Криптографические протоколы
Часть 1

А.М. Миронов

Оглавление

Глава 1

Введение

1.1 Понятие криптографического протокола

Криптографический протокол (**КП**), называемый также просто **протоколом** – это распределённый алгоритм, определяющий порядок обмена сообщениями между несколькими **агентами**, в качестве которых могут выступать, например, люди, компьютерные программы, вычислительные комплексы, базы данных, сети связи, банковские карточки, и т.д. Агенты, принимающие участие в работе протокола, называются **участниками** этого протокола.

Действия, выполняемые каждым из участников протокола, могут иметь следующий вид:

- **посылка сообщения** другому участнику этого протокола (или группе участников),
- **приём сообщения** от другого участника,
- **внутренние действия**, к числу которых относятся
  - выполнение участником некоторых вычислений,
  - проверка логических условий, и
  - обновление значений переменных.

Криптографические протоколы предназначены для обеспечения безопасности передачи, обработки и хранения информации в небезопасной среде. Свойства безопасности, которые должен обеспечивать протокол, 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.

1.2 Шифрование сообщений

1.2.1 Понятие шифрования и дешифрования

Некоторые из сообщений, пересылаемых участниками КП, 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 $Encr$ (называемого **алгоритмом шифрования**) к паре (k, m), где

- k – битовая строка, называемая **ключом шифрования** (или просто **ключом**), и

- m – шифруемое сообщение, называемое в данном случае **открытым текстом (ОТ)**.

Мы будем обозначать результат применения алгоритма $Encr$ 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 $Decr$, называемый **алгоритмом дешифрования**, и обладающий следующими свойствами:

- алгоритм $Decr$ получает на вход пару вида (d, u), где

  - d – битовая строка, называемая **ключом дешифрования**, и
  - u – сообщение,

  результат применения алгоритма $Decr$ к паре (d, u) будем обозначать записью $d(u)$,

- каждому ключу шифрования k должен соответствовать ключ дешифрования d_k, такой, что для каждого сообщения m верно равенство

$$d_k(k(m)) = m. \tag{1.1}$$

Будем использовать следующее обозначение: если k – ключ шифрования или дешифрования, и m_1, ..., m_n – список сообщений, то запись $k(m_1, \ldots, m_n)$ обозначает результат соответствующего криптографического преобразования с ключом k, которое применено к сообщению, являющемуся конкатенацией сообщений $m_1, \ldots, m_n$.

1.2.2 Системы шифрования

Система шифрования (СШ) представляет собой набор следующих данных:

- пара алгоритмов *Encr* и *Decr* 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.
2. В **асимметричных** СШ **(АСШ)** ключи шифрования могут отличаться от соответствующих им ключей дешифрования, и
   - 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 (и, возможно, некоторым доверенным агентам).

Некоторые АСШ обладают следующим свойством: для каждого сообщения m верно равенство

$$a^+a^-(m) = m. \tag{1.2}$$

Такие АСШ можно использовать для **аутентификации** агентов: если какой-либо агент a, использующий эту АСШ, хочет доказать свою подлинность (т.е. доказать, что он действительно является тем, за кого себя выдаёт), то он может сделать это путем предъявления пары (m, m'), где m – произвольное сообщение, и $m' = a^-(m)$. Доказательство подлинности a будет принято, если будет выполнено условие

$$a^+(m') = m. \tag{1.3}$$

Данное условие обосновывается предположением о том, что

- без знания закрытого ключа a^- создать сообщение m', удовлетворяющее условию (1.3), невозможно, и
- никто, кроме агента a, не должен знать ключ a^-.

1.3 Электронная подпись

Электронная подпись является одним из средств, предназначенных для доказательства подлинности передаваемых сообщений и их отправителей.

Протокол электронной подписи – это алгоритм, преобразующий пару (m, a), где m – сообщение и a – имя агента, в строку,

- обозначаемую записью $\langle m \rangle_a^s$, и
- называемую **электронной подписью (ЭП)** сообщения m, созданной агентом a.

При вычислении $\langle m \rangle_a^s$ используется **закрытый ключ** a^-, который должен быть известен только агенту a.

Тройку $(m, a, \langle m \rangle_a^s)$ мы будем обозначать записью $\langle m \rangle_a$.

ЭП должна обладать следующими свойствами:

- **возможность проверки подлинности ЭП**: существует открытый алгоритм позволяющий по тройке (m, a, u) проверить истинность равенства $u = \langle m \rangle_a^s$,

- **невозможность подделки ЭП**: задача вычисления $\langle m \rangle_a^s$ без знания 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.

В том случае, когда подписывающий агент a и проверяющий агент b могут использовать одну и ту же АШС, $\langle m \rangle_a^s$ может иметь один из следующих видов: $a^-(m)$, $b^+a^-(m)$, $a^-(a, a^-(m), t)$, где t – метка времени, и т.п.

1.4 Хэш-функции

Хэш-функция (ХФ) - это функция h вида

$$h : D \to \{0,1\}^n, \quad \text{где } D \subseteq \{0,1\}^*$$

(где n – заданное число, $\{0,1\}^n$ – множество битовых последовательностей длины n, $\{0,1\}^*$ – множество конечных битовых последовательностей), алгоритм вычисления которой должен быть общеизвестен, и которая обладает следующими свойствами:

- h – **односторонняя функция**, т.е. не существует быстрого алгоритма нахождения по заданному $y \in \{0,1\}^n$ 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 $h(x_1) = h(x_2)$.

ХФ применяются для контроля целостности данных, аутентификации источников данных, и многих других целей.

Будем использовать следующее обозначение: если h – ХФ, и m_1, ..., m_k – список сообщений, то запись $h(m_1, \ldots, m_k)$ обозначает значение функции h на сообщении, являющемся конкатенацией сообщений $m_1, \ldots, m_k$.

1.5 Формальные описания и примеры криптографических протоколов

1.5.1 Понятие формального описания протокола

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

$$a \to b : [\![\varphi]\!]\, m, \tag{1.4}$$

$$a \to \{b_1, \ldots, b_n\} : [\![\varphi]\!]\, m, \tag{1.5}$$

или

$$a : [\![\varphi]\!] \text{ внутреннее действие}, \tag{1.6}$$

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 КП передаёт другому в процессе функционирования КП, этот объект может быть как символьной строкой, так и набором банкнот, товаром, и т.п.). Записи (1.4), (1.5) и (1.6) изображают действия, которые должны выполнять агенты в процессе функционирования КП.

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

- (1.4) и (1.5) выполняется путем проверки φ, и
  - если φ выполнено, то a посылает сообщение m агенту b (в случае (1.4)), или агентам $b_1, \ldots, b_n$ (в случае (1.5)),
  - если же φ не выполнено, то данное действие пропускается, и выполняется следующее по списку действие,
- (1.6) 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 КП, то этот участник не функционирует в момент исполнения этого действия.

В формальных записях протоколов могут использоваться выражения с переменными. Во время выполнения протокола каждой из таких переменных сопоставлено некоторое значение, и это значение может изменяться во время выполнения протокола. Операции изменения значений переменных изображаются записями вида

$$x := e \tag{1.7}$$

где x – переменная, и e – выражение. Значение, которое имеет запись (1.7) во время выполнения действия содержащего (1.7), равно значению выражения e на текущих значениях входящих в него переменных, и это же значение присваивается переменной x после выполнения действия, в которое входит запись (1.7).

Также могут использоваться выражения вида

- $[\![e]\!]\, e_1 : e_2$ и $[\![e]\!]\, e_1$, значения которых равны e_1, если $e = 1$, и e_2 (или не определено), если $e \neq 1$,

- $e_1 \oplus e_2$, где значениями e_1 и e_2 являются битовые строки одинакового размера, значение $e_1 \oplus e_2$ получается путем побитового сложения (по модулю 2) этих строк.

В формальной записи протоколов могут также использоваться выражения вида (1.4) и (1.5), в которых символ m обозначает не одно сообщение, а множество сообщений. При выполнении действий такого типа предполагается, что перед тем, как a посылает b данное множество, он его случайно и равновероятно перетасовывает (т.е. последовательность, в которой b получит сообщения из данного множества, является случайной равновероятной перестановкой последовательности, в которой a посылал эти сообщения).

