Протоколы безопасности
часть 1

А.М. Миронов

Излагаются основные криптографические примитивы, используемые в протоколах безопасности (симметричные и асимметричные системы шифрования, хэш-функции, схемы разделения секрета), протоколы аутентификации, и алгоритмы цифровой подписи.
Ключевые слова: протоколы, безопасность, криптография, хэш-функции, аутентификация, цифровая подпись

1. Введение

Данная статья является первой частью обзорного текста по **протоколам безопасности** (**security protocols**), называемых также **криптографическими протоколами**. В ней изложены основные криптографические примитивы, используемые в протоколах безопасности (симметричные и асимметричные системы шифрования, хэш-функции, схемы разделения секрета), протоколы аутентификации, и алгоритмы цифровой подписи. Во второй части будут изложены протоколы распределения криптографических ключей, протоколы голосования, протоколы электронных платежей, и некоторые другие протоколы. Более подробное изложение основных протоколов безопасности можно найти в текстах [1]-[5].

1.1. Понятие протокола безопасности

Протокол безопасности (ПБ), называемый также просто **протоколом** – это распределённый алгоритм, определяющий порядок обмена сообщениями между несколькими **агентами**, целью которого является обеспечение безопасности передачи, обработки и хранения информации в небезопасной среде. В качестве агентов, принимающих участие в работе протокола (их называют также **участниками** этого протокола) могут выступать, например, люди, компьютерные программы, вычислительные комплексы, базы данных, сети связи, банковские карточки, и т.д.

Действия, выполняемые участниками протокола, имеют следующий вид: **посылка сообщения** другому участнику (или группе участников), **приём сообщения** от другого участника, **внутренние действия** (проверка логических условий, обновление значений переменных).

Свойства безопасности, которые должен обеспечивать ПБ, могут иметь, например, следующий вид:

- **целостность** передаваемых сообщений, которая заключается в том, что всякое изменение сообщений в процессе их передачи будет обнаружено в ходе выполнения протокола,

- **секретность** передаваемых сообщений, которая заключается в отсутствии неавторизованной утечки информации в процессе работы протокола.

В последующих пунктах этого параграфа мы излагаем основные концепции и конструкции, используемые при построении ПБ.

1.2. Шифрование сообщений

1.2.1. Понятие шифрования и дешифрования

Некоторые из сообщений, пересылаемых участниками ПБ, могут быть зашифрованными. Шифрование сообщений делается для того, чтобы противник, которому станут доступны пересылаемые сообщения, не смог ознакомиться с их содержанием.

Шифрование сообщения m представляет собой применение некоторого алгоритма $Encrypt$ (называемого **алгоритмом шифрования**) к паре (k, m), где k – битовая строка, называемая **ключом шифрования** (или просто **ключом**). Мы будем обозначать результат применения алгоритма $Encrypt$ к паре (k, m) записью $k(m)$ и называть её **шифртекстом (ШТ)** сообщения m на ключе k.

Для того, чтобы извлечь из ШТ $k(m)$ исходное сообщение m, должен быть задан алгоритм $Decrypt$, называемый **алгоритмом дешифрования**, и обладающий следующими свойствами:

- алгоритм $Decrypt$ получает на вход пару вида (d, u), где d – битовая строка, называемая **ключом дешифрования**, и u – дешифруемое сообщение, результат применения $Decrypt$ к паре (d, u) обозначается записью $d(u)$,

- каждому ключу шифрования k должен соответствовать ключ дешифрования d_k, такой, что для каждого сообщения m верно равенство $d_k(k(m)) = m$.

1.2.2. Системы шифрования

Система шифрования (СШ) представляет собой набор следующих данных:

- пара алгоритмов *Encrypt* и *Decrypt* шифрования и дешифрования соответственно, и

- набор ограничений, которым должны удовлетворять ключи шифрования и дешифрования, а также шифруемые сообщения.

Системы шифрования принято подразделять на два класса: симметричные и асимметричные.

1) В **симметричных** СШ **(ССШ)** каждый ключ шифрования k совпадает с соответствующим ему ключом дешифрования d_k. Как правило, алгоритмы шифрования и дешифрования в ССШ тоже совпадают.

   Если ССШ используется для связи между несколькими агентами, то ключ шифрования этой ССШ, который используется в текущий момент, обозначается записью $k_{a_1 \ldots a_n}$, где $a_1, \ldots, a_n$ – список всех агентов, которым известен этот ключ. Если после использования этого ключа агенты $a_1, \ldots, a_n$ перейдут на новый ключ, то этот новый ключ будет обозначаться записью $k'_{a_1 \ldots a_n}$.

2) В **асимметричных** СШ **(АСШ)** ключи шифрования могут отличаться от соответствующих им ключей дешифрования, причем

   - ключи шифрования и дешифрования в АСШ, как правило, связаны с конкретными агентами, мы будем обозначать эти ключи записями a^+ и a^- соответственно, где a – идентификатор агента, с которым связаны эти ключи, и
   - ключ a^+ является **открытым** (т.е. известен всем агентам), в то время как ключ a^- **закрыт**: он не должен быть известен никому кроме агента a.

Некоторые АСШ обладают следующим свойством: для каждого сообщения m верно равенство $a^+(a^-(m)) = m$. Такие АСШ можно использовать для **аутентификации** (т.е. проверки подлинности) агентов: если какой-либо агент a, использующий эту АСШ, хочет доказать, что он действительно является тем агентом, имя которого – a, то он может сделать это, если для любого предъявляемого ему сообщения m он сможет вычислить сообщение $m' \stackrel{\text{def}}{=} a^-(m)$. Подлинность a будет доказана, если выполнено условие

$$a^+(m') = m. \tag{1}$$

Данный метод доказательства подлинности обосновывается предположением о том, что

- без знания закрытого ключа a^- создать сообщение m', удовлетворяющее условию (1), невозможно, и
- никто, кроме агента a, не знает ключ a^-.

1.3. Формальные описания и примеры протоколов

1.3.1. Понятие формального описания протокола

Одним из простейших видов формального описания ПБ является список P записей вида

$$a \to b : [\![\varphi]\!]\, m, \tag{2}$$

$$a \to \{b_1, \ldots, b_n\} : [\![\varphi]\!]\, m, \tag{3}$$

или

$$a : [\![\varphi]\!] \text{ внутреннее действие}, \tag{4}$$

объединенных фигурной скобкой слева, где a, b, $b_1, \ldots, b_n$ – имена агентов, φ – условие, m – выражение, значением которого является сообщение (**сообщением** мы называем любой объект, который один из участников ПБ передаёт другому в процессе работы ПБ, этот объект может быть как символьной строкой, так и набором банкнот, товаром, и т.п.). Записи (2), (3) и (4) изображают действия, которые должны выполнять агенты в процессе работы ПБ.

Действия, входящие в список P, выполняются последовательно, их выполнение происходит следующим образом:

- (2) и (3) выполняется путем проверки φ, и

– если φ выполнено, то a посылает m агенту b (в случае (2)), или агентам $b_1, \ldots, b_n$ (в случае (3)),

– если φ не выполнено, то это действие пропускается, и выполняется следующее по списку действие,

- (4) выполняется аналогично, только в данном случае происходит не посылка сообщения от a, а вычисления и изменение значений переменных агента a.

Если φ выполнено всегда, то компонента $[\![\varphi]\!]$ в записи действий опускается.

Если текущее исполняемое действие не относится к какому-либо из участников ПБ, то этот участник не функционирует в момент исполнения этого действия.

В формальных описаниях протоколов мы будем использовать следующее соглашение: запись вида $x := e$, где x – переменная, и e – выражение, обозначает переменную x, значение которой равно значению выражения e.

1.3.2. Пример формального описания протокола

В этом пункте мы рассмотрим пример формального описания протокола, решающего задачу продажи компьютера агента a агенту b. Мы предполагаем, что у a есть компьютер, а у b есть деньги, и b хочет на эти деньги купить у a компьютер. Агенты a и b не доверяют друг другу, поэтому протоколы продажи компьютера, имеющие вид

$$\begin{cases} a \to b: \text{ компьютер} \\ b \to a: \text{ деньги} \end{cases} \quad \text{или} \quad \begin{cases} b \to a: \text{ деньги} \\ a \to b: \text{ компьютер} \end{cases}$$

для них неприемлемы: каждый из них не верит, что если он выполнит первое действие в первом или втором протоколе, то его коллега обязательно выполнит второе действие.

Одним из возможных решений задачи продажи компьютера агента a агенту b может быть протокол, в котором, помимо a и b, принимает участие доверенный посредник I. Данный протокол может иметь, например,

следующий вид:

$$\begin{cases} a \to I: \ \text{компьютер} \\ b \to a: \ \text{деньги} \\ a \to I: \begin{pmatrix} \text{подтверждение или опровержение} \\ \text{того, что полученная от } b \text{ сумма} \\ \text{соответствует стоимости компьютера} \end{pmatrix} \\ I \to b: \ [\![\text{от } a \text{ поступило подтверждение}]\!] \ \text{компьютер} \\ I \to a: \ [\![\text{от } a \text{ поступило опровержение}]\!] \ \text{компьютер} \end{cases}$$

a и b могут считать данный протокол приемлемым, например, по следующим причинам:

- a верит, что до окончания проверки денег I не передаст компьютер агенту b, и I вернёт компьютер a, если b передаст a недостаточную сумму,
- b верит, что пока a не пошлёт I подтверждение, компьютер будет находиться у I, и сразу после того, как a пошлёт I подтверждение, I передаст b компьютер.

Однако данный протокол некорректно работает в том случае, когда какой-либо из агентов ведёт себя нечестно (например, b посылает a правильную сумму, но a посылает опровержение, получает обратно свой компьютер, и не отдает b полученные от него деньги).

1.4. Уязвимости протоколов

1.4.1. Понятие уязвимости протокола

Нарушения свойств безопасности в процессе работы ПБ могут происходить по причине противодействия со стороны агентов, называемых **противниками**.

Противники подразделяются на следующие два класса:

- **пассивные противники**, они могут перехватывать сообщения, пересылаемые участниками ПБ, и анализировать их,
- **активные противники**, они могут делать то же, что и пассивные противники, а также модифицировать или удалять перехваченные сообщения, генерировать новые сообщения и посылать их участникам ПБ вместо перехваченных, выдавать себя за участников ПБ.

Также нарушения свойств безопасности ПБ возможны из-за действий участников ПБ, которые нарушают (умышленно или неумышленно) предписанные протоколом правила взаимодействия с другими участниками этого ПБ.

Возможности нарушения свойств безопасности в процессе работы ПБ называют **уязвимостями** этого ПБ.

При решении задач анализа уязвимостей ПБ используются следующие предположения о противнике:

- противник активен и полностью знает ПБ,
- противник не может извлечь из каждого перехваченного ШТ $k(m)$ сообщения m, если он не знает k.

1.4.2. Пример уязвимости протокола

В этом пункте мы рассмотрим пример уязвимости ПБ, который использовался много лет в банковских транзакциях. Данный протокол очень прост, он состоит всего из трёх действий, и сначала его правильность не вызывала сомнений. Однако после 15 лет его использования выяснилось, что он содержит уязвимость (которая была обнаружена автоматической системой верификации). Этот ПБ называется **протоколом Нидхема--Шредера** (Needham-Schroeder, 1979), мы будем обозначать его записью NS. Целью данного ПБ является взаимная аутентификация агентов a и b.

