) = 9 \cdot (-43) \bmod 10 = 3.$$

Подписью хэш-значения $m = 5$ является пара $(a, b) = (6, 3)$.

Проверка $3^6 \cdot 6^3 \bmod 11 = 2^5 \bmod 11 = 10$.

Нахождение обратного элемента с помощью расширенного алгоритма Евклида:

Пусть нужно найти элемент d^{-1} такой, что $d \cdot d^{-1} \equiv 1 \pmod{f}$.

Пусть $x = (1, 0, f)$, $y = (0, 1, d)$. В цикле выполняются следующие действия:

1. Если $y_3 = 0$, то не существует элемента, обратного к d по модулю f .
2. Если $y_3 = 1$, то $d^{-1} = y_2$.
3. Иначе выполняются следующие преобразования, после которых выполняется переход на шаг 1:

$$q = \left\lfloor \frac{x_3}{y_3} \right\rfloor,$$
$$t = x - q \cdot y, \quad x = y, \quad y = t.$$

Пример нахождения обратного элемента:

$$\begin{aligned} d &= 9, & f &= 10. \\ x &= (1, 0, 10), & y &= (0, 1, 9). \\ q &= 1. \\ t &= (1, 0, 10) - 1 \cdot (0, 1, 9) = (1, -1, 1). \\ x &= (0, 1, 9), & y &= (1, -1, 1). \\ y_3 &= 1, \Rightarrow d^{-1} = y_2 = -1 + 1 \cdot 10 = 9. \end{aligned}$$

Задание

1. Реализовать приложение, позволяющее вычислять и проверять ЭЦП, сформированную по алгоритмам RSA и Эль-Гамала.

2. С помощью реализованного приложения выполнить следующие задания:
 - 2.1. Протестировать правильность работы разработанного приложения.
 - 2.2. Для заданных в варианте открытых ключей пользователя проверить подлинность подписанных по алгоритму RSA хэш-значений m некоторых сообщений M .
 - 2.3. Абоненты некоторой сети применяют подпись Эль-Гамала с известными общими параметрами p и g . Для указанных в варианте секретных параметров абонентов найти открытый ключ и построить подпись для хэш-значения m некоторого сообщения M . Проверить правильность подписи

Варианты

Для построения подписи Эль-Гамала следует использовать открытые параметры $p = 23$, $g = 5$.

Вариант	ЭЦП по алгоритму RSA		ЭЦП по алгоритму Эль-Гамала	
	Открытые ключи	Проверяемые сообщения $\langle m, s \rangle$, где m – хэш-значение сообщения M	Секретные параметры	m – хэш сообщения M
1	$n = 55, e = 3$	$\langle 7, 28 \rangle, \langle 22, 15 \rangle, \langle 16, 36 \rangle$	$x = 11, k = 3$	$m = 15$
2	$n = 65, e = 5$	$\langle 10, 30 \rangle, \langle 6, 42 \rangle, \langle 6, 41 \rangle$	$x = 10, k = 15$	$m = 5$
3	$n = 77, e = 7$	$\langle 13, 41 \rangle, \langle 11, 28 \rangle, \langle 5, 26 \rangle$	$x = 3, k = 13$	$m = 8$
4	$n = 91, e = 5$	$\langle 15, 71 \rangle, \langle 11, 46 \rangle, \langle 16, 74 \rangle$	$x = 18, k = 7$	$m = 16$
5	$n = 33, e = 3$	$\langle 17, 8 \rangle, \langle 10, 14 \rangle, \langle 24, 18 \rangle$	$x = 9, k = 19$	$m = 3$
6	$n = 143, e = 37$	$\langle 46, 85 \rangle, \langle 16, 74 \rangle, \langle 129, 116 \rangle$	$x = 19, k = 5$	$m = 11$
7	$n = 221, e = 43$	$\langle 59, 19 \rangle, \langle 79, 164 \rangle, \langle 58, 20 \rangle$	$x = 14, k = 17$	$m = 14$
8	$n = 85, e = 15$	$\langle 24, 39 \rangle, \langle 39, 51 \rangle, \langle 83, 42 \rangle$	$x = 6, k = 13$	$m = 9$
9	$n = 187, e = 77$	$\langle 139, 90 \rangle, \langle 62, 163 \rangle, \langle 95, 57 \rangle$	$x = 4, k = 3$	$m = 4$
10	$n = 221, e = 79$	$\langle 207, 142 \rangle, \langle 112, 9 \rangle, \langle 82, 147 \rangle$	$x = 15, k = 15$	$m = 17$
11	$n = 57, e = 31$	$\langle 25, 28 \rangle, \langle 12, 42 \rangle, \langle 48, 15 \rangle$	$x = 12, k = 13$	$m = 18$
12	$n = 133, e = 41$	$\langle 52, 89 \rangle, \langle 82, 120 \rangle, \langle 67, 128 \rangle$	$x = 7, k = 7$	$m = 12$
13	$n = 209, e = 67$	$\langle 49, 125 \rangle, \langle 105, 17 \rangle, \langle 136, 97 \rangle$	$x = 13, k = 19$	$m = 20$
14	$n = 247, e = 71$	$\langle 249, 124 \rangle, \langle 95, 214 \rangle, \langle 173, 10 \rangle$	$x = 17, k = 5$	$m = 7$
15	$n = 323, e = 79$	$\langle 312, 122 \rangle, \langle 142, 29 \rangle, \langle 229, 134 \rangle$	$x = 8, k = 17$	$m = 13$

Контрольные вопросы:

1. Что такое хэш-функция, для чего она используется? В чём заключается устойчивость к столкновениям?

2. Как обмануть подписчика, если требование устойчивости к столкновению не выполняется?
3. Схемы хэширования с длиной хэш-значения, равной длине блока.
4. Схемы хэширования с длиной хэш-значения, равной удвоенной длине блока.
5. Для чего нужна цифровая подпись? Основные свойства цифровой подписи.
6. Какие схемы цифровой подписи существуют? Какая схема самая распространенная и почему?
7. Как осуществляется подпись RSA? В чем отличие подписи RSA от алгоритма шифрования RSA?
8. Как осуществляются подпись и проверка на подлинность подписи по алгоритму Эль-Гамала?