1.5.2 Нонсы

Некоторые сообщения, используемые в протоколах, имеют особый смысл. Одними из таких сообщений являются **нонсы**. Каждый нонс (**nonce**, это слово является аббревиатурой словосочетания number used once) представляет собой большую (несколько сотен битов) псевдослучайную строку, сгенерированную во время какого-либо сеанса работы протокола. Нонсы обозначаются символом r, как правило, с индексом, обозначающим того агента который сгенерировал этот нонс. Мы будем предполагать, что нонсы, сгенерированные в разных сеансах протокола (или нонсы, сгенерированные в одном и том же сеансе, но обозначаемые разными записями), являются уникальными.

1.5.3 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

$$a \to b : \text{ компьютер}$$
$$b \to a : \text{ деньги}$$

и

$$b \to a: \text{ деньги}$$

$$a \to b: \text{ компьютер}$$

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 s. Данный протокол может иметь, например, следующий вид:

- $a \to s$: компьютер
- $b \to a$: деньги
- $a \to s: \left(\begin{array}{l} \text{подтверждение или опровержение} \\ \text{того, что полученная от } b \text{ сумма} \\ \text{соответствует стоимости компьютера} \end{array} \right)$
- $s \to b: [\![\text{от } A \text{ поступило подтверждение}]\!]$ компьютер
- $s \to a: [\![\text{от } A \text{ поступило опровержение}]\!]$ компьютер

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

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

1.6 Уязвимости протоколов

1.6.1 Понятие уязвимости протокола

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

Противники подразделяются на следующие два класса:

- **пассивные противники**, они могут перехватывать сообщения, пересылаемые участниками КП, 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 КП,
  - выдавать себя за участников КП.

Кроме того, нарушения свойств безопасности КП возможны из-за действий участников КП, которые могут нарушать (умышленно или неумышленно) предписанные протоколом правила взаимодействия с другими участниками этого КП.

Возможность нарушения свойств безопасности в процессе работы КП называют **уязвимостями** этого КП.

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

- противник полностью знает КП,
- противнику доступны все сообщения, пересылаемые между участниками КП, он может их модифицировать, удалять, и заменять своими сообщениями,
- противник не может извлечь из перехваченных ШТ соответствующие ОТ, если он не знает необходимых ключей.

1.6.2 Пример уязвимости протокола

В этом пункте мы рассмотрим пример уязвимости КП, 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,

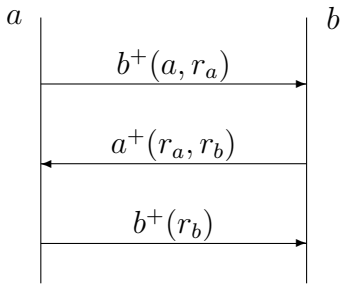
что он содержит уязвимость (которая была обнаружена автоматической системой верификации). Этот КП называется **протоколом Нидхема--Шредера** (Needham-Schroeder, 1979), мы будем обозначать его записью NS. Целью данного протокола является взаимная аутентификация агентов a и b.

Предполагается, что a и b используют общую АСШ.

Последовательность действий, из которых состоит протокол NS, имеет следующий вид:

$$\begin{aligned} a &\to b : b^+(a, r_a) \\ b &\to a : a^+(r_a, r_b) \\ a &\to b : b^+(r_b) \end{aligned} \tag{1.8}$$

Ниже мы объясним смысл этих действий. Отметим, что для формальной записи КП используется также нотация в виде диаграмм. Каждое действие в КП, связанное с пересылкой сообщений, изображается в диаграмме горизонтальной стрелкой, помеченной пересылаемым сообщением. Диаграмма, соответствующая КП (1.8), имеет вид



Действия, входящие в КП NS, имеют следующий смысл.

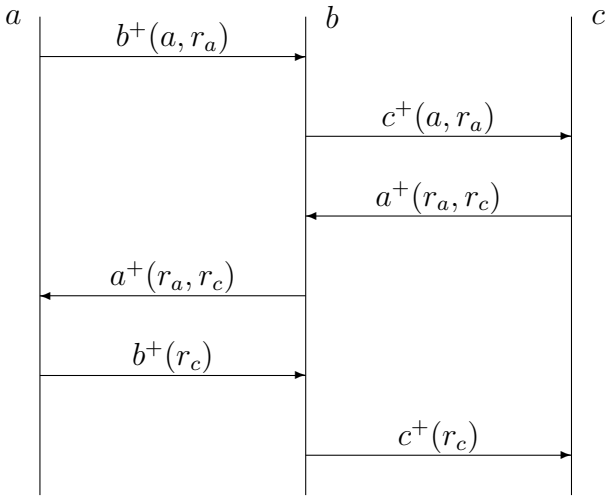
- Первое действие заключается в пересылке от a к b ШТ $b^+(a, r_a)$ где соответствующий ОТ – пара, состоящая из имени агента a, и нонса r_a, который используется для того, чтобы дать возможность b доказать свою подлинность путем извлечения данного нонса из $b^+(a, r_a)$.

- Второе действие заключается в пересылке от b к a ШТ $a^+(r_a, r_b)$ где соответствующий ОТ – пара, состоящая из нонса r_a, который агент b извлёк из полученного ШТ $b^+(a, r_a)$, и нонса r_b, сгенерированного агентом b. После его получения a убеждается, что его отправителем может быть только b, т.к. никто кроме b не может получить r_a. Нонс r_b предназначен для того, чтобы дать возможность a тоже доказать 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.

Одна из уязвимостей данного КП связана с тем, что агент b может использовать свой статус участника КП NS для того, чтобы во взаимодействии с другими агентами выдавать себя за агента a. Использование агентом b своего статуса участника КП NS для совершения мошеннических действий можно изобразить следующей диаграммой:



Данную диаграмму можно рассматривать как объединение двух диаграмм, изображающих выполнение двух сеансов выполнения КП NS:

- в первом сеансе участвуют агенты a и b,
- а во втором – агенты b и c.

Действия агентов 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 КП NS (выдавая себя за a).
- Получив этот ШТ и дешифруя его, агент c предполагает, что этот ШТ был послан агентом a (о чём говорит ему первая компонента

соответствующего ОТ). Поэтому, согласно протоколу NS, в качестве второго действия агент c должен сгенерировать нонс r_c и послать ШТ $a^+(r_a, r_c)$ тому агенту, от которого он получил первый ШТ (думая, что он посылает этот ШТ агенту a).

- b пересылает полученный от c ШТ $a^+(r_a, r_c)$ агенту a.
- a рассматривает полученный от b ШТ как тот ШТ, который b должен послать ему в соответствии со вторым шагом КП NS, т.е. a рассматривает извлечённый из полученного ШТ нонс r_c как нонс, который был сгенерирован агентом b.
- Согласно третьему шагу NS, a посылает b ШТ $b^+(r_c)$.
- b извлекает из полученного ШТ нонс r_c, и выполняет

  $$b \to c : c^+(r_c)$$

  как третье действие во втором сеансе КП NS.

После этого c верит, что тот агент, с которым он выполнял этот сеанс КП NS, является агентом a (что неверно).

1.7 Другие примеры протоколов

1.7.1 Конфиденциальное вычисление суммы

Агенты $a_1, \ldots, a_n$ имеют числа $x_1, \ldots, x_n$ соответственно. Они хотят вычислить $\sum_{i=1}^{n} x_i$, причём каждый агент a_i не хочет, чтобы другим агентам стало известно его число x_i.

Один из протоколов для решения данной задачи имеет следующий вид:

- $a_1 \to a_2 : {a_2}^+(x_1 + r)$, где $r \underset{r}{\in} \mathbf{Z}$
  (для произвольного множества M запись $m \underset{r}{\in} M$ означает, что m является случайно выбранным элементом множества M, а если M – конечное множество то m предполагается равновероятно выбранным элементом множества M)
- $a_2 \to a_3 : {a_3}^+(x_2 + x_1 + r)$
- ...

- $a_n \to a_1 : {a_1}^+(x_n + \ldots + x_1 + r)$
- a_1 вычитает r из полученного результата.

Недостатком данного протокола является отсутствие контроля честности участников.

1.7.2 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,
- 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 КП.

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, утверждает 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.

(Упражнение для читателя – доказать, что данный протокол решает поставленную задачу.)

Уязвимость этого протокола – в отсутствии контроля честности участников.