Предполагается, что a и b используют общую АСШ.

Протокол NS имеет следующий вид:

$$\begin{cases} a \to b : b^+(a, r_a) \\ b \to a : a^+(r_a, r_b) \\ a \to b : b^+(r_b) \end{cases} \tag{5}$$

где r_a и r_b – **уникальные значения**, называемые в литературе по протоколам безопасности **нонсами** (**nonces**, что является сокращением от **number used once**), каждое из которых представляет собой большую (несколько сотен битов) псевдослучайную строку, сгенерированную агентом a и b соответственно. Мы предполагаем что все нонсы, сгенерированные в различных сеансах выполнения любого ПБ, являются различными и отличаются от других значений. Нонсы обозначаются символом r (возможно, с индексами).

Действия, входящие в ПБ NS, имеют следующий смысл.

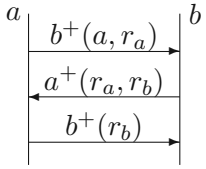
- Первое действие заключается в пересылке от a к b ШТ $b^+(a, r_a)$, получаемого шифрованием на ключе b^+ пары, состоящей из имени агента a и нонса r_a, который используется для того, чтобы дать возможность b доказать свою подлинность путем извлечения данного нонса из полученного ШТ.

- Второе действие заключается в пересылке от b к a ШТ $a^+(r_a, r_b)$, получаемого шифрованием на ключе a^+ пары, состоящей из нонса r_a, который b извлёк из полученного ШТ, и нонса r_b, сгенерированного агентом b. После его получения a убеждается в подлинности b, т.к. никто кроме b не может извлечь r_a. Нонс r_b предназначен для того, чтобы дать возможность a тоже доказать b свою подлинность.

- Третье действие заключается в пересылке от a к b ШТ $b^+(r_b)$. После получения r_b агент b убеждается в подлинности агента a.

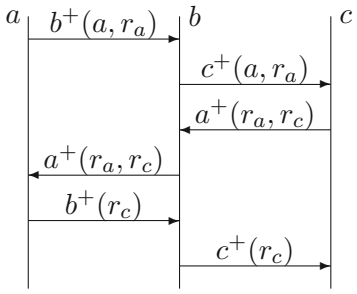
Таким образом, после успешного завершения данного протокола a и b будут убеждены в подлинности друг друга.

Для формальной записи ПБ используется также нотация в виде диаграмм. Каждое действие в ПБ, связанное с пересылкой сообщений, изображается в диаграмме горизонтальной стрелкой, помеченной пересылаемым сообщением. Диаграмма, соответствующая протоколу (5), имеет вид



Одна из уязвимостей данного ПБ связана с тем, что агент b может использовать свой статус участника ПБ NS для того, чтобы во взаимодействии с другими агентами выдавать себя за агента a. Использование агентом b своего статуса участника ПБ NS для совершения мошенниче-

ских действий можно изобразить следующей диаграммой:



Данную диаграмму можно рассматривать как объединение двух диаграмм, изображающих выполнение двух сеансов выполнения ПБ NS: в первом сеансе участвуют агенты a и b, а во втором – агенты b и c. Действия a, b и c во время выполнения этих сеансов выглядят следующим образом.

- После того, как a и b выполнят первое действие первого сеанса, агент b создает с использованием полученного нонса r_a ШТ $c^+(a, r_a)$, и посылает этот ШТ агенту c в качестве своего первого действия во втором сеансе ПБ NS (выдавая себя за a).

- Получив этот ШТ и дешифруя его, агент c приходит к выводу, что этот ШТ был послан агентом a. Согласно протоколу NS, агент c генерирует нонс r_c и посылает ответный ШТ $a^+(r_a, r_c)$.

- b пересылает полученный от c ШТ $a^+(r_a, r_c)$ агенту a.

- a рассматривает полученный от b ШТ как ответ, который b должен послать ему в соответствии со вторым шагом ПБ NS, т.е. a полагает, что извлечённый из полученного ШТ нонс r_c был сгенерирован агентом b.

- Согласно третьему шагу NS, a посылает b ШТ $b^+(r_c)$.

- b извлекает из полученного ШТ нонс r_c, и посылает агенту c ШТ $c^+(r_c)$.

После этого c верит, что тот агент, с которым он выполнял этот сеанс ПБ NS, является агентом a.

1.5. Другие примеры протоколов

1.5.1. Обедающие криптографы

Рассмотрим следующую задачу.

За круглым столом сидят три криптографа и обедают. После того, как они пообедали и хотят заплатить, официант сообщает им, что их обед уже оплачен, но не уточняет, кто именно платил. Возможен один из двух вариантов: обед оплатил один из криптографов, или обед оплатила ФСБ.

Криптографы хотят выяснить, какой именно из вариантов имеет место, причём, если имеет место первый вариант, то те криптографы, которые не платили, не должны узнать, кто же конкретно оплатил их обед.

Для решения этой задачи предлагается следующий ПБ.

Поскольку участники сидят за круглым столом, то каждая пара соседей может подбрасывать монету между собой, так, чтобы результат был известен только им двоим. Подбрасывание монеты двумя участниками можно рассматривать как получение ими сообщения, состоящего из одного случайным образом порождённого бита, от доверенного посредника. После того, как все три пары подбросили монету, каждый участник знает результаты двух подбрасываний (решка или орёл). Эти результаты могут быть

- либо одинаковыми (т.е. оба раза была решка, или оба раза был орёл),
- либо разными (т.е. при одном подбрасывании была решка, а при другом - орёл).

Каждый участник говорит другим “одинаково” или “по разному”, причём тот, кто заплатил, говорит противоположное (т.е. если надо сказать “одинаково” то он говорит “по-разному”, и наоборот). Если число ответов “по разному” чётно, то это значит, что обед оплатила ФСБ, иначе - один из них.

Одна из уязвимостей этого протокола заключается в отсутствии контроля честности участников.

1.5.2. Протокол с подтверждением приёма

Рассмотрим следующую ситуацию: агенты a и b используют общую АСШ для секретного обмена сообщениями, и для контроля правильности пере-

дачи каждое получаемое сообщение отсылается назад отправителю (чтобы отправитель был уверен, что сообщение получено в неискажённом виде), согласно следующему протоколу:

$$\begin{cases} a \to b : b^+a^-(m) \\ b \to a : a^+b^-(m) \end{cases}$$

(из полученного ШТ извлекается сообщение m и возвращается отправителю в зашифрованном виде в качестве подтверждения приёма).

Данный протокол уязвим к следующей атаке: активный противник e перехватывает первое сообщение (обозначим его m') и посылает его b (от своего имени). Согласно протоколу, b посылает e ответное сообщение $e^+b^-e^+b^-(m')$ (т.е. $e^+b^-e^+a^-(m)$), из которого e сможет извлечь m.

1.5.3. Вычисление суммы

Рассмотрим следующую задачу: агенты $a_1, \ldots, a_n$ имеют числа $x_1, \ldots, x_n$ соответственно. Они хотят вычислить $\sum\limits_{i=1}^{n} x_i$, причём каждый агент a_i не хочет разглашать своё число x_i. Один из протоколов для решения данной задачи имеет следующий вид:

$$\begin{cases} a_1 \to a_2 : {a_2}^+(x_1 + r), \text{ где } r \text{ – случайное целое число} \\ a_2 \to a_3 : {a_3}^+(x_2 + x_1 + r) \\ \ldots \\ a_n \to a_1 : {a_1}^+(x_n + \ldots + x_1 + r) \\ a_1 \text{ вычитает } r \text{ из полученного результата} \end{cases}$$

Недостаток этого протокола – нет контроля честности участников.

1.5.4. Сравнение двух чисел

Излагаемый ниже протокол предназначен для решения следующей задачи: агенты a и b имеют числа x и y соответственно (предполагается, что $x, y \in \{1, \ldots, 100\}$), они хотят узнать без разглашения чисел x и y, верно ли, что $x \leq y$.

Один из ПБ для решения этой задачи имеет следующий вид: (предполагается, что a и b используют общую АСШ)

$$\begin{cases} a \to b : z - x, \text{ где } z = b^{+}(r),\ r \text{ – случайное целое число} \\ b \text{ вычисляет } \{z_i := b^{-}(z - x + i) \mid i = 1, \ldots, 100\}, \\ \quad \text{и проверяет условие } \forall\, i \neq j \quad |z_i - z_j| \geq 2 \\ \quad \text{(если это условие не вып., то } b \text{ просит } a \text{ начать} \\ \quad \text{выполнение протокола заново и выбрать новое } r) \\ b \to a : \ z_1, \ldots, z_y, z_{y+1} + 1, \ldots, z_{100} + 1 \\ a \to b : \text{ответ} = (x \leq y), \text{ если } r = x\text{-й член этой посл–сти.} \end{cases}$$

Данный протокол тоже содержит уязвимости.

2. Необходимые математические сведения

2.1. Кольцо вычетов по модулю n

Ниже символ $\mathbf{Z}$ обозначает множество целых чисел.

Пусть n – положительное целое число. Обозначим записью $\mathbf{Z}_n$ множество $\{0, 1, \ldots, n-1\}$. $\forall\, a \in \mathbf{Z}$ существует единственное число $r \in \mathbf{Z}_n$, такое, что $\exists\, q \in \mathbf{Z} : a = nq + r$. Данное число называется **остатком** от деления a на n и обозначается записью $(a)_n$.

$\forall\, a, b \in \mathbf{Z}$ запись $a \underset{n}{=} b$ означает, что $(a)_n = (b)_n$.

Множество $\mathbf{Z}_n$ является коммутативным кольцом, операции сложения и умножения на котором обозначаются записями $\underset{n}{+}$ и $\underset{n}{\cdot}$ соответственно, и определяются следующим образом: $\forall\, a, b \in \mathbf{Z}_n$, $a \underset{n}{+} b \stackrel{\text{def}}{=} (a + b)_n$, $a \underset{n}{\cdot} b \stackrel{\text{def}}{=} (ab)_n$. $\forall\, a \in \mathbf{Z}_n$ запись $\underset{n}{-} a$ обозначает число $n - a$ (если $a \neq 0$) и 0, если $a = 0$. Ниже числа $a \underset{n}{+} b$, $a \underset{n}{+} (\underset{n}{-} b)$ и $a \underset{n}{\cdot} b$ могут обозначаться записями $a + b$, $a - b$ и ab соответственно.

Подмножество множества $\mathbf{Z}_n$, состоящее из чисел, взаимно простых с n, является группой относительно операции умножения на $\mathbf{Z}_n$, её нейтральным элементом является число 1. Данная группа обозначается записью $\mathbf{Z}_n^*$. Число элементов в $\mathbf{Z}_n^*$ называется **функцией Эйлера** числа n, оно обозначается записью $\varphi(n)$ и равно $n(1 - \frac{1}{p_1}) \ldots (1 - \frac{1}{p_k})$, где

$p_1, \ldots, p_k$ – различные простые делители n. Элемент $g \in \mathbf{Z}_n^*$ называется **примитивным**, если $\mathbf{Z}_n^* = \{g^i \mid i \geq 1\}$.