1.7.3 Протокол с подтверждением приёма

Рассмотрим следующую ситуацию: агенты a и b используют общую АСШ для секретного обмена сообщениями, и для контроля правильности передачи каждое получаемое сообщение отсылается назад отправителю (чтобы отправитель был уверен, что сообщение получено в неискажённом виде), согласно следующему протоколу:

$$\begin{aligned} a &\to b : b^+a^-(m) \\ b &\to a : a^+b^-(m) \end{aligned} \tag{1.9}$$

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

К сожалению, данный КП уязвим к следующей атаке: активный противник e может перехватить первое сообщение и послать его b (от своего имени), в результате чего будут выполнены следующие действия:

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

т.к. при взаимодействии с e агент b действует согласно протоколу (1.9): получив сообщение m' $(= b^+a^-(m))$, он должен послать в ответ сообщение

$$e^+b^-e^+b^-(m') \ (= e^+b^-e^+a^-(m)). \tag{1.10}$$

Получив (1.10), e может извлечь из него m.

1.7.4 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 протоколов для решения данной задачи имеет следующий вид: a и b выбирают целое число n, и

- $a \to b : z - x + 1$, где $z := b^+(r)$, $r \underset{r}{\in} \{0,1\}^n$, r рассматривается как целое число в двоичной записи
- b генерирует простое число p размера примерно $n/2$, вычисляет

$$\{z_i := \left(b^-(z - x + i)\right)_p \mid i = 1, \ldots, 100\}$$

  и проверяет условие

$$\forall\, i = 1, \ldots, 100 \quad 0 < z_i < p$$
$$\forall\, i \neq j \quad |z_i - z_j| \geq 2$$

  (если это условие не выполняется, то b генерирует другое p и выполняет этот шаг еще раз с новым p)
- $b \to a :\ p,\ z_1, \ldots, z_y, z_{y+1} + 1, \ldots, z_{100} + 1,$
- $a \to b$: ответ $= (x \leq y)$, если $(r)_p = x$-й член этой последовательности.

Одна из уязвимостей данного протокола связана с тем, что в нём нет контроля честности участников.

Глава 2

Необходимые математические сведения

2.1 Теория групп

2.1.1 Определение группы

Группой называется множество G, на котором задана бинарная операция, т.е. правило, сопоставляющее каждой паре a, b элементов G некоторый элемент $ab \in G$, называемый **произведением** элементов a и b, причём выполнены следующие условия:

- операция **ассоциативна**, т.е. для всех $a, b, c \in G$

$$a(bc) = (ab)c$$

- существует элемент $e \in G$ (называемый **нейтральным элементом**), такой, что для каждого $a \in G$

$$ae = ea = a$$

- для каждого $a \in G$ существует элемент $a^{-1} \in G$, называемый **обратным к** a, такой, что

$$aa^{-1} = a^{-1}a = e$$

Группа G называется **коммутативной** (или **абелевой**) если для всех $a, b \in G \quad ab = ba$.

В группах имеют место **законы сокращения**:

- если $ab = ac$, то $b = c$, и
- если $ba = ca$, то $b = c$.

Действительно, если $ab = ac$, то, умножая обе части данного равенства слева на a^{-1}, получаем:

$$a^{-1}(ab) = a^{-1}(ac)$$

что по свойству ассоциативности равносильно равенству $(a^{-1}a)b = (a^{-1}a)c$, или $eb = ec$, или $b = c$. Аналогично доказывается другой закон сокращения.

Подгруппой группы G называется непустое подмножество $H \subseteq G$, удовлетворяющее следующим условиям:

$$\forall\, a, b \in H \quad ab \in H$$
$$\forall\, a \in H \quad a^{-1} \in H.$$

В частности, $e \in H$, т.к., беря произвольный элемент $a \in H$, на основании приведённых выше условий получаем: $aa^{-1} = e \in H$.

2.1.2 Смежные классы

Пусть заданы группа G и её подгруппа H.

Смежным классом по подгруппе H называется подмножество группы G, состоящее из произведений вида gh, где

- g – некоторый фиксированный элемент группы G, и
- h – произвольный элемент подгруппы H.

Данное множество обозначается символом gH.

Очевидно, что $g \in gH$ поскольку $e \in H$.

Если H состоит из конечного числа элементов, и список всех различных элементов H имеет вид $h_1, \ldots, h_k$, то все элементы смежного класса gH содержатся в списке

$$gh_1 \ \ldots \ gh_k \tag{2.1}$$

Все элементы списка (2.1) различны т.к. из $gh_i = gh_j$ по закону сокращения следует, что $h_i = h_j$.

Следовательно,

$$|gH| = |H| \tag{2.2}$$

(для каждого конечного множества A знакосочетание $|A|$ обозначает количество элементов в A)

Пусть g_1 и g_2 – произвольные элементы группы G. Докажем, что смежные классы g_1H и g_2H обладают следующим свойством: либо

$$g_1H \cap g_2H = \emptyset \tag{2.3}$$

либо

$$g_1H = g_2H \tag{2.4}$$

Если неверно (2.3), то существует элемент a, принадлежащий обоим классам, т.е.

- $a \in g_1H$, т.е. $a = g_1h_1$ для некоторого $h_1 \in H$, и
- $a \in g_2H$, т.е. $a = g_2h_2$ для некоторого $h_2 \in H$.

Таким образом, $a = g_1h_1 = g_2h_2$. Умножая равенство $g_1h_1 = g_2h_2$ справа на h_1^{-1}, получаем:

$$g_1 = g_2h_2h_1^{-1} \tag{2.5}$$

Произвольный элемент класса g_1H имеет вид g_1h для некоторого $h \in H$. Согласно (2.5), элемент g_1h равен произведению

$$g_2h_2h_1^{-1}h \tag{2.6}$$

т.е. элемент g_1h принадлежит g_2H.

Мы доказали включение $g_1H \subseteq g_2H$. Аналогично доказывается обратное включение. Таким образом, мы доказали, что если неверно (2.3), то имеет место (2.4).

2.1.3 Теорема Лагранжа

Пусть группа G состоит из конечного числа элементов, и список всех её элементов имеет вид

$$g_1 \ \dots \ g_n \tag{2.7}$$

Пусть H – произвольная подгруппа группы G.

Рассмотрим список всех смежных классов по H:

$$g_1H \ \dots \ g_nH \tag{2.8}$$

Очевидно, что объединение всех множеств из списка (2.8) совпадает с G, т.к. каждый элемент g_i из списка (2.7) принадлежит смежному классу g_iH.

Возможно, что некоторые смежные классы из списка (2.8) совпадают. В этом случае мы удалим из списка (2.8) лишние копии смежных классов, т.е. если $g_iH = g_jH$ при $i \neq j$, то один из данных смежных классов мы удаляем, и поступаем так до тех пор, пока все из оставшихся смежных классов не станут различными (т.е. среди них не будет двух совпадающих).

Пусть список оставшихся смежных классов имеет вид

$$g_{i_1}H \ \ldots \ g_{i_m}H \tag{2.9}$$

Очевидно, что

- объединение смежных классов из списка (2.9) по прежнему совпадает с G, и,
- как было установлено выше, все смежные классы из списка (2.9) попарно не пересекаются.

Следовательно,

$$|G| = |g_{i_1}H| \ + \ \ldots \ + \ |g_{i_m}H| \tag{2.10}$$

Из (2.10) и (2.2) следует, что

$$|G| = \underbrace{|H| \ + \ \ldots \ + \ |H|}_{m \text{ слагаемых}} = m|H|$$

Мы доказали **теорему Лагранжа**, которая утверждает, что

если G – конечная группа, и H – её подгруппа,
то $|G|$ делится на $|H|$

2.1.4 Циклические подгруппы

Пусть G – конечная группа, и g – элемент G, отличный от e.

Рассмотрим список элементов G следующего вида:

$$g^0, \ g^1, \ \ g^2, \ g^3, \ g^4, \ \ldots \tag{2.11}$$

где

$$\begin{aligned} g^0 &\stackrel{\text{def}}{=} e \\ g^1 &\stackrel{\text{def}}{=} g \\ g^2 &\stackrel{\text{def}}{=} gg \\ &\ldots \\ g^k &\stackrel{\text{def}}{=} \underbrace{gg\ldots g}_{k \text{ множителей}} \end{aligned}$$

Поскольку в группе G число элементов конечно, то список (2.11) не может быть бесконечным, и, следовательно, некоторые его элементы совпадают, т.е. существуют различные числа i, j, такие, что

$$g^i = g^j \tag{2.12}$$

Пусть например $i < j$.