Элементы $\mathbf{Z}_2$ $(= \{0,1\})$ называются **битами**. Для каждого $n \geq 1$ числа из $\mathbf{Z}_{2^n}$ в двоичной записи содержат не более n бит, в некоторых случаях мы будем отождествлять их с битовыми последовательностями длины n, т.е. с элементами $\mathbf{Z}_2^n$. Если $n = kl$, то элементы $\mathbf{Z}_{2^n}$ можно отождествлять с последовательностями l битовых блоков длины k, т.е. с элементами $\mathbf{Z}_{2^k}^l$.

Для каждого положительного $n \in \mathbf{Z}$ запись $|n|$ обозначает количество битов в двоичной записи n. $\forall\, m, n \in \mathbf{Z}$ запись $m|n$ означает, что m является делителем n. Для каждого множества M запись $|M|$ обозначает число элементов множества M.

$\forall\, g \in \mathbf{Z}_n^*$ запись $ord(g)$ обозначает порядок элемента g в группе $\mathbf{Z}_n^*$, т.е. наименьшее целое $n > 0$, такое, что $g^n \underset{\varphi(n)}{=} 1$.

Элемент g группы $\mathbf{Z}_n^*$ называется **порождающим**, если $\mathbf{Z}_n^* = \{g^i \mid i \geq 0\}$ (т.е. $ord(g) = |\mathbf{Z}_n^*|$).

$\forall\, g \in \mathbf{Z}_n^*$ запись g^{-1} обозначает элемент группы $\mathbf{Z}_n^*$, являющийся обратным к g. $\forall\, g \in \mathbf{Z}_n^*, \forall\, m \in \mathbf{Z}$, $m > 0$, запись g^{-m} обозначает элемент $(g^{-1})^m$ группы $\mathbf{Z}_n^*$.

Если в какое-либо выражение входит число, изначально выбранное как элемент множества $\mathbf{Z}_n$ или $\mathbf{Z}_n^*$, и это число является аргументом операций сложения, вычитания, умножения, или основанием в операции возведения в степень, то по умолчанию предполагается, что операции в этом выражении выполняются как операции в $\mathbf{Z}_n$.

2.2. Автоматы Мура

Автомат Мура (называемый ниже просто **автоматом**) – это набор

$$M = (X, S, Y, s^0, \delta, \lambda) \tag{6}$$

где X, S и Y – множества, элементы которых называются соответственно **входными сигналами**, **состояниями**, и **выходными сигналами**, $s^0 \in S$ – **начальное состояние**, $\delta : S \times X \to S$ – **функция перехода**, $\lambda : S \to Y$ – **функция выхода**.

Автомат является моделью дискретной динамической системы, работа которой заключается в изменении состояний под воздействием входных сигналов, поступающих на её вход, и выдаче в каждый момент времени $t = 0, 1, 2, \ldots$ некоторого выходного сигнала.

В начальный момент времени ($t = 0$) автомат находится в состоянии s^0. В каждый момент времени $t = 0, 1, 2, \ldots$ автомат получает входной сигнал $x(t) \in X$, выдаёт выходной сигнал $y(t) \stackrel{\text{def}}{=} \lambda(s(t)) \in Y$, и в следующий момент времени $(t+1)$ переходит в состояние $s(t+1) \stackrel{\text{def}}{=} \delta(s(t), x(t))$.

Автомат называется **автономным**, если множество его входных сигналов состоит из одного элемента. Если M – автономный автомат с множеством состояний S, то можно считать, что его функция перехода δ имеет вид $S \to S$.

Для каждого множества X и каждого $n \geq 0$ запись X^n обозначает множество последовательностей длины n с компонентами из X (если $n = 0$, то X^n состоит из одной пустой последовательности, обозначаемой символом ε). Запись X^* обозначает множество $\bigcup_{n \geq 0} X^n$.

Пусть M – автомат вида (6). $\forall\, (s, u) \in S \times X^*$ запись su обозначает состояние, в которое перейдет автомат M из состояния s после поступления на его вход последовательности входных сигналов u, т.е. $s\varepsilon \stackrel{\text{def}}{=} s$, $sx \stackrel{\text{def}}{=} \delta(s, x)$, $sx_1 \ldots x_{n-1}x_n \stackrel{\text{def}}{=} (sx_1 \ldots x_{n-1})x_n$.

Автомат (6) называется **автоматом без выхода**, если $Y = S$, и $\lambda = id_S$. Если (6) – автомат без выхода, то будем обозначать его четверкой (X, S, s^0, δ).

3. Криптографические примитивы

Криптографические примитивы – это математические понятия и конструкции, используемые в качестве элементарных компонентов при построении ПБ. Криптографические примитивы предназначены для обеспечения различных свойств безопасности ПБ. К криптографическим примитивам относятся системы шифрования, хэш-функции, и некоторые другие понятия и конструкции.

3.1. Системы шифрования

3.1.1. Симметричные системы шифрования

Симметричные СШ в основном относятся к следующим двум классам: блочные и поточные.

В **блочных СШ** шифруемое сообщение разбивается на блоки одинакового размера, и каждый блок шифруется при помощи одного и того же алгоритма.

Шифрующие преобразования блоков заключаются в суперпозиции нескольких простых отображений, называемых **базовыми преобразованиями**.

Среди базовых преобразований блоков наибольшее распространение получили **преобразования Фейстеля**, которые заключаются в разделении обрабатываемого блока на левую и правую половины L и R, и преобразовании блока (L, R) в блок (L', R') (где L' и R' – левая и правая половины преобразованного блока) по следующему принципу:

$$L' := R, \quad R' := L \oplus f(R, k),$$

где $\oplus$ – побитовое сложение по модулю 2, и k – ключ.

Алгоритм шифрования реализуется несколькими итерациями преобразования Фейстеля, при этом очередная итерация использует в качестве входного блока результат предыдущей итерации.

Для дешифрования применяется обратное преобразование, которое вычисляется по той же схеме, как и исходное:

$$L = R' \oplus f(L', k), \quad R = L'.$$

В **поточных СШ** шифрование заключается в сложении по модулю 2 каждого бита открытого текста с соответствующим битом псевдослучайной последовательности, называемой **гаммой**. Дешифрование осуществляется по той же схеме, как и шифрование.

Для порождения гаммы используются **регистры сдвига с линейной обратной связью (РСЛОС)**. РСЛОС – это автономный автомат вида $(\{1\}, \mathbf{Z}_2^n, \mathbf{Z}_2, k, \delta, \lambda)$, функция переходов которого сопоставляет произвольному состоянию $(q_1, \ldots, q_n)$ состояние $(q_2, \ \ldots, \ q_n, \ \sum_{i=1}^{n} c_i q_i)$, где $c_1, \ldots, c_n$ – фиксированные элементы $\mathbf{Z}_2$.

Гамма, которую порождает РСЛОС, представляет собой последовательность его выходных сигналов (которые он выдает в моменты 0, 1, и т.д.). Для шифрования или дешифрования при помощи РСЛОС его начальное состояние полагается равным ключу.

Существуют и другие способы порождения гаммы:

- первый заключается в использовании двух РСЛОС: если гаммы, порожденные ими, имеют вид $a_0, a_1, \ldots$ и $b_0, b_1, \ldots$ соответственно, то результирующая гамма получается из гаммы $a_0, a_1, \ldots$ удалением её компонентов с такими номерами i, что $b_i = 0$,

- другой способ заключается в том, что вместо РСЛОС используются автономные автоматы с состояниями из R^n (где R – конечное кольцо, элементы R^n рассматриваются как вектор-столбцы длины n над R), отображение переходов которых переводит состояние $q \in R^n$ в состояние $Aq+b$, где A – матрица порядка n над R, $b \in R^n$.

3.1.2. Асимметричные системы шифрования

В асимметричных СШ каждый агент a использует пару ключей a^+, a^-, где

- a^+ – **открытый ключ**, он используется для шифрования, и должен быть известен всем агентам,
- a^- – **закрытый ключ**, он используется для дешифрования, и должен быть известен только агенту a.

Одним из примеров ассиметричных СШ является **СШ RSA**. Её название является аббревиатурой, связанной с фамилиями её создателей (Rivest, Shamir и Adleman). Криптографическая стойкость данной СШ (т.е. сложность нахождения по ШТ $k(m)$ сообщения m без знания ключа дешифрования) основывается на вычислительной сложности задачи разложения на множители больших целых чисел.

Для задания конкретной реализации СШ RSA агент a должен сгенерировать два больших (несколько сотен битов) простых числа p и q, примерно одинаковых по размеру, и таких, что НОД $(p-1, q-1)$ – небольшое число.

Ключи a^+ и a^- имеют следующий вид:

- $a^+ = \{n, e\}$, где $n = pq$, $e \underset{r}{\in} \mathbf{Z}^*_{\varphi(n)}$
  (запись вида $x \underset{r}{\in} X$ означает, что x – случайно и равновероятно выбранный элемент множества X),
- $a^- = \{d, p, q\}$, где $ed \underset{\varphi(n)}{=} 1$.

Для шифрования шифруемое сообщение разбивается на блоки, каждый из которых можно рассматривать как двоичную запись числа из $\mathbf{Z}_n$. Каждый из этих блоков шифруется отдельно. Шифрование и дешифрование определяются следующим образом: (все вычисления – в $\mathbf{Z}_n$)

$$\forall\, m \in \mathbf{Z}_n \quad a^+(m) \stackrel{\text{def}}{=} m^e, \;\; a^-(m) \stackrel{\text{def}}{=} m^d.$$

Нетрудно доказать, что СШ RSA обладает свойством

$$\forall\, m \in \mathbf{Z}_n \;\; a^-(a^+(m)) = m, \quad a^+(a^-(m)) = m.$$

Другим примером асимметричной СШ является **СШ Эль-Гамаля**. Её криптографическая стойкость основывается на вычислительной сложности задачи дискретного логарифмирования, т.е. задачи нахождения по паре $a, b \in \mathbf{Z}_p$ (где p – простое число) такого $x \in \mathbf{Z}$, что $a^x \underset{p}{=} b$.

Для задания конкретной реализации СШ Эль-Гамаля агент a должен сгенерировать большое простое число p, и примитивный элемент $g \in \mathbf{Z}_p^*$. Ключи a^+ и a^- имеют следующий вид: $a^- = x \underset{r}{\in} \mathbf{Z}_p^*$, $a^+ = \{p, g, y\}$, где $y \underset{p}{=} g^x$.