Из (2.12) следует, что

$$g^{j-i} = e$$

Пусть k – наименьшее положительное целое число, удовлетворяющее условию

$$g^k = e$$

Очевидно, что совокупность элементов

$$e,\ g,\ \ g^2,\ \ldots,\ g^{k-1} \tag{2.13}$$

является подгруппой группы G.

Подгруппа (2.13) называется **циклической подгруппой**, порождённой элементом g.

Поскольку число элементов в (2.13) равно k, то, по теореме Лагранжа, $|G| = km$ для некоторого целого числа m.

Поскольку $g^k = e$, то $g^{|G|} = g^{km} = (g^k)^m = e^m = e$.

Таким образом, мы доказали, что

$$\begin{array}{l}\text{для любого элемента } g \text{ конечной группы } G \\ \text{имеет место соотношение} \quad g^{|G|} = e\end{array} \tag{2.14}$$

2.1.5 Гомоморфизмы групп

Пусть задана пара групп G_1, G_2.

Гомоморфизм из G_1 **в** G_2 – это произвольное отображение f вида

$$G_1 \xrightarrow{\ f\ } G_2 \tag{2.15}$$

такое, что для всех $a, b \in G_1 \quad f(ab) = f(a)f(b)$.

Пусть e_1 и e_2 – нейтральные элементы групп G_1 и G_2 соответственно. Докажем, что $f(e_1) = e_2$.

Действительно, в любой группе нейтральный элемент - это единственный элемент a, обладающий свойством $a = aa$, и, поскольку $e_1 = e_1e_1$,

то $f(e_1) = f(e_1)f(e_1)$, и, следовательно, $f(e_1)$ является нейтральным элементом G_2.

Ядро гомоморфизма f – это подмножество $Ker(f)$ группы G_1, определяемое следующим образом:

$$Ker(f) \stackrel{\text{def}}{=} \{a \in G_1 \mid f(a) = e_2\} \tag{2.16}$$

Нетрудно доказать, что $Ker(f)$ является подгруппой G_1.

Образ гомоморфизма f – это подмножество $Im(f)$ группы G_2, определяемое следующим образом:

$$Im(f) \stackrel{\text{def}}{=} \{b \in G_2 \mid b = f(a) \text{ для некоторого } a \in G_1\}$$

Нетрудно доказать, что $Im(f)$ является подгруппой G_2.

Пусть группа G_1 конечна, и список всех различных смежных классов группы G_1 по подгруппе $Ker(f)$ имеет вид

$$g_1Ker(f) \quad g_2Ker(f) \quad \ldots \quad g_mKer(f) \tag{2.17}$$

где $g_1, \ldots, g_m$ – некоторые элементы группы G_1.

Заметим, что для каждого $i \in \{1, \ldots, m\}$ все элементы смежного класса

$$g_iKer(f) \tag{2.18}$$

переходят при отображении f в один и тот же элемент группы G_2, т.к. произвольный элемент смежного класса (2.18) имеет вид g_ih для некоторого $h \in Ker(f)$, и, следовательно,

$$f(g_ih) = f(g_i)f(h) = f(g_i)e_2 = f(g_i)$$

В частности, поскольку каждый элемент из G_1 попадает в один из классов в списке (2.17), то все элементы $Im(f)$ содержатся в списке

$$f(g_1) \quad f(g_2) \quad \ldots \quad f(g_m) \tag{2.19}$$

Заметим, что все элементы в списке (2.19) различны, поскольку если $f(g_i) = f(g_j)$ для некоторой пары различных индексов i, j, то

$$f(g_i^{-1}g_i) = f(g_i^{-1})f(g_i) = f(g_i^{-1})f(g_j) = f(g_i^{-1}g_j) \tag{2.20}$$

Но левая часть в цепочке равенств (2.20) равна элементу $f(e_1)$, который, как было установлено выше, совпадает с e_2.

Таким образом, $f(g_i^{-1}g_j) = e_2$, т.е. $g_i^{-1}g_j = h \in Ker(f)$, т.е. $g_j = g_ih$, где $h \in Ker(f)$, т.е. смежные классы

$$g_jKer(f) \quad \text{и} \quad g_iKer(f)$$

имеют непустое пересечение, и, следовательно, совпадают, что невозможно, т.к. все смежные классы в списке (2.17) по предположению различны.

Таким образом

$$|Im(f)| \quad = \quad m \tag{2.21}$$

где m есть количество различных смежных классов по $Ker(f)$.

Из (2.21) и из теоремы Лагранжа следует, что

$$|G_1| = |Ker(f)||Im(f)| \tag{2.22}$$

В частности, имеет место импликация (которая будет использоваться ниже)

$$|Im(f)| > 1 \quad \Rightarrow \quad |Ker(f)| \leq \frac{1}{2}|G_1| \tag{2.23}$$

Гомоморфизм (2.15) называется **эпиморфизмом**, если отображение f является сюръективным, т.е. $Im(f) = G_2$.

Из (2.22) следует, что если f – эпиморфизм, то

$$|G_1| = |Ker(f)||G_2| \tag{2.24}$$

Гомоморфизм (2.15) называется **изоморфизмом**, если отображение f является взаимно-однозначным.

Если пара групп G_1, G_2 такова, что существует изоморфизм из G_1 в G_2, то мы будем считать группы G_1 и G_2 равными.

Наше желание считать изоморфные группы равными обосновывается тем, что при анализе алгебраических свойств группы природа её элементов не имеет никакого значения – важно лишь то, как на этих элементах действует операция произведения.

2.1.6 Прообразы элементов относительно гомоморфизмов

Пусть заданы

- пара групп G_1, G_2, и
- гомоморфизм f из G_1 в G_2.

Для каждого $g \in G_2$ символ $f^{-1}(g)$ обозначает множество

$$f^{-1}(g) \stackrel{\text{def}}{=} \{a \in G_1 \mid f(a) = g\} \tag{2.25}$$

Из данного определения следует, что множество (2.25)

- либо является пустым,
- либо является некоторым смежным классом по подгруппе $Ker(f)$.

$\forall\, g \in Im(f)$ запись $\overline{f^{-1}(g)}$ обозначает множество, определяемое следующим образом:

$$\overline{f^{-1}(g)} \stackrel{\text{def}}{=} \{a \in G_1 \mid f(a) \neq g\} \tag{2.26}$$

2.1.7 Декартово произведение групп

Пусть задан конечный список групп вида

$$G_1, \ldots, G_n \tag{2.27}$$

Декартовым произведением групп из списка (2.27) называется группа, обозначаемая записью

$$G_1 \times \ldots \times G_n \tag{2.28}$$

элементами которой являются списки вида

$$(g_1, \ldots, g_n) \tag{2.29}$$

где $g_1 \in G_1, \ldots, g_n \in G_n$.

Операция произведения на (2.28) определяется покомпонентно:

$$(g_1, \ldots, g_n)(g'_1, \ldots, g'_n) \stackrel{\text{def}}{=} (g_1 g'_1, \ldots, g_n g'_n)$$

т.е. $\forall\, i \in \{1, \ldots, n\}$ компоненты с индексом i перемножаются согласно операции произведения в группе G_i.

Нейтральным элементом группы (2.28) является список

$$(e_1, \ldots, e_n)$$

где $\forall\, i \in \{1, \ldots, n\}$ элемент e_i является нейтральным элементом группы G_i.

Обратным элементом к произвольному элементу (2.29) группы (2.28) является список вида

$$(g_1^{-1}, \ldots, g_n^{-1})$$

где для каждого $i \in \{1, \ldots, n\}$ элемент g_i^{-1} является обратным к g_i в группе G_i.

2.2 Группы, связанные с целыми числами

2.2.1 Группа целых чисел и её подгруппы

Множество целых чисел $\mathbf{Z}$ с операцией сложения является группой. Нейтральным элементом этой группы является число 0, и для каждого $a \in \mathbf{Z}$ обратным к a является число $-a$.

Нетрудно доказать, что для каждого целого числа $n > 0$ множество

$$n\mathbf{Z} \stackrel{\text{def}}{=} \{ni \mid i \in \mathbf{Z}\} \tag{2.30}$$

является подгруппой группы $\mathbf{Z}$.

Докажем, что любая подгруппа группы $\mathbf{Z}$ имеет вид (2.30).

Для этого сначала докажем **теорему о делении с остатком**, которая утверждает, что для произвольной пары a, n целых чисел, где $n > 0$, существуют единственная пара чисел q, r, удовлетворяющих следующим условиям:

$$a = nq + r \tag{2.31}$$

$$0 \leq r < n \tag{2.32}$$

Числа из множества (2.30) разбивают множество $\mathbf{Z}$ на отрезки длины n: каждый отрезок имеет вид

$$[ni, n(i+1)] \qquad (i \in \mathbf{Z}) \tag{2.33}$$

Для произвольного числа a существуют две возможности: либо

1. a является концом одного из отрезков вида (2.33), либо
2. a является внутренней точкой одного из отрезков вида (2.33).

В первом случае

$$a = nq \tag{2.34}$$

для некоторого $q \in \mathbf{Z}$, и мы полагаем r равным 0, а во втором –

$$nq < a < n(q+1) \tag{2.35}$$

для некоторого $q \in \mathbf{Z}$, и мы полагаем r равным разности

$$a - nq$$

Получаем, что в обоих случаях r удовлетворяет равенству (2.31) и неравенству (2.32).

Если q', r' – другая пара чисел, удовлетворяющих соотношениям

$$a = nq' + r' \tag{2.36}$$

$$0 \leq r' < n \tag{2.37}$$

то из (2.31) и (2.36) следует, что

$$n(q' - q) = r - r' \tag{2.38}$$

Поэтому

- если $q' = q$, то $r' = r$, и
- если $q' \neq q$, то пусть например $q' > q$, тогда левая часть (2.38) удовлетворяет соотношению

  $$n(q' - q) \geq n$$

  а правая –

  $$r - r' < n$$

  т.к. $r - r'$ есть расстояние между точками r и r', которые обе лежат в отрезке $[0, n-1]$, и мы получили противоречие.

Таким образом, пара чисел q, r, удовлетворяющих условиям (2.31) и (2.32), определена однозначно.

Теперь можно доказать, что произвольная подгруппа H группы $\mathbf{Z}$ имеет вид (2.30).

1. Если $H = \{0\}$, то $H = 0\mathbf{Z}$.
2. Если $H \neq \{0\}$, то H содержит ненулевое число a.

   Следовательно, H содержит положительное число, т.к. если $a < 0$, то $(-a) > 0$ и $(-a) \in H$.

   Определим n как наименьшее положительное число, принадлежащее H.

   Очевидно, что

   $$n\mathbf{Z} \subseteq H \tag{2.39}$$

   Если

   $$n\mathbf{Z} \neq H \tag{2.40}$$

   то существует число $a \in H$, такое, что $a \notin n\mathbf{Z}$, т.е. в разложении (2.31)

   $$r \neq 0 \tag{2.41}$$

Т.к. $r = a - nq = a + n(-q)$, то $r \in H$.

Но из (2.32) и из (2.41) следует, что r – положительное число, принадлежащее H, которое меньше чем n.

Это противоречит определению n.

Таким образом, предположение (2.40) ошибочно, и из (2.39) следует, что $H = n\mathbf{Z}$.

2.2.2 Представление наибольшего общего делителя

Пусть a, b – некоторая пара целых чисел, отличных от нуля.

Рассмотрим множество

$$a\mathbf{Z} \ + \ b\mathbf{Z} \stackrel{\text{def}}{=} \{ai + bj \mid i, j \in \mathbf{Z}\} \tag{2.42}$$

Нетрудно проверить, что данное множество является подгруппой в $\mathbf{Z}$.

Следовательно, по доказанному в предыдущей секции, существует целое положительное число d, такое, что подгруппа (2.42) имеет вид

$$d\mathbf{Z} \tag{2.43}$$

Поскольку числа a и b принадлежат (2.42), то следовательно они принадлежат и (2.43), т.е.

$$a = da_1 \quad \text{и} \quad b = db_1$$

т.е. d является делителем чисел a и b.

Докажем, что d является наибольшим общим делителем чисел a и b.

Пусть d' – какой-либо общий положительный делитель чисел a и b, т.е.

$$a = d'a'_1 \quad \text{и} \quad b = d'b'_1 \tag{2.44}$$

Поскольку d принадлежит множеству (2.43), то, следовательно, d принадлежит множеству (2.42), т.е. d имеет вид

$$d = ai + bj \tag{2.45}$$

для некоторых целых чисел i, j.

Подставим в (2.45) вместо чисел a и b их представления из (2.44), получим

$$d = d'a'_1 i + d'b'_1 j = d'k \tag{2.46}$$

где $k \stackrel{\text{def}}{=} a'_1 i + b'_1 j$.

Поскольку числа d и d' положительные, то число k тоже положительное.

Следовательно,

$$d = d'k \geq d'$$

т.е. d – наибольший общий делитель чисел a и b.

В частности, если a и b – взаимно простые числа, то существуют такие целые числа u и v, что

$$au + bv = 1 \tag{2.47}$$

Очевидно, что верно и обратное - если числа a и b таковы, что существуют целые числа u и v, удовлетворяющие условию (2.47), то числа a и b взаимно просты.

2.2.3 Группа вычетов по модулю n

Пусть n – некоторое положительное целое число.

Обозначим символом $\mathbf{Z}_n$ множество

$$\{0, 1, \ldots, n-1\}$$

Для каждого $a \in \mathbf{Z}$ обозначим записью $(a)_n$ то единственное r, которое удовлетворяет (2.31) и (2.32), т.е.

$$\exists\, q \in \mathbf{Z}: \quad a = nq + (a)_n, \quad 0 \leq (a)_n < n.$$

Докажем, что для всех $a, b \in \mathbf{Z}$

$$(a+b)_n = ((a)_n + (b)_n)_n \tag{2.48}$$

и

$$(ab)_n = ((a)_n (b)_n)_n \tag{2.49}$$

Из определения чисел вида $(a)_n$ следуют равенства

$$a = nq_1 + (a)_n \tag{2.50}$$

$$b = nq_2 + (b)_n \tag{2.51}$$

$$a + b = nq_3 + (a+b)_n \tag{2.52}$$

$$(a)_n + (b)_n = nq_4 + ((a)_n + (b)_n)_n \tag{2.53}$$

$$ab = nq_5 + (ab)_n \tag{2.54}$$

$$(a)_n (b)_n = nq_6 + ((a)_n (b)_n)_n \tag{2.55}$$

где $q_1, \ldots, q_6$ – некоторые целые числа.

Сложив (2.50), (2.51), (2.53), и сократив одинаковые слагаемые в обоих частях полученной суммы, получаем равенство

$$a + b = nq_1 + nq_2 + nq_4 + ((a)_n + (b)_n)_n \tag{2.56}$$

т.е.

$$a + b = nq_7 + ((a)_n + (b)_n)_n \tag{2.57}$$

Из (2.57) и (2.52) следует (2.48).

Далее, из (2.50) и (2.51) следует, что

$$(a - nq_1)(b - nq_2) = (a)_n(b)_n \tag{2.58}$$

т.е.

$$ab = nq_8 + (a)_n(b)_n \tag{2.59}$$

Сложив (2.59) с (2.55), и сократив одинаковое слагаемое в обоих частях полученной суммы, получаем равенство

$$ab = nq_9 + ((a)_n(b)_n)_n \tag{2.60}$$

Из (2.60) и (2.54) следует (2.49).

Определим операции

$$\underset{n}{+} \quad \text{и} \quad \underset{n}{\cdot} \tag{2.61}$$

на множестве $\mathbf{Z}_n$ следующим образом: для всех a, b из множества $\mathbf{Z}_n$

$$a \underset{n}{+} b \stackrel{\text{def}}{=} (a + b)_n, \quad a \underset{n}{\cdot} b \stackrel{\text{def}}{=} (ab)_n$$

В новых обозначениях соотношения (2.48) и (2.49) выглядят следующим образом:

$$(a + b)_n = (a)_n \underset{n}{+} (b)_n \tag{2.62}$$

и

$$(ab)_n = (a)_n \underset{n}{\cdot} (b)_n \tag{2.63}$$

Из соотношений (2.62) и (2.63) нетрудно вывести, что

- операции (2.61) ассоциативны и коммутативны, и
- операция $\underset{n}{\cdot}$ дистрибутивна относительно операции $\underset{n}{+}$.