Как и в RSA, для шифрования шифруемое сообщение разбивается на блоки, каждый из которых можно рассматривать как двоичную запись числа из $\mathbf{Z}_p$. Каждый из этих блоков шифруется отдельно. Шифрование и дешифрование определяются следующим образом:

$$a^+(m) = (g^z,\ my^z), \qquad \text{где } z \underset{r}{\in} \mathbf{Z}_p^*, \quad a^-(u, v) = u^{-x}v.$$

3.2. Хэш-функции

3.2.1. Понятие хэш-функции

Хэш-функция (ХФ) - это функция вида $h : D \to \mathbf{Z}_2^n$ (где $D \subseteq \mathbf{Z}_2^*$), удовлетворяющая условиям:

- h – **односторонняя функция**, т.е. не существует быстрого алгоритма нахождения по заданному $y \in \mathbf{Z}_2^n$ такого $x \in D$, что $y = h(x)$,
- h **устойчива к коллизиям**, т.е. сложно найти различные $x_1, x_2 \in D$, такие, что $h(x_1) = h(x_2)$.

Алгоритм вычисления ХФ должен быть общеизвестен. ХФ применяются для контроля целостности данных, аутентификации источников данных, и многих других целей.

Ниже символ h (возможно с индексами) обозначает ХФ.

3.2.2. Пример построения хэш–функции

Каждый автомат без выхода M вида $(\mathbf{Z}_2^n, \mathbf{Z}_2^n, s^0, \delta)$ определяет функцию h_M, которая вычисляется следующим образом: $\forall\, x \in \mathbf{Z}_2^* \;\; x$ дополняется

до размера, кратного n, представляется в виде конкатенации $x_1 \dots x_k$, где $\forall\, i = 1, \dots, k\ x_i \in \mathbf{Z}_2^n$, и $h_M(x) \stackrel{\text{def}}{=} s^0 x_1 \dots x_k$.

h_M может быть ХФ, если $\forall\, s, x \in \mathbf{Z}_2^n \quad \delta(s, x) = s(x) \oplus x$, где $s(x)$ – ШТ, получаемый шифрованием x на ключе s. Если же $\delta(s, x)$ имеет вид $k(s \oplus x)$, где k – фиксированный ключ, то h_M скорее всего не является ХФ.

3.2.3. Стандарт хэш–функции SHS

Стандарт ХФ SHS (Secure Hash Standard) был разработан в 1993 г., он основан на алгоритме MD4 Ривеста. Определяемая ниже ХФ h, построенная по данному стандарту, преобразует битовые строки длины $\leq 2^{64}$ в битовые строки длины 160. Значение $h(m)$ вычисляется следующим образом.

Пусть m состоит из n бит. Сначала к m справа приписывается битовая строка вида $10 \dots 0 l_1 \dots l_{64}$, такая, что $l_1 \dots l_{64}$ – двоичная запись числа n, и размер получившейся строки u делится на 512 ($= 32{\cdot}16$). Представим u в виде конкатенации блоков $u_1 \dots u_k$, где длина каждого блока u_i равна 512 (т.е. u_i можно рассматривать как число из $\mathbf{Z}_{2^{32}}^{16}$ $(i = 1, \dots, k)$). Искомое значение $h(m)$ равно состоянию $s^0 u_1 \dots u_k$ автомата без выхода $(\mathbf{Z}_{2^{32}}^{16}, \mathbf{Z}_{2^{32}}^{5}, s^0, \delta)$, где $\delta(s, x) = s + g(s, x)$, и

- s и $g(s, x)$ рассматриваются как последовательности из пяти блоков, которые $\in \mathbf{Z}_{2^{32}}$, сложение выполняется поблочно, блоки складываются в $\mathbf{Z}_{2^{32}}$, и

- $g(s, x)$ – состояние, в которое перейдёт автомат без выхода $(\mathbf{Z}_{2^{32}} \times \mathbf{Z}_{2^{32}} \times \mathbf{Z}, \mathbf{Z}_{2^{32}}^{5}, s, \delta')$ после поступления на его вход последовательности $(v_0, z_0, 0) \ \dots\ (v_{79}, z_{79}, 79)$, которая удовлетворяет условиям:

$$
\begin{aligned}
&(v_0, \dots, v_{15}) = x,\\
&v_i = v_{i-3} \oplus v_{i-8} \oplus v_{i-14} \oplus v_{i-16} \quad (i = 16, \dots, 79),\\
&z_{20i} = z_{20i+1} = \dots = z_{20i+19} \quad (i = 0, 1, 2, 3),
\end{aligned}
$$

и $\delta'((a, b, c, d, e), (v, z, i)) =$
$= (T^5(a) + f(b, c, d, i) + e + v + z, a, T^{30}(b), c, d)$, сложение – в $\mathbf{Z}_{2^{32}}$,

T – циклический сдвиг влево на 1 бит, и

$$f(b,c,d,i) = \begin{cases} (b \wedge c) \vee (\neg b \wedge d) & (i = 0, \ldots, 19) \\ b \oplus c \oplus d & (i = 20, \ldots, 39, 60, \ldots, 79) \\ (b \wedge c) \vee (b \wedge d) \vee (c \wedge d) & (i = 40, \ldots, 59) \end{cases}$$

($\neg$, $\wedge$, $\vee$, $\oplus$ – побитовые операции над блоками).

3.3. Цифровая подпись

Цифровая подпись является криптграфическим примитивом, предназначенным для доказательства подлинности передаваемых сообщений и их отправителей.

Алгоритм вычисления цифровой подписи преобразует пару (m, a), где m – подписываемое сообщение, и a – имя агента, подписывающего это сообщение, в строку $\langle m \rangle_a^s$, называемую **цифровой подписью (ЦП)** сообщения m, созданной агентом a. При вычислении $\langle m \rangle_a^s$ используется **закрытый ключ ЦП** агента a, который обозначается записью s_a, и должен быть известен только агенту a. Тройка $(m, a, \langle m \rangle_a^s)$ обозначается записью $\langle m \rangle_a$, и называется **подписанным сообщением**.

ЦП должна обладать следующими свойствами:

- **возможность проверки подлинности ЦП**: существует открытый алгоритм позволяющий по тройке (m, a, u) проверить истинность равенства $u = \langle m \rangle_a^s$,
- **невозможность подделки ЦП**: задача вычисления $\langle m \rangle_a^s$ без знания s_a является труднорешаемой,
- **невозможность подмены**: задача нахождения по $\langle m \rangle_a^s$ такого $m' \neq m$, что $\langle m' \rangle_a^s = \langle m \rangle_a^s$, является труднорешаемой.

Если подписывающий агент a и проверяющий агент b используют одну и ту же АШС, $\langle m \rangle_a^s$ может иметь, например, один из следующих видов:

$$a^-(h(m)), \quad b^+a^-(h(m)), \quad a^-(a, a^-(h(m)), t)$$

где h –ХФ, t – метка времени, $s_a = a^-$.

3.4. Схемы разделения секрета

Еще одним классом криптографических примитивов являются схемы разделения секрета.

3.4.1. Понятие схемы разделения секрета

Схема разделения секрета (СРС) – это способ распределения секретной информации между несколькими агентами, используя которую, они могут вычислить некоторое заданное значение s (называемое **секретом**). Информация, которую при этом получает каждый агент, называется **долей** этого агента. Например, секрет $s \in \mathbf{Z}_2^l$ распределяется между n агентами, доли которых имеют вид $s_1 \underset{r}{\in} \mathbf{Z}_2^l$, ..., $s_{n-1} \underset{r}{\in} \mathbf{Z}_2^l$, $s_n = s \oplus s_1 \oplus \ldots \oplus s_{n-1}$. Значение s могут вычислить только все агенты совместно, а если доля хотя бы одного из агентов неизвестна, то все остальные агенты, даже открыв друг другу свои доли, не могут извлечь из них никакой информации о значении s.

3.4.2. Схема разделения секрета Шамира

Излагаемая в этом пункте СРС Шамира относится к числу (n, k)-пороговых СРС (где $1 < k \leq n$), т.е. таких СРС, в которых секретное значение s распределяется между n агентами таким образом, что

- любая совокупность из k агентов может, используя свои доли, вычислить s, и
- любая совокупность из $< k$ агентов не сможет извлечь из своих долей никакой информации об s.

СРС Шамира используется в том случае, когда секретное значение s является элементом некоторого поля P.

Для распределения секрета s среди агентов $a_1, \ldots, a_n$ выбираются многочлен $f \in P[x]$ вида $s + c_1 x + \ldots + c_{k-1} x^{k-1}$ ($c_{k-1} \neq 0$) и различные элементы $x_1, \ldots, x_n \in P \setminus \{0\}$.

$\forall\, i = 1, \ldots, n$ доля агента a_i имеет вид пары $(x_i, f(x_i))$.

Для вычисления секрета используется интерполяционная теорема Лагранжа: для каждого k–элементного подмножества $\{y_1, \ldots, y_k\} \subseteq \{x_1, \ldots, x_n\}$ верно равенство

$$f(x) = \sum_{i=1}^{k} f(y_i) \prod_{j \neq i} \frac{x - y_j}{y_i - y_j}$$

из которого следует, что $s = f(0) = \sum\limits_{i=1}^{k} f(y_i) \prod\limits_{j \neq i} \frac{y_j}{y_j - y_i}$.

Если доли вычисляются и распределяются агентом s, которому агенты $a_1, \ldots, a_n$ не доверяют, и хотят проверить правильность своих долей, то в том случае, когда $P = \mathbf{Z}_p$, это можно сделать следующим образом. Выбирается несекретный элемент $g \in P$, и s посылает всем агентам множество $\{d_i := g^{c_i} \mid i = 0, \ldots, k-1\}$ $(c_0 = s)$. Агент a_i проверяет правильность своей доли z_i путем проверки равенства

$$g^{z_i} = d_0 d_1^{x_i} \ldots d_{k-1}^{\binom{x_i}{k-1}}.$$

3.4.3. Схема Карнина-Грини-Хеллмана

Другим примером (n, k)–пороговой схемы является схема Карнина-Грини-Хеллмана. В ней секрет s тоже является элементом поля P. Для вычисления долей выбираются различные числа $r_0, \ldots, r_n \in P$, и вычисляется множество строк $\vec{v}_0, \ldots, \vec{v}_n$, где $\forall\, i = 0, \ldots, n \quad \vec{v}_i = (1\, r_i\, r_i^2\, \ldots\, r_i^{k-1})$. Обозначим символом $u^{\downarrow}$ столбец из P^k с компонентами $u_0, \ldots, u_{k-1}$, где $u_1 \underset{r}{\in} P, \ldots, u_{k-1} \underset{r}{\in} P$, $u_0 = s - \sum_{i=1}^{k} r_0^i u_i$ (т.е. $s = \vec{v}_0 u^{\downarrow}$).

$\forall\, i = 1, \ldots, n$ доля агента a_i имеет вид $\vec{v}_i u^{\downarrow}$. Каждые k долей порождают невырожденную систему из k линейных уравнений, неизвестными в которой являются $u_0, \ldots, u_{k-1}$.

3.4.4. Схема разделения секрета с двумя группами агентов

Излагаемая ниже СРС предназначена для решения следующей задачи: имеется две группы агентов $\{a_1, \ldots, a_{n_1}\}$ и $\{b_1, \ldots, b_{n_2}\}$, и для вычисления секрета s требуются доли любых k_1 агентов из первой группы, и любых k_2 агентов из второй группы.