Докажем например, что операция $\underset{n}{+}$ ассоциативна, т.е. для всех a, b, c из множества $\mathbf{Z}_n$ имеет место равенство

$$a \underset{n}{+} (b \underset{n}{+} c) = (a \underset{n}{+} b) \underset{n}{+} c$$

Т.к.

$$a = (a)_n, \quad b = (b)_n, \quad c = (c)_n$$

то

$$\begin{aligned}
& a \underset{n}{+} (b \underset{n}{+} c) = \\
= \; & (a)_n \underset{n}{+} ((b)_n \underset{n}{+} (c)_n) = \\
= \; & (a)_n \underset{n}{+} (b + c)_n = \\
= \; & (a + (b + c))_n = \\
= \; & ((a + b) + c)_n = \\
= \; & (a + b)_n \underset{n}{+} (c)_n = \\
= \; & ((a)_n \underset{n}{+} (b)_n) \underset{n}{+} (c)_n = \\
= \; & (a \underset{n}{+} b) \underset{n}{+} c
\end{aligned}$$

Другие упомянутые выше свойства операций (2.61) доказываются аналогично.

Ниже мы будем называть операции (2.61) соответственно сложением и умножением по модулю n (или просто сложением и умножением).

Будем использовать следующие обозначения:

- $\forall\, a, b \in \mathbf{Z}_n$ будем обозначать элементы

  $$a \underset{n}{+} b \quad \text{и} \quad a \underset{n}{\cdot} b$$

  записями $a + b$ и ab соответственно,

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

- запись $x \underset{n}{:=} e$, где x – переменная, и e – арифметическое выражение, обозначает действие, связанное с присваиванием переменной x значения выражения $(e)_n$.

Из вышесказанного вытекает, что

- операция сложения на множестве $\mathbf{Z}_n$ является ассоциативной

- число 0 является нейтральным элементом относительно операции сложения, и
- для каждого $a \in \mathbf{Z}_n$ число

$$(-a)_n = \begin{cases} n - a, & \text{если } a > 0 \\ 0, & \text{если } a = 0 \end{cases}$$

является обратным к a относительно операции сложения.

Таким образом, множество $\mathbf{Z}_n$ является группой относительно операции сложения.

Данная группа называется **группой вычетов по модулю** n.

2.2.4 Мультипликативная группа в $\mathbf{Z}_n$

Как было отмечено выше, операция умножения на множестве $\mathbf{Z}_n$ тоже является ассоциативной, и нетрудно доказать, что число 1 является нейтральным элементом относительно операции умножения.

Однако $\mathbf{Z}_n$ не является группой относительно операции умножения, т.к. не для каждого элемента $a \in \mathbf{Z}_n$ существует обратный элемент относительно операции умножения (например, для числа 0 не существет обратного относительно умножения).

Тем не менее, в $\mathbf{Z}_n$ существует подмножество, которое является группой относительно операции умножения. Данное подмножество обозначается записью $\mathbf{Z}_n^*$ и состоит из всех элементов $\mathbf{Z}_n$, которые взаимно просты с n.

Из рассуждений в конце в пункта 2.2.2 вытекает, что число $a \in \mathbf{Z}_n$ является взаимно простым с n тогда и только тогда, когда существуют целые числа u и v, такие, что

$$au + nv = 1 \tag{2.64}$$

Следовательно, $(au + nv)_n = (1)_n$. Учитывая, что

$$(a)_n = a, \quad (n)_n = 0, \quad (1)_n = 1$$

получаем, что в $\mathbf{Z}_n$ имеет место равенство

$$a(u)_n = 1 \tag{2.65}$$

т.е. если элемент a принадлежит подмножеству $\mathbf{Z}_n^*$, то к нему существует обратный элемент относительно операции умножения.

Нетрудно доказать, что верно и обратное: если к элементу $a \in \mathbf{Z}_n$ существует обратный относительно операции умножения, то a является взаимно простым с n.

Таким образом,

- для каждой пары a, b элементов $\mathbf{Z}_n^*$ их произведение ab тоже принадлежит $\mathbf{Z}_n^*$
- операция умножения на множестве $\mathbf{Z}_n^*$ ассоциативна
- число 1 принадлежит $\mathbf{Z}_n^*$ и является нейтральным элементом относительно операции умножения
- для каждого элемента $a \in \mathbf{Z}_n^*$ существует элемент $a^{-1} \in \mathbf{Z}_n^*$, такой, что
$$aa^{-1} = a^{-1}a = 1$$

т.е. $\mathbf{Z}_n^*$ является группой относительно операции умножения. Эта группа называется **мультипликативной группой в $\mathbf{Z}_n$**. Мы будем обозначать записью $\varphi(n)$ число элементов этой группы. Функция, сопоставляющая числу n число $\varphi(n)$, называется **функцией Эйлера**.

Нетрудно доказать, что

$$\varphi(n) = n(1 - \tfrac{1}{p_1}) \dots (1 - \tfrac{1}{p_k}) \tag{2.66}$$

где $p_1, \dots, p_k$ – различные простые делители n. Равенство (2.66) следует из

- равенства $\varphi(n) = \varphi(u)\varphi(v)$, если $n = uv$ и u и v взаимо просты (см. (2.92)), и
- легко проверяемого равенства $\varphi(p^k) = p^k - p^{k-1}$, где p – простое число.

Можно доказать, что если n – простое число, то группа $\mathbf{Z}_n^*$ является циклической. Произвольный элемент $g \in \mathbf{Z}_n^*$, такой, что $\mathbf{Z}_n^*$ совпадает с множеством степеней g, называется **примитивным** (или **порождающим**) элементом группы $\mathbf{Z}_n^*$.

2.2.5 Малая теорема Ферма

Пусть число n является простым. В этом случае

$$\mathbf{Z}_n^* = \{1, \dots, n-1\} \tag{2.67}$$

В частности,

$$|\mathbf{Z}_n^*| = n - 1 \tag{2.68}$$

Из (2.14) следует, что для любого a из множества (2.68) имеет место соотношение

$$a^{n-1} \underset{n}{=} 1 \tag{2.69}$$

Это утверждение называется **малой теоремой Ферма**.

2.2.6 Разложение группы $\mathbf{Z}_n$ в декартово произведение

Пусть число n являетя произведением вида

$$n = uv \tag{2.70}$$

где числа u и v взаимно просты.

Докажем, что в этом случае имеет место равенство

$$\mathbf{Z}_n = \mathbf{Z}_u \times \mathbf{Z}_v$$

Определим отображение

$$\mathbf{Z}_n \xrightarrow{f} \mathbf{Z}_u \times \mathbf{Z}_v \tag{2.71}$$

следующим образом: для каждого $a \in \mathbf{Z}_n$

$$f(a) \stackrel{\text{def}}{=} ((a)_u, (a)_v)$$

Докажем, что (2.71) является изоморфизмом, т.е.

- (2.71) является взаимно однозначным, и
- (2.71) сохраняет операцию сложения.

Из (2.70) следует, что количество элементов в множестве $\mathbf{Z}_n$ равно количеству элементов в множестве $\mathbf{Z}_u \times \mathbf{Z}_v$.

Поэтому для того, чтобы доказать взаимную однозначность отображения (2.71), достаточно доказать, что отображение (2.71) является инъективным, т.е. для всех $a, b \in \mathbf{Z}_n$ из

$$f(a) = f(b) \tag{2.72}$$

следует, что

$$a = b \tag{2.73}$$

Пусть верно (2.72), т.е.

$$(a)_u = (b)_u \tag{2.74}$$

и

$$(a)_v = (b)_v \tag{2.75}$$

(2.74) эквивалентно тому, что существует число q_1, такое, что

$$a - b = uq_1 \tag{2.76}$$

и (2.75) эквивалентно тому, что существует число q_2, такое, что

$$a - b = vq_2 \tag{2.77}$$

Из (2.76) и (2.77) следует, что

$$uq_1 = vq_2 \tag{2.78}$$

Поскольку u и v взаимно просты, то существуют такие целые числа i и j, что

$$ui + vj = 1 \tag{2.79}$$

Из (2.79) следует, что

$$uiq_1 + vjq_1 = q_1 \tag{2.80}$$

Из (2.78) и (2.80) следует, что

$$vq_2i + vjq_1 = q_1 \tag{2.81}$$

т.е.

$$vq_3 = q_1 \tag{2.82}$$

для некоторого целого q_3.

Из (2.76) и (2.82) следует, что

$$a - b = uvq_3 \tag{2.83}$$

т.е.

$$a - b = nq_3 \tag{2.84}$$

Поскольку a и b являются элементами $\mathbf{Z}_n$, то, следовательно, (2.84) возможно только в том случае, когда $q_3 = 0$, т.е. только тогда, когда a и b совпадают.

Теперь докажем, что (2.71) сохраняет операцию сложения, т.е. для всех $a, b \in \mathbf{Z}_n$

$$f(a \underset{n}{+} b) = f(a) + f(b)$$

т.е.

$$((a \underset{n}{+} b)_u, (a \underset{n}{+} b)_v) = ((a)_u, (a)_v) + ((b)_u, (b)_v)$$

т.е.

$$((a \underset{n}{+} b)_u, (a \underset{n}{+} b)_v) = ((a)_u \underset{u}{+} (b)_u, (a)_v \underset{v}{+} (b)_v)$$

т.е.

$$(a \underset{n}{+} b)_u = (a)_u \underset{u}{+} (b)_u \tag{2.85}$$

и

$$(a \underset{n}{+} b)_v = (a)_v \underset{v}{+} (b)_v \tag{2.86}$$

Доказательства истинности равенств (2.85) и (2.86) аналогичны, поэтому мы обоснуем лишь равенство (2.85).

Из (2.62) следует, что правая часть (2.85) равна $(a + b)_u$, т.е. (2.85) эквивалентно равенству

$$(a \underset{n}{+} b)_u = (a + b)_u \tag{2.87}$$

Имеют место равенства

$$a + b = nq_1 + (a \underset{n}{+} b) \tag{2.88}$$

и

$$(a \underset{n}{+} b) = uq_2 + (a \underset{n}{+} b)_u \tag{2.89}$$

где q_1 и q_2 – некоторые целые числа.

Складывая данные равенства почленно, и учитывая (2.70) получаем равенство

$$a + b = u(vq_1 + q_2) + (a \underset{n}{+} b)_u \tag{2.90}$$

Из (2.90) вытекает желаемое равенство (2.87).

Заметим, что аналогичным образом можно доказать, что (2.71) сохраняет также и операцию умножения, где умножение на $\mathbf{Z}_u \times \mathbf{Z}_v$ определяется покомпонентно:

$$(a, b)(a', b') \stackrel{\text{def}}{=} (a \underset{u}{\cdot} a', b \underset{v}{\cdot} b')$$

Нетрудно доказать, что

- пара $(1, 1)$ является нейтральным элементом относительно операции умножения на $\mathbf{Z}_u \times \mathbf{Z}_v$
- к элементу $(a, b) \in \mathbf{Z}_u \times \mathbf{Z}_v$ существует обратный элемент относительно данной операции умножения тогда и только тогда, когда

$$a \in \mathbf{Z}_u^* \quad \text{и} \quad b \in \mathbf{Z}_v^*$$

- $\forall\, a \in \mathbf{Z}_n^*$ для элемента a существует в $\mathbf{Z}_n^*$ обратный по умножению тогда и только тогда, когда его образ $f(a)$ обладает обратным по умножению (в $\mathbf{Z}_u \times \mathbf{Z}_v$).

Из сказанного выше следует, что

- $\{f(a) \mid a \in \mathbf{Z}_n^*\} = \mathbf{Z}_u^* \times \mathbf{Z}_v^*$, и
- сужение отображения f на подмножество $\mathbf{Z}_n^*$ является изоморфизмом из группы $\mathbf{Z}_n^*$ в группу $\mathbf{Z}_u^* \times \mathbf{Z}_v^*$ (относительно операции умножения).

Таким образом, в том случае, когда n является произведением двух взаимно простых чисел u и v, имеют место равенства

$$\mathbf{Z}_n = \mathbf{Z}_u \times \mathbf{Z}_v \tag{2.91}$$

и

$$\mathbf{Z}_n^* = \mathbf{Z}_u^* \times \mathbf{Z}_v^* \tag{2.92}$$

2.3 Автоматы

Автомат – это набор

$$M = (X, Q, Y, q^0, \delta, \lambda) \tag{2.93}$$

где X, Q и Y – множества, элементы которых называются соответственно **входными сигналами**, **состояниями**, и **выходными сигналами**, и

- $q^0 \in Q$ – **начальное состояние**,
- $\delta : Q \times X \to Q$ и $\lambda : Q \to Y$ – функции, называемые соответственно **функцией перехода** и **функцией выхода**.

Автомат является моделью динамической системы, работа которой происходит в дискретном времени, и заключается в

- изменении состояний под воздействием входных сигналов, поступающих на её вход, и
- выдаче в каждый момент времени $t \geq 0$ некоторого выходного сигнала.

В начальный момент времени ($t = 0$) автомат находится в начальном состоянии q^0.

В каждый момент времени $t = 0, 1, 2, \ldots$ автомат

- получает входной сигнал $x(t) \in X$,
- выдаёт выходной сигнал $y(t) \stackrel{\text{def}}{=} \lambda(q(t)) \in Y$

и в следующий момент времени $(t+1)$ переходит в состояние $q(t+1) \stackrel{\text{def}}{=} \delta(q(t), x(t))$.

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

Будем обозначать записью X^* множество конечных последовательностей (которые также называют **строками**) с компонентами из X, каждая последовательность $u \in X^*$ либо является пустой (и обозначается символом ε), либо имеет вид $x_1 \dots x_n$, где $x_1, \dots, x_n \in X$.

Пусть M – автомат вида (2.93). $\forall\, (q, u) \in Q \times X^*$ запись qu обозначает состояние, в которое перейдет автомат M из состояния q после поступления на его вход последовательности входных сигналов u, т.е.

$$qu = \begin{cases} q, & \text{если } u = \varepsilon, \\ \delta(q, x), & \text{если } u = x \in X, \\ (\dots((qx_1)x_2)\dots)x_n, & \text{если } u = x_1 \dots x_n. \end{cases}$$

Автомат (2.93) называется **автоматом без выхода**, если $Y = Q$ и $\lambda = id_Y$. Если (2.93) – автомат без выхода, то мы будем обозначать его четверкой (X, Q, q^0, δ).

Глава 3

Вероятностные алгоритмы

3.1 Вероятностный алгоритм проверки целых чисел на простоту

Многие криптографические алгоритмы связаны с использованием больших простых чисел. Как правило, такие числа порождаются следующим образом: случайным образом генерируется битовая строка размером в несколько сотен битов, которая рассматривается как двоичная запись целого числа, и это число проверяется на простоту. Известные на сегодня детерминированные алгоритмы проверки целых чисел на простоту имеют неприемлемо высокую вычислительную сложность, поэтому на практике применяются более эффективные вероятностные алгоритмы (которые хотя и могут дать ошибочный ответ, но вероятность такой ошибки может быть сделана сколь угодно малой). В настоящем параграфе мы излагаем один из таких алгоритмов.

3.2 Описание алгоритма

Пусть задано некоторое целое число $n \geq 2$.

Требуется определить, является ли n простым числом.

Для решения этой задачи предлагается нижеследующий алгоритм.

В данном алгоритме

- все числа интерпретируются как соответствующие им элементы $\mathbf{Z}_n$ (в частности, символ "-1", рассматриваемый, как элемент $\mathbf{Z}_n$, обозначает число $n-1$), и
- все операции понимаются как соответствующие операции в $\mathbf{Z}_n$.

Алгоритм состоит из последовательного выполнения излагаемых ниже шагов. Если на каком-либо шаге алгоритм определил, что n – составное, то после этого он сразу заканчивает работу, в противном случае происходит переход к следующему шагу.

1. Если n – чётное, то n – составное.

2. Если n имеет вид m^i для некоторых целых $m \geq 2$ и $i \geq 2$, то n – составное.

3. Представим число $n-1$ в виде $2^k l$, где l – нечётное число.