Если s можно представить в виде ненулевого элемента некоторого поля P, то для решения этой задачи можно использовать СРС, аналогичную СРС Шамира: в данном случае для распределения секрета s выбираются

- пара многочленов $f_1, f_2 \in P[x]$ степени $k_1 - 1$ и $k_2 - 1$ соответственно, причём $s = f_1(0) f_2(0)$, и
- пара подмножеств $\{x_1, \ldots, x_{n_1}\}, \{y_1, \ldots, y_{n_2}\}$ множества $P \setminus \{0\}$.

$\forall\, i = 1, \ldots, n_1$ доля агента a_i имеет вид пары $(x_i, f_1(x_i))$, и $\forall\, j = 1, \ldots, n_2$ доля агента b_j имеет вид пары $(y_j, f_2(y_j))$.

4. Протоколы аутентификации

4.1. Понятие протокола аутентификации

Важный класс протоколов безопасности составляют **протоколы аутентификации (ПА)**. Они предназначены для решения задачи **аутентификации** (т.е. доказательства подлинности) агентов, ключей, сообщений, времени создания сообщений, сеансов связи, и т.д.

Мы будем рассматривать в этом параграфе только ПА агентов. Проблема аутентификации агентов представляет большую актуальность, например, в том случае, когда агенты выражают желание получить доступ к ресурсам, безопасность которых представляет повышенный интерес (банковские счета, секретные базы данных, государственные здания, и т.д.). Как правило, в ПА агентов

- принимают участие два обычных агента, которых мы будем обозначать символами a и b (в литературе по ПБ их обычно называют Алиса и Боб),
- а также может принимать участие доверенный посредник I, в этом случае мы будем предполагать, что a, b и I могут использовать общую ССШ.

Как правило, аутентификация агента a перед другим агентом b заключается в том, что a доказывает b знание некоторого секретного значения s, причём во многих случаях a должен убедить b что он знает s, не раскрывая s. Если после завершения этого доказательства у b не появляется новой информации о том, в каком диапазоне может содержаться значение s, то такое доказательство называется **доказательством с нулевым разглашением (Д0Р)**.

ПА агентов включают в себя следующие два класса:

- протоколы **односторонней аутентификации (ПОА)**, в которых только один из агентов (a) доказывает свою подлинность другому агенту (b), и
- протоколы **двусторонней аутентификации (ПДА)**, в которых оба агента a и b доказывают свою подлинность друг другу.

Некоторые ПА предназначены для одновременного решения двух задач: аутентификации агентов, и выработки ими (или передачи им от I) нового сеансового ключа ССШ, который эти агенты могут использовать

для организации сеанса шифрованной связи друг с другом после завершения работы ПА.

В ПА часто используются **метки времени (МВ)**, они обозначаются символом t (возможно, с индексами), и символ t (возможно, с индексами) обозначает только МВ. Если какой-либо агент получает сообщение, содержащее МВ, то проводит дополнительную проверку того, что эта МВ принадлежит заданному промежутку $[t_{min}, t_{max}]$.

4.2. Простейшие протоколы аутентификации

4.2.1. Простейшие протоколы односторонней аутентификации

- $\{a \to b : k_{ab}(b, t)$, в этом протоколе $s = k_{ab}$.
- $\{a \to b : \langle b, t \rangle_a$, в этом протоколе $s = s_a$.

4.2.2. Протоколы односторонней аутентификации с использованием паролей

Пароль - это строка, являющаяся общим секретом a и b.

Примеры простейших ПОА с использованием паролей:

- $\{a \to b : a, \text{пароль}$.
- $\{a \to b : a, r, h(\text{пароль}, r)$, этот ПОА обеспечивает защиту от перехвата паролей.

Как правило, в системах аутентификации с использованием паролей выполняется регулярное обновление паролей, которое может происходить по одной из следующих схем.

1) Агенты a и b имеют общий список паролей, и заранее договариваются о порядке смены паролей.

2) Сначала a и b используют пароль p_0. Каждый сеанс аутентификации агентов a и b имеет свой порядковый номер $i = 0, 1, \ldots$. На сеансе аутентификации с номером i используется пароль p_i. Пароли $p_1, \ldots$ генерируются агентом a. Агенты a и b используют алгоритм, который по каждому паролю p_i вырабатывает ключ ССШ k_i. i–й сеанс ПОА имеет вид $a \to b : p_i, k_i(p_{i+1})$.

3) Агенты a и b выбирают число n, представляющее собой максимальное количество сеансов аутентификации, которые они собираются выполнить.

   a генерирует последовательность паролей $p_0, \ldots, p_n$, где $p_0 = r$, $p_i = h(p_{i-1})$ $(i = 1, \ldots, n)$.

   b каким-либо образом получает p_n. $\forall\, i = 1, \ldots, n$ в i–м сеансе аутентификации a посылает b пароль p_{n-i}, и b проверяет равенство $h(p_{n-i}) = p_{n-i+1}$.

4.2.3. Простейшие протоколы двусторонней аутентификации

1) $\begin{cases} b \to a : r_b \\ a \to b : k_{ab}(b, r_a, r_b) \\ b \to a : k_{ab}(r_a, r_b) \end{cases}$

2) $\begin{cases} b \to a : r_b \\ a \to b : \langle b, r_a, r_b \rangle_a \\ b \to a : \langle a, r_a, r_b \rangle_b \end{cases}$

3) $\begin{cases} b \to a : r_b \\ a \to b : r_a, h(a, r_a, r_b) \\ b \to a : h(b, r_a) \end{cases}$

4.3. Вопросно-ответные протоколы односторонней аутентификации

Работа **вопросно-ответного ПОА** состоит из нескольких раундов, в каждом из которых a посылает b доказательство знания секрета s, и b либо принимает это доказательство, либо не принимает. Как правило, в каждом раунде b посылает a некоторый вопрос, и доказательство знания s представляет собой ответ на этот вопрос.

a проходит аутентификацию только в том случае, если в каждом из раундов b принимает доказательство, которое ему прислал a. Если в одном раунде вероятность того, что b пример ответ a, не превосходит p, то вероятность того, что a правильно ответит в d раундах, не превосходит p^d (поскольку все раунды независимы).

Для защиты от атаки с повторной передачей (**replay**) в вопросы и ответы можно включать нонсы и МВ.

4.3.1. Вопросно-ответные однораундовые протоколы односторонней аутентификации

1) $\begin{cases} b \to a : r \\ a \to b : k_{ab}(b, r) \end{cases} \quad (s = k_{ab})$

2) $\begin{cases} b \to a : r \\ a \to b : \langle b, r \rangle_a \end{cases} \quad (s = s_a)$

3) $\begin{cases} b \to a : r \\ a \to b : a, a^-(r) \end{cases} \quad (s = a^-)$

4) $\begin{cases} b \to a : \ h(r), b, a^+(a, r) \\ a \to b : r \end{cases} \quad (s = a^-).$

4.3.2. Протокол аутентификации Шнорра

Открытые параметры:

- простые числа p и q, где $|p| \geq 512$, $|q| \geq 140$, $q \mid p - 1$,
- элемент $g \in \mathbf{Z}_p^*$, такой, что $\mathrm{ord}(g) = q$.

Секретным значением является число $s \underset{r}{\in} \mathbf{Z}_q^*$, для проверки знания s используется открытое значение $v = g^{-s}$.

ПОА Шнорра имеет следующий вид:

$$\begin{cases} a \to b : x := g^z, \text{ где } z \underset{r}{\in} \mathbf{Z}_q^* \\ b \to a : e \underset{r}{\in} \mathbf{Z}_q^*, \text{ где } |e| < 100 \\ a \to b : y := z + se \\ b \text{ принимает ответ, если } x = g^y v^e \end{cases}$$

Если a не знает s, но хочет послать то значение y, которое примет b, он должен уметь вычислять для каждого возможного e значение $y = z + se$, что равносильно знанию $s = (y - z)e^{-1}$.

Можно доказать, что данный ПА является Д0Р.

4.3.3. Протокол Шаума

Открытые параметры:

- целое число $n > 0$, и
- порождающий элемент g группы $\mathbf{Z}_n^*$.

Секретным значением является число $s \underset{r}{\in} \mathbf{Z}_{\varphi(n)}$, для проверки знания s используется открытое значение $v = g^s$.

Протокол состоит из d раундов, имеющих вид

$$\begin{cases} a \to b : x := g^z, \text{ где } z \underset{r}{\in} \mathbf{Z}_{\varphi(n)} \\ b \to a : e \underset{r}{\in} \{0,1\} \\ a \to b : y := z + se \\ b \text{ принимает } y, \text{ если } g^y = xv^e \end{cases}$$

Если a не знает s, но хочет послать такой ответ y, который примет b, то он может смошенничать следующим образом: выбрать $e' \underset{r}{\in} \{0,1\}$, и послать в качестве x значение $g^z v^{-e'}$, а в качестве y – значение z. В этом случае b примет ответ, если будет верно равенство $g^z = g^z v^{-e'} v^e$, т.е. если e' совпадает со значением e, которое ему пришлет b. Таким образом, вероятность успеха в одном раунде при описанном выше мошенничестве равна $1/2$.

Модификации протокола Шаума:

1) Открытые параметры те же, что и в протоколе Шаума. Секретным значением является кортеж $s = (s_1, \ldots, s_l) \underset{r}{\in} \mathbf{Z}_{\varphi(n)}^l$, для проверки знания s используется открытый кортеж $(v_1, \ldots, v_l)$, где $v_i = g^{s_i}$ $(i = 1, \ldots, l)$.

   Протокол состоит из одного раунда, имеющего вид

$$\begin{cases} a \to b : x := g^z, \text{ где } z \underset{r}{\in} \mathbf{Z}_{\varphi(n)} \\ b \to a : (e_1, \ldots, e_l) \underset{r}{\in} \{0,1\}^l \\ a \to b : y := z + s_1 e_1 + \ldots + s_l e_l \\ b \text{ принимает } y, \text{ если } g^y = x v_1^{e_1} \ldots v_l^{e_l} \end{cases}$$

2) Открытые параметры: целое число $n > 0$, и элементы $g_1, \ldots, g_l$ группы $\mathbf{Z}_n^*$, имеющие большой порядок. Секретное значение – кортеж $s = (s_1, \ldots, s_l) \underset{r}{\in} \mathbf{Z}_{\varphi(n)}^l$, для проверки знания s используется открытое значение $v = g_1^{s_1} \ldots g_l^{s_l}$.

Протокол состоит из d раундов, имеющих вид

$$\begin{cases} a \to b : (x_1, \ldots, x_l), \text{где } x_i := g_i^{z_i}, z_i \underset{r}{\in} \mathbf{Z}_{\varphi(n)} \ (i = 1, \ldots, l) \\ b \to a : e \underset{r}{\in} \{0, 1\} \\ a \to b : (y_1, \ldots, y_l), \text{ где } y_i := z_i + s_i e \ (i = 1, \ldots, l) \\ b \text{ принимает } y, \text{ если } g_1^{y_1} \ldots g_l^{y_l} = x_1 \ldots x_l v^e \end{cases}$$

3) Открытые параметры: те же, что и в предыдущем протоколе. Секретным значением является $s \underset{r}{\in} \mathbf{Z}_{\varphi(n)}$, для проверки знания s используется открытый кортеж $(v_1, \ldots, v_l)$, где $v_i = g_i^s \ (i = 1, \ldots, l)$.

Протокол состоит из d раундов, имеющих вид

$$\begin{cases} a \to b : (x_1, \ldots, x_l), \text{ где } x_i := g_i^z \ (i = 1, \ldots, l), z \underset{r}{\in} \mathbf{Z}_{\varphi(n)} \\ b \to a : e \underset{r}{\in} \{0, 1\} \\ a \to b : y := z + se \\ b \text{ принимает } y, \text{ если } \forall\, i = 1, \ldots, l \quad g_i^y = x_i v_i^e \end{cases}$$

4.3.4. Протоколы Фиата-Шамира и Гиллу-Кискате

ПА Фиата-Шамира имеет следующие закрытые параметры: простые числа p, q, где $|p|, |q| \geq 512$. Открытый параметр: $n := pq$. Секретным значением является число $s \underset{r}{\in} \mathbf{Z}_n^*$, для проверки знания s используется число $v := s^2$.

Протокол состоит из d раундов, имеющих вид

$$\begin{cases} a \to b : x := z^2, \text{ где } z \underset{r}{\in} \mathbf{Z}_n, \ z \neq 0 \\ b \to a : e \underset{r}{\in} \{0, 1\} \\ a \to b : y := zs^e \\ b \text{ принимает } y, \text{ если } y^2 = xv^e \end{cases}$$

Если a хочет пройти аутентификацию, не зная s, то для каждого значения $e \in \{0,1\}$, которое ему пришлёт b, a должен уметь вычислить значение y_e, которое он пошлет b в качестве y. Из условия принятия y следует, что $y_1^2 = y_0^2 v$. Нетрудно видеть, что возможность $\forall\, v \in \mathbf{Z}_n^*$ за полиномиальное время вычислить числа y_0, y_1 с описанным выше свойством равносильна возможности за полиномиальное время вычислить $\sqrt{v}$ в $\mathbf{Z}_n$. Однако можно доказать, что при описанных выше условиях на n задача вычисления $\sqrt{v}$ в $\mathbf{Z}_n$ без знания p и q является вычислительно сложной.

Если a не знает s, то для прохождения аутентификации он может попытаться смошенничать, аналогично тому как это делается в протоколе Шаума: a случайно выбирает число $e' \in \{0,1\}$, в качестве x посылает значение $z^2 v^{-e'}$, и в качестве y – значение z. b примет этот ответ, если будет верно равенство $z^2 = z^2 v^{-e'} v^e$, которое равносильно равенству $e' = e$, и которое будет верно с вероятностью $1/2$.

Можно доказать, что данный протокол является Д0Р.

Обобщением ПА Фиата-Шамира является излагаемый ниже ПА **Гиллу-Кискате** получаемый из ПА Фиата-Шамира путем замены возведения в квадрат на возведение в степень l, где $l \geq 2$ – открытое число. В нём для проверки знания секретного значения s используется число $v := s^{-l}$. Каждый раунд имеет вид

$$\begin{cases} a \to b : x := z^l, \text{ где } z \underset{r}{\in} \mathbf{Z}_n^*,\ z \neq 0 \\ b \to a : e \underset{r}{\in} \mathbf{Z}_l \\ a \to b : y := z s^e \\ b \text{ принимает } y, \text{ если } x = y^l v^e \end{cases}$$

Данный ПА тоже является Д0Р.

4.3.5. Протокол Фейге-Фиата-Шамира

Открытый параметр: $n := pq$, где p, q – простые числа, и $(p)_4 = (q)_4 = 3$ (числа n подобного вида называются **числами Блюма**), причем p, q секретны и $|p|, |q| \geq 512$.

Секретным значением s является кортеж $(s_1, \ldots, s_l)$, где $s_i \underset{r}{\in} \mathbf{Z}_n^*$ $(i = 1, \ldots, l)$. Для проверки знания s используется открытый кортеж $(v_1, \ldots, v_l)$, где $v_i = \pm s_i^{-2}$ $(i = 1, \ldots, l)$, причем числа $v_1, \ldots, v_l$ должны быть различны (запись вида $\pm u$ обозначает выражение, значение ко-

торого равно либо значению выражения u, либо значению выражения $-u$).

Протокол состоит из d раундов, имеющих вид

$$\begin{cases} a \to b : x := \pm z^2, \text{ где } z \underset{r}{\in} \mathbf{Z}_n \setminus \{0\} \\ b \to a : (e_1, \ldots, e_l) \underset{r}{\in} \{0,1\}^l \\ a \to b : y := z s_1^{e_1} \ldots s_l^{e_l} \\ b \text{ принимает } y, \text{ если } x = \pm y^2 v_1^{e_1} \ldots v_l^{e_l} \end{cases}$$

Если a не знает s, то для прохождения аутентификации он может попытаться смошенничать: a выбирает кортеж $e' = (e'_1, \ldots, e'_l) \underset{r}{\in} \{0,1\}^l$, в качестве x посылает значение $z^2 v_1^{e'_1} \ldots v_l^{e'_l}$, и в качестве y – значение z. b примет этот ответ, если $e' = e$, что может случиться с вероятностью 2^{-l}.

Можно доказать, что данный протокол является Д0Р.

Рекомендуемые значения для l и d: $l = 5$, $d = 4$.

4.4. Протоколы аутентификации с передачей нового сеансового ключа

В некоторых случаях после аутентификации агентов начинается новый сеанс связи между ними, с использованием для шифрования пересылаемых сообщений нового ключа. Как правило, в этих случаях передача ключа для нового сеанса связи совмещена с аутентификацией в одном протоколе. Этот ключ может быть создан как одним из агентов a, b, так и доверенным посредником I. В этом пункте мы изложим несколько протоколов аутентификации с передачей нового сеансового ключа, который мы будем обозначать символом k.

4.4.1. Односторонняя аутентификация с передачей сеансового ключа

1) Простейшие протоколы (k создается агентом a):

   а) $\{a \to b : b^+\langle b, k, t\rangle_a$

   б) $\{a \to b : \langle b, b^+(a, k), t\rangle_a$

   в) $\{a \to b : b^+(k, t), \langle b, k, t\rangle_a$

2) Вопросно-ответный протокол (k создается агентом a):

$$\begin{cases} b \to a : r \\ a \to b : k_{ab}(b, k, r) \qquad (\text{или } a \to b : k \oplus h(b, k_{ab}, r)) \end{cases}$$

3) Протокол Wide Mouth Frog (k создается агентом a):

$$\begin{cases} a \to I : a, k_{aI}(b, k, t_a) \\ I \to b : k_{bI}(a, k, t_b) \end{cases}$$

4) Протокол Otway–Rees (k создается агентом I):

$$\begin{cases} a \to b : a, b, k_{aI}(a, b, r, r_a), r \\ b \to I : a, b, k_{aI}(a, b, r, r_a), k_{bI}(a, b, r, r_b), r \\ I \to b : k_{aI}(k, r_a), k_{bI}(k, r_b), r \\ b \to a : k_{aI}(k, r_a), r \end{cases}$$

4.4.2. Двусторонняя аутентификация с передачей сеансового ключа

1) Простейшие протоколы (k создается агентом a):

а) $$\begin{cases} a \to b : r_a \\ b \to a : k_{ab}(r_a, r_b) \\ a \to b : k_{ab}(b, k, r_b) \qquad (\text{или } k \oplus h(b, k_{ab}, r_b)) \end{cases}$$

б) $$\begin{cases} a \to b : \langle b^+(k) \rangle_a, k(t_a) \\ b \to a : k(t_b) \end{cases}$$

2) Otway–Rees (k создается агентом I):

$$\begin{cases} a \to b : a, b, k_{aI}(a, b, r, r_a), r \\ b \to I : a, b, k_{aI}(a, b, r, r_a), k_{bI}(a, b, r, r_b), r \\ I \to b : k_{aI}(k, r_a), k_{bI}(k, r_b), r \\ b \to a : k_{aI}(k, r_a), k(r_a, r_b) \\ a \to b : k(r_b) \end{cases}$$

3) Yahalom (k создается агентом I):

$$\begin{cases} a \to b : a, r_a \\ b \to I : b, k_{bI}(a, r_a, r_b) \\ I \to a : k_{aI}(b, k, r_a, r_b), k_{bI}(a, k) \\ a \to b : k_{bI}(a, k), k(r_b) \end{cases}$$

4) Woo–Lam (k создается агентом I):

$$\begin{cases} a \to b : b^+(a, r_a) \\ b \to I : a, b, I^+(r_a) \\ I \to b : b^+\langle a, b, k, r_a\rangle_I \\ b \to a : a^+(\langle a, b, k, r_a\rangle_I, r_b) \\ a \to b : k(r_b) \end{cases}$$

5) Needham–Schroeder (k создается агентом I):

$$\begin{cases} a \to I : a, b, r_a \\ I \to a : k_{aI}(b, k, k_{bI}(k, a, t), r_a) \\ a \to b : k_{bI}(k, a, t) \\ b \to a : k(r_b) \\ a \to b : k(r_b - 1) \end{cases}$$

6) Neuman–Stubblebine (k создается агентом I):

$$\begin{cases} a \to b : \ a, \ r_a \\ b \to I : b, r_b, k_{bI}(a, r_a, t) \\ I \to a : k_{aI}(b, k, r_a, t), k_{bI}(k, a, t), r_b \\ a \to b : k_{bI}(k, a, t), k(r_b) \end{cases}$$

7) Kerberos:

- Прототип (k создается агентом I):

$$\begin{cases} a \to I: \ a, \ b, \ r \\ I \to a : k_{aI}(b, k, r, l), k_{bI}(a, k, l) \\ a \to b : k_{bI}(a, k, l), k(a, r', t) \\ b \to a : k(r', t) \end{cases}$$

где l = время действия ключа k.

- Основной протокол Kerberos предполагает работу с несколькими доверенными посредниками: I (authentication server) и $I_1, \ldots, I_n$ (tickets grant servers). k создается одним из агентов $I_1, \ldots, I_n$.

$$\begin{cases} a \to I : a, I_i, r \\ I \to a : k_{aI}(I_i, k_{aI_i}, r, l_1), k_{I_iI}(a, k_{aI_i}, l_1) \\ a \to I_i : k_{I_iI}(a, k_{aI_i}, l_1), k_{aI_i}(a, t_1), b, r' \\ I_i \to a : k_{aI_i}(b, r', k, l_2), k_{bI_i}(a, k, l_2) \\ a \to b : k_{bI_i}(a, k, l_2), k(a, r'', t_2) \\ b \to a : k(r'', t_2) \end{cases}$$

5. Алгоритмы вычисления цифровой подписи

Как было определено в пункте 3.3, алгоритм вычисления цифровой подписи преобразует пару (m, a), где m – подписываемое сообщение, и a – имя агента, подписывающего это сообщение, в строку $\langle m \rangle_a^s$, называемую цифровой подписью сообщения m, созданной агентом a. Ниже мы будем называть **цифровой подписью (ЦП)** не только строку $\langle m \rangle_a^s$, но и алгоритм вычисления этой строки.

В каждой из излагаемых ниже ЦП входными данными являются подписываемое сообщение m, а также дополнительные аргументы, называемые **параметрами**, каждый из которых может быть

- закрытым (т.е. известным только подписывающему агенту, один из таких параметров - закрытый ключ s_a подписывающего агента a), или

- открытым (т.е. известным всем, один из таких параметров – значение v_a, предназначенное для проверки подлинности ЦП, создаваемых агентом a).

Каждый из параметров по умолчанию считается открытым, а если он закрыт, то это специально оговаривается.

Проверка подлинности подписанного сообщения $\langle m\rangle_a$ заключается в вычислении значения некоторого булевозначного выражения e, аргументами которого являются это подписанное сообщение и открытые параметры ЦП. Как правило, выражение e (которое мы будем называть **проверкой ЦП**) имеет вид равенства. Подписанное сообщение считается подлинным, если e истинно.

Ниже используется следующее обозначение: если x – битовая строка, и n – целое положительное число, то запись $Pref(x, n)$ обозначает строку, состоящую из первых n битов строки x (если $|x| \geq n$), или строку, получаемую приписыванием к x справа $n - |x|$ нулей (если $|x| < n$).

5.1. Простейшие цифровые подписи

5.1.1. Цифровая подпись DSA

ЦП DSA (Digital Signature Algorithm) имеет следующие параметры: простые числа p и q, где $q \mid p-1$, $|p| \geq 512$ и делится на 64, $|q|$ – примерно 160, элемент $g \in \mathbf{Z}_p^*$ порядка q, ХФ h со значениями в $\mathbf{Z}_q^*$, $s_a \underset{r}{\in} \mathbf{Z}_q$, $v_a := g^{s_a}$.

$$\langle m\rangle_a^s := (s_1, s_2), \text{ где } \begin{cases} s_1 := (g^z)_q, \text{ где } z \underset{r}{\in} \mathbf{Z}_q^*, \\ s_2 := (h(m) + s_1 s_a) z^{-1}. \end{cases}$$

Проверка ЦП: $s_1 = (g^{h(m)s_2^{-1}} v_a^{s_1 s_2^{-1}})_q$.

5.1.2. Цифровая подпись ГОСТ

Параметры этой ЦП те же, что и у DSA, только $|q| = 256$.

$$\langle m\rangle_a^s := (s_1, s_2), \text{ где } \begin{cases} s_1 := (g^z)_q, \quad \text{где } z \underset{r}{\in} \mathbf{Z}_q^*, \\ s_2 := h(m) z + s_1 s_a. \end{cases}$$

Проверка ЦП: $s_1 = (g^{s_2 h(m)^{-1}} v_a^{-s_1 h(m)^{-1}})_q$.

5.1.3. Цифровая подпись Эль-Гамаля

Параметры: открытое простое число p, примитивный элемент $g \in \mathbf{Z}_p^*$, ХФ h со значениями в $\mathbf{Z}_{p-1}$, $s_a \underset{r}{\in} \mathbf{Z}_{p-1}$, $v_a := g^{s_a}$.

$\langle m\rangle_a^s := (s_1, s_2)$, где $\begin{cases} s_1 := g^z, \quad z \underset{r}{\in} \mathbf{Z}_{p-1}^* \\ s_2 := (h(m) - s_a s_1) z^{-1}. \end{cases}$

Проверка ЦП: $v_a^{s_1} s_1^{s_2} = g^{h(m)}$.

5.1.4. Слепая цифровая подпись

В некоторых случаях требуется, чтобы агент a, не получая никакой информации о сообщении m, создал бы такое значение, из которого можно извлечь $\langle m\rangle_a^s$. ЦП, получаемую при таких условиях, называют **слепой** ЦП.

Один из протоколов слепой ЦП имеет следующий вид. Параметры этой ЦП: секретные большие простые числа p и q, открытое число $n \overset{\text{def}}{=} pq$, ХФ h со значениями в $\mathbf{Z}_n$, $v_a \underset{r}{\in} \mathbf{Z}_{\varphi(n)}^*$, $s_a \in \mathbf{Z}_{\varphi(n)}^*$ удовлетворяет равенству $s_a v_a = 1$. Получение агентом b от агента a слепой ЦП сообщения m происходит по следующему протоколу:

$$\begin{cases} b \to a : u := h(m) x^{v_a}, \text{ где } x \underset{r}{\in} \mathbf{Z}_n^* \\ a \to b : v := u^{s_a} \\ b \text{ вычисляет искомое значение } \langle m\rangle_a^s := v x^{-1} \end{cases}$$

Нетрудно видеть, что

$$\langle m\rangle_a^s = v x^{-1} = (u^{s_a}) x^{-1} = ((h(m) x^{v_a})^{s_a}) x^{-1} =$$
$$= h(m)^{s_a} x^{v_a s_a} x^{-1} = h(m)^{s_a} x x^{-1} = h(m)^{s_a}.$$

Проверка подлинности: утверждение $y = \langle m\rangle_a^s$ считается верным, если $y^{v_a} = h(m)$.

5.2. Цифровая подпись, получаемая из протоколов аутентификации

5.2.1. Цифровая подпись Шнорра

ЦП Шнорра получается из ПА Шнорра, изложенного в пункте 4.3.2, в ней используются следующие параметры:

- простые числа p и q, где $|p| \geq 512$, $|q| \geq 140$, $q \mid p - 1$,
- элемент $g \in \mathbf{Z}_p^*$, такой, что $\operatorname{ord}(g) = q$,

- ХФ h со значениями в $\mathbf{Z}_q$,
- закрытый и открытый ключи: $s_a \underset{r}{\in} \mathbf{Z}_q^*$, $v_a := g^{-s_a}$.

$\langle m \rangle_a^s := (e, y)$, где $e := h(m, g^z)$, $y := z + s_a e$, $z \underset{r}{\in} \mathbf{Z}_q^*$.

Проверка ЦП (e, y): $h(m, v_a^e g^y) = e$.

5.2.2. Цифровая подпись Фиата-Шамира

ЦП Фиата-Шамира получается из ПА Фиата-Шамира, изложенного в пункте 4.3.4, заменой последовательности случайных битов $e_1, \ldots, e_d$, которые b посылает a в каждом из d раундов после получения x_i от a $(i = 1, \ldots, d)$, на префикс длины d строки $h(m, x_1, \ldots, x_d)$.

Параметры: n – открытое число вида pq, где p, q – секретные простые числа, $|p|, |q| \geq 512$, $s_a \underset{r}{\in} \mathbf{Z}_n^*$, $v_a := s_a^2$.

$\langle m \rangle_a^s := (e, y)$, где

- $e = Pref(h(m, z_1^2, \ldots, z_d^2), d)$, где $(z_1, \ldots, z_d) \underset{r}{\in} (\mathbf{Z}_n \setminus \{0\})^d$
- $y = (z_1 s_a^{e_1}, \ldots, z_d s_a^{e_d})$, где $(e_1, \ldots, e_d) = e$.

Проверка ЦП: $Pref(h(m, y_1^2 v_a^{-e_1}, \ldots, y_d^2 v_a^{-e_d}), d) = e$.

5.2.3. Цифровая подпись Гиллу-Кискате

Эта ЦП получается из ПА Гиллу-Кискате, изложенного в пункте 4.3.4, заменой случайного значения $e \underset{r}{\in} \mathbf{Z}_l$, которое генерирует b после получения $x := z^l$ от a, на $h(m, x)$, где h – ХФ с множеством значений $\mathbf{Z}_l$.

Параметры: n – открытое число вида pq, где p, q – секретные простые числа, $|p|, |q| \geq 512$, $l \geq 2$, $s_a \underset{r}{\in} \mathbf{Z}_n^*$, $v_a := s^{-l}$.

$\langle m \rangle_a^s := (e, y)$, где $e := h(m, z^l)$, $z \underset{r}{\in} \mathbf{Z}_n^* \setminus \{0\}$, $y := z s_a^e$.

Проверка подлинности: $e = h(m, y^l v_a^e)$.

5.2.4. Цифровая подпись Фейге-Фиата-Шамира

Эта ЦП получается из ПА Фейге-Фиата-Шамира, её параметры: открытое число $n := pq$, где p, q – секретные простые числа Блюма, $|p|, |q| \geq 512$, открытые числа l и d (рекомендуемые значения: $l = 9, d = 8$), $s_a = (s_1, \ldots, s_l) \underset{r}{\in} (\mathbf{Z}_n^*)^l$, $v_a = (v_1, \ldots, v_l)$, где $v_i := s_i^{-2}$ $(i = 1, \ldots, l)$, причем числа $v_1, \ldots, v_l$ различны.

$\langle m \rangle_a^s := (e, y)$, где

- $e = Pref(h(m, z_1^2, \ldots, z_d^2), dl)$, где $(z_1, \ldots, z_d) \underset{r}{\in} (\mathbf{Z}_n \setminus \{0\})^d$, обозначим строку e записью $(e_{11}, \ldots, e_{1l}, \ldots, e_{d1}, \ldots, e_{dl})$,

- $y = (y_1, \ldots, y_d)$, где $\forall\, i = 1, \ldots, d \quad y_i := z_i s_1^{e_{i1}} \ldots s_l^{e_{il}}$.

Проверка ЦП: $e = Pref(h(m, u_1, \ldots, u_d), d)$, где $\forall\, i = 1, \ldots, d \quad u_i := y_i^2 v_1^{e_{i1}} \ldots v_l^{e_{il}}$.

5.3. Стираемая цифровая подпись

5.3.1. Понятие стираемой цифровой подписи

Иногда требуется создать такую ЦП $\langle m \rangle_a^s$, чтобы

- подлинность этой ЦП могла быть доказана (или опровергнута) только с участием **уполномоченных агентов** (**designated confimers**), и

- если какой-либо уполномоченный агент доказал (или опроверг) подлинность этой ЦП какому-либо неуполномоченному агенту b, то b не мог бы использовать запись этого доказательства (или опровержения) для того, чтобы доказать (или опровергнуть) подлинность $\langle m \rangle_a^s$ другим агентам.

ЦП такого вида называется **стираемыми** (**undeniable**).

Доказательство или опровержение подлинности определяемых в этом параграфе стираемых ЦП производится при помощи интерактивных протоколов, где под **интерактивным протоколом (ИП)** доказательства справедливости какого-либо утверждения A понимается протокол, удовлетворящий следующим условиям:

- если A верно, то это будет установлено в результате работы этого протокола с большой вероятностью, и

- если A неверно, то установить в результате работы этого протокола обратное (т.е. то, что A верно) можно с небольшой вероятностью.

В пунктах 5.3.2 и 5.3.3 рассматриваются случаи, когда уполномоченным является только a, а в 5.3.4 – случай когда уполномоченными являются a и ещё один агент c.

5.3.2. Простейшая стираемая цифровая подпись

Излагаемая в этом пункте стираемая ЦП принадлежит Шауму и Антверпену. Её параметры: простые числа p и q, где $q \mid p-1$, элемент $g \in \mathbf{Z}_p^*$ порядка q, $s_a \underset{r}{\in} \mathbf{Z}_q^*$, $v_a := g^{s_a}$.

Будем предполагать, что подписываемое сообщение m является элементом $\mathbf{Z}_p^*$ (если это не выполняется, то заменим m на $h(m)$, где h – ХФ со значениями в $\mathbf{Z}_p^*$).

$\langle m \rangle_a^s := m^{s_a}$.

Протокол подтверждения подлинности $\langle m \rangle_a^s$ (т.е. доказательства утверждения $z = m^{s_a}$):

$$\begin{cases} b \to a : y := z^u v_a^v, \text{ где } u, v \underset{r}{\in} \mathbf{Z}_q \\ a \to b : w := y^{(s_a^{-1})} \\ b \text{ принимает ЦП, если } w = m^u g^v \end{cases}$$

(отметим, что этот протокол не обеспечивает нулевого разглашения s_a).

Стираемость этой ЦП обосновывается тем, что b может самостоятельно вычислить значение $m^u g^v$ и предъявить его в качестве того значения w, которое ему якобы переслал a.

5.3.3. Стираемая цифровая подпись с протоколом опровержения

Излагаемая в этом пункте стираемая ЦП с протоколом опровержения принадлежит Шауму. Её параметры: простое число p, примитивный элемент $g \in \mathbf{Z}_p^*$, $s_a \underset{r}{\in} \mathbf{Z}_p^*$, $v_a := g^{s_a}$.

Будем предполагать, что подписываемое сообщение m является элементом $\mathbf{Z}_p^*$ (если это не выполняется, то заменяем m на $h(m)$, где h – ХФ со значениями в $\mathbf{Z}_p^*$).

$\langle m \rangle_a^s := m^{s_a}$.

1) Протокол подтверждения подлинности $\langle m \rangle_a^s$ (т.е. доказательство утверждения $z = m^{s_a}$, где $z \in \mathbf{Z}_p^*$):

$$\begin{cases} b \to a : y := m^u g^v, \text{ где } u, v \underset{r}{\in} \mathbf{Z}_{p-1} \\ a \to b : (w_1, w_2), \text{ где } w_1 := yg^w, w_2 := w_1^{s_a}, \ w \underset{r}{\in} \mathbf{Z}_{p-1} \\ b \to a : (u, v) \\ a \to b : [\![y = m^u g^v]\!] \ w \\ b \text{ принимает ЦП, если } w_1 = yg^w \text{ и } w_2 = z^u v_a^{v+w} \end{cases}$$

Отметим, что данный протокол не обеспечивает нулевого разглашения s_a.

Стираемость этой ЦП обосновывается тем, что b может сам выбрать $w \underset{r}{\in} \mathbf{Z}_{p-1}$, и предъявить $(yg^w, z^u v_a^{v+w})$ как пару (w_1, w_2), которую ему якобы переслал a.

2) Протокол опровержения подлинности $\langle m \rangle_a^s$ (т.е. доказательство утверждения $z \neq m^{s_a}$, где $z \in \mathbf{Z}_p^*$): выбирается небольшое число $k \geq 2$, и

$$\begin{cases} b \to a : & (y_1 := m^u g^v, \ y_2 := z^u v_a^v), \text{ где } u \underset{r}{\in} \mathbf{Z}_k, v \underset{r}{\in} \mathbf{Z}_{p-1} \\ a \to b : & ru', \text{ где } r \underset{r}{\in} \mathbf{Z}_p^*, \text{ и } u' \in \mathbf{Z}_k \text{ – решение уравнения} \\ & \frac{y_1^{s_a}}{y_2} = (\frac{m^{s_a}}{z})^{u'}, \text{ которое ищется перебором} \\ & (u \text{ – одно из решений этого уравнения}) \\ b \to a : & v \\ a \to b : & [\![(y_1 = m^{u'} g^v) \wedge (y_2 = z^{u'} v_a^v)]\!] \ r \\ b : & \text{принимает опровержение, если } u' = u \end{cases}$$

Для обоснования корректности данного протокола заметим, что если $z = m^{s_a}$, то $\forall\, u' \in \mathbf{Z}_k \ \ u'$ является решением уравнения в описанном выше протоколе, поэтому вероятность того что $u' = u$, равна $1/k$.

5.3.4. Стираемая цифровая подпись, подтверждаемая двумя уполномоченными агентами

В этом пункте мы рассмотрим пример такой ЦП, в которой уполномоченными агентами являются подписывающий агент a и ещё один агент c. Её параметры: большое простое число p, элемент $g \in \mathbf{Z}_p^*$ порядка $p-1$, $s_a = (p_1, p_2)$, где p_1, p_2 – большие простые числа, $s_c \underset{r}{\in} \mathbf{Z}_{p-1}$, $v_a = (p, g, \eta, n)$, где $\eta := g^{s_c}$, $n := p_1 p_2$, ХФ h со значениями в $\mathbf{Z}_n$.

$\langle m \rangle_a^s := (s_1, s_2, s_3)$, где

$$s_1 = g^y \ \ (y \underset{r}{\in} \mathbf{Z}_{p-1}), \quad s_2 = \eta^y, \quad s_3 = \sqrt[3]{h(m) \oplus h(s_1, s_2)},$$

где $\sqrt[3]{\ }$ – функция вида $\mathbf{Z}_n \to \mathbf{Z}_n$ (т.к. a знает разложение n на простые множители, то он может вычислять её быстро).

Доказательство подлинности ЦП $\langle m \rangle_a^s$ агентом a:

$$\begin{cases} b \to a : z := g^u \eta^v, \text{ где } u, v \underset{r}{\in} \mathbf{Z}_{p-1} \\ a \to b : (d := g^w, e := (zd)^y), \text{ где } w \underset{r}{\in} \mathbf{Z}_{p-1} \\ b \to a : (u, v) \\ a \to b : [\![g^u \eta^v = z]\!] \ w \\ b \text{ принимает ЦП, если } g^w = d,\ e = s_1^{u+w} {s_2}^v, \text{ и} \\ \qquad h(m) \oplus h(s_1, s_2) = {s_3}^3 \end{cases}$$

Доказательство подлинности ЦП $\langle m \rangle_a^s$ агентом c получается из предыдущего протокола заменой a на c, y на s_c, η на s_1, и s_1 на η.

На базе этой ЦП и схемы разделения секрета можно построить такую ЦП, в которой доказывать подлинность ЦП a могут любые m агентов из заданной совокупности.

5.4. Совместная цифровая подпись

5.4.1. Понятие совместной цифровой подписи

В некоторых случаях требуется, чтобы сообщение m было одновременно подписано несколькими агентами $a_1, \ldots, a_k$. Соответствующая ЦП называется **совместной ЦП**, обозначается записью $\langle m \rangle_{a_1 \ldots a_k}^s$.

5.4.2. Примеры совместных цифровых подписей

Совместная ЦП может иметь например следующий вид.

1) $\langle m\rangle^s_{a_1\dots a_k} := (\langle m\rangle^s_{a_1}, \dots, \langle m\rangle^s_{a_k})$.

2) $\langle m\rangle^s_{a_1\dots a_k} := a_k^-(\dots(a_1^-(h(m)))\dots)$ (в данном случае агенты a_1, …, a_k используют общую АСШ).

3) Ещё одна совместная ЦП имеет следующие параметры: открытые натуральные числа n и u, ХФ h со значениями в $\mathbf{Z}_u$, $\forall\, i = 1, \dots, k \quad s_{a_i} \underset{r}{\in} \mathbf{Z}_n^*,\ v_{a_i} := s_{a_i}^{-u}$.

   $\langle m\rangle^s_{a_1\dots a_k} := (c, d)$, где $c := h(m, r_1^u \dots r_k^u)$,
   $d := r_1 s_{a_1}^c \dots r_k s_{a_k}^c$, $\forall\, i = 1, \dots, k \quad r_i \underset{r}{\in} \mathbf{Z}_n \setminus \{0\}$.

   Проверка подлинности: $c = h(m, d^u v_{a_1}^c \dots v_{a_k}^c)$.

5.4.3. Совместная цифровая подпись Брикелла-Ли-Якоби

Эта ЦП имеет следующие параметры: простое число p, элемент $g \in \mathbf{Z}_p^*$ порядка $p-1$, большое натуральное число l, ХФ h, $\forall\, i = 1, \dots, k \ \ s_{a_i} = \{s_{i1}, \dots, s_{il}\},\ \ v_{a_i} = \{v_{i1}, \dots, v_{il}\}$, где $\forall\, j = 1, \dots, l \ \ s_{ij} \underset{r}{\in} \mathbf{Z}_{p-1}$, $v_{ij} := g^{-s_{ij}}$.

В вычислении ЦП $\langle m\rangle^s_{a_1\dots a_k}$ принимает участие доверенный посредник I, с которым агенты $a_1, \dots, a_k$ могут обмениваться сообщениями с использованием общей АСШ. Протокол вычисления $\langle m\rangle^s_{a_1\dots a_k}$ имеет следующий вид:

$$\begin{cases} \forall\, i = 1, \dots, k \quad a_i \to I : I^+(y_i), \text{ где } y_i := g^{x_i}, x_i \underset{r}{\in} \mathbf{Z}_{p-1} \\ I \to \{a_1, \dots, a_k\} : y := y_1 \dots y_k \\ a_i \to I : z_i := x_i + \sum\limits_{j\in\{1,\dots,l\},\ b_j=1} s_{ij}, \quad b_j = \ j\text{-й бит } g^{h(m,y,a)} \\ \qquad\qquad \text{где } a \text{ – конкатенация имён агентов } a_1, \dots, a_k \\ I \text{ вычисляет } \langle m\rangle^s_{a_1\dots a_k} = z := \sum_{i=1}^k z_i \end{cases}$$

Проверка подлинности: $g^z \prod_{i=1}^k \prod\limits_{j\in\{1,\dots,l\},\ b_j=1} v_{ij} = y$.

Можно доказать, что данный протокол обеспечивает нулевое разглашение $s_{a_1}, \dots, s_{a_k}$, и если совместная ЦП не может быть создана по причине того, что некоторые агенты отказались вносить свой вклад в её

создание, в то время как другие агенты свой вклад уже внесли, то по результату работы агентов, внесших свой вклад в создание общей ЦП, невозможно идентифицировать этих агентов.

Список литературы

[1] Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. - М.: Триумф, 2002. - 816 с.

[2] Черемушкин А.В. Криптографические протоколы. Основные свойства и уязвимости. - М.: Академия, 2009. - 269 с.

[3] Алферов А.П., Зубов А.Ю., Кузьмин А.С., Черемушкин А.В. Основы криптографии. - М.: Гелиос АРВ, 2002. - 480 с.

[4] Запечников С.В. Криптографические протоколы и их применение в финансовой и коммерческой деятельности. - М.: Горячая Линия - Телеком, 2007. - 320 с.

[5] Анохин М.И., Варновский Н.П., Сидельников В.М., Ященко В.В. Криптография в банковском деле. - М.: МИФИ, 1997.

Security Protocols, Part 1

Mironov A.M.

The main cryptographic primitives used in security protocols are described (symmetric and asymmetric encryption systems, hash functions, secret sharing schemes), authentication protocols, and digital signature algorithms.

Keywords: protocols, security, cryptography, hash functions, authentication, digital signature