4. Выберем случайным образом число a из множества

$$\{1, \ldots, n-1\} \tag{3.1}$$

5. Вычислим следующие числа

$$\begin{aligned} b_0 &\stackrel{\text{def}}{=} a^l \\ b_1 &\stackrel{\text{def}}{=} b_0^2 \\ b_2 &\stackrel{\text{def}}{=} b_1^2 \\ &\ldots \\ b_k &\stackrel{\text{def}}{=} b_{k-1}^2 \end{aligned}$$

   Если для некоторого

$$j \in \{0, \ldots, k-1\}$$

   выполняются условия

$$\begin{cases} b_j \neq 1 \\ b_j \neq -1 \\ b_{j+1} = 1 \end{cases}$$

   то n – составное.

6. Если $b_k \neq 1$, то n – составное.

Если после шестого шага алгоритм не установил, что n – составное, то он объявляет n простым.

3.2.1 Анализ сложности алгоритма

Нетрудно доказать, что сложность данного влгоритма, т.е. количество исполняемых действий, можно оценить следующим выражением:

$$O((\log_2(n))^2) \tag{3.2}$$

Мы проанализируем лишь сложность шага 2.

Очевидно, что для выяснения того, представимо ли число n в виде m^i, достаточно перебрать значения для показателя i в диапазоне

$$\{2, \ldots, [\log_2 n]\}$$

и для каждого конкретного значения показателя i, поскольку функция $y = x^i$ монотонна при положительном значении аргумента, то уравнение

$$x^i = n \tag{3.3}$$

можно решать методом половинного деления, что требует не более $[\log_2 n]$ проверок для возможных кандидатов на решение данного уравнения.

Сложность проверки возможного кандидата x на решение уравнения (3.3) (т.е. сложность вычисления x^i) можно оценить например как $O(\log_2(i))$.

Такую сложность имеет например следующий рекурсивный алгоритм:

$$x^i := \begin{cases} x & \text{если } i = 1 \\ (x^2)^{\frac{i}{2}} & \text{если } i\text{–чётное} \\ x(x^2)^{\frac{i-1}{2}} & \text{если } i\text{–нечётное} \end{cases}$$

Таким образом, шаг 2 требует выполнения (3.2) действий.

3.2.2 Обоснование корректности алгоритма

Алгоритм может выдать ответ

$$n - \text{составное} \tag{3.4}$$

только на шагах 1,2,5 и 6.

Очевидно, что если ответ (3.4) был выдан на шагах 1 или 2, то n действительно является составным.

Докажем, что если ответ (3.4) был выдан на шагах 5 или 6, то n тоже действительно является составным.

Пусть ответ (3.4) был выдан на шаге 5, т.е. для некоторого $j \in \{0, \ldots, k-1\}$ выполняются условия

$$b_j \neq 1, \quad b_j \neq -1 \tag{3.5}$$

$$b_j^2 = 1 \tag{3.6}$$

Из (3.6) следует, что

$$(b_j - 1)(b_j + 1) = 0 \tag{3.7}$$

причём из (3.5) следует, что оба сомножителя в (3.7) отличны от нуля.

Если бы n было простым, то из (2.67) следует, что произведение отличных от нуля элементов $\mathbf{Z}_n$ не может быть равно нулю, т.е. (3.7) было бы невозможно.

Следовательно, если ответ был выдан на шаге 5, то он является правильным.

Теперь предположим, что ответ (3.4) был выдан на шаге 6.

Из определений чисел $b_0, \ldots, b_k$ следует, что

$$b_k = a^{2^k l} = a^{n-1}$$

Если бы n было простым, то, согласно малой теореме Ферма, имело бы место соотношение (2.69), т.е. $b_k = 1$, что противоречит нашему предположению.

Следовательно, если ответ был выдан на шаге 6, то он тоже является правильным.

Таким образом,

1. в том случае, когда n является простым, наш алгоритм никогда не выдаст ответ (3.4), т.е. в случае простого n ответ алгоритма всегда будет правильным

2. в том случае, когда n является составным, наш алгоритм иногда может выдать ошибочный ответ

   $$n \text{ – простое}$$

   и это произойдёт в том и только в том случае, когда на шагах 1,2,5 и 6 не был выдан ответ (3.4), т.е. имеют место следующие соотношения:

   $$n \text{ – нечётное число} \tag{3.8}$$

   $$n \text{ не имеет вид } m^i, \text{ где } m \geq 2 \text{ и } i \geq 2 \tag{3.9}$$

$$\left.\begin{array}{l}\text{для каждого } j \in \{0,\ldots,k-1\}\\ \text{выполнено одно из условий :}\\ b_j = 1 \quad \text{или}\\ b_j = -1 \quad \text{или}\\ b_{j+1} \neq 1\end{array}\right\} \tag{3.10}$$

$$b_k = 1 \tag{3.11}$$

3.2.3 Оценка вероятности ошибки

Из (3.8) и (3.9) следует, что в том случае, когда алгоритм выдаёт ошибочный ответ, число n должно обладать следующим свойством:

$$\begin{array}{l}\text{существуют некоторые взаимно простые}\\ \text{нечётные числа } u, v \text{ такие, что } n = uv\end{array} \tag{3.12}$$

Истинность или ложность соотношений (3.10) и (3.11) зависит только от выбора числа a, поскольку все числа из списка

$$b_0, b_1, \ldots, b_n$$

являются степенями числа a.

Назовём число из множества (3.1) **плохим**, если при выборе этого числа в качестве значения a выполняются соотношения (3.10) и (3.11).

Очевидно, что вероятность выдачи ошибочного ответа в рассматриваемом случае равна доле плохих a среди всех чисел из множества (3.1).

Заметим, что

$$\text{все плохие } a \text{ принадлежат множеству } \mathbf{Z}_n^* \tag{3.13}$$

т.к. если $a \notin \mathbf{Z}_n^*$, то a и n имеют общий делитель $d > 1$:

$$a = da_1$$
$$n = dn_1$$

и в этом случае не будет выполнено соотношение (3.11), т.к. если

$$b_k = a^{n-1} = 1$$

то

$$\begin{array}{rl} & n_1 a^{n-1} = n_1 \quad \Rightarrow\\ \Rightarrow & n_1 (da_1)^{n-1} = n_1 \quad \Rightarrow\\ \Rightarrow & n_1 d^{n-1} a_1^{n-1} = n_1 \quad \Rightarrow\\ \Rightarrow & n d^{n-2} a_1^{n-1} = n_1 \end{array}$$

но т.к. $n \underset{n}{=} 0$, то

$$nd^{n-2}a_1^{n-1} = 0$$

и мы получили противоречие.

(3.13) позволяет переписать соотношения (3.10) и (3.11) в другой форме, для чего мы введём следующие обозначения.

Пусть

- A – произвольная абелева группа, и
- m – некоторое положительное целое число.

Обозначим символом A^m совокупность элементов A вида

$$\{g^m \mid g \in A\}$$

Очевидно, что A^m является подгруппой в A.

Обозначим символом $\wedge m$ отображение вида

$$\wedge m : \ A \longrightarrow A^m \tag{3.14}$$

которое сопоставляет каждому $g \in A$ элемент g^m.

Очевидно, что $\wedge m$ является эпиморфизмом.

Рассмотрим следующую цепочку эпиморфизмов:

$$G \xrightarrow{\wedge l} G_0 \xrightarrow{\wedge 2} G_1 \xrightarrow{\wedge 2} \ldots \xrightarrow{\wedge 2} G_k$$

где

$$G \stackrel{\text{def}}{=} \mathbf{Z}_n^*$$
$$G_0 \stackrel{\text{def}}{=} G^l$$
$$G_1 \stackrel{\text{def}}{=} G_0^2$$
$$\ldots$$
$$G_k \stackrel{\text{def}}{=} G_{k-1}^2$$

Для каждого $j \in \{0, \ldots, k\}$ обозначим символом f_j сквозной эпиморфизм из G в G_j в данной цепочке, т.е.

$$f_j = \wedge(2^j l)$$

Очевидно, что

$$b_0 = f_0(a)$$
$$b_1 = f_1(a)$$
$$\ldots$$
$$b_k = f_k(a)$$

Поэтому

- соотношение (3.10) можно эквивалентным образом сформулировать так:

$$\left.\begin{array}{l}\text{для каждого } j \in \{0,\ldots,k-1\}\\ \text{выполнено одно из условий :}\\ f_j(a) = 1 \quad \text{или}\\ f_j(a) = -1 \quad \text{или}\\ f_{j+1}(a) \neq 1\end{array}\right\} \tag{3.15}$$

- соотношение (3.11) можно эквивалентным образом сформулировать так:

$$f_k(a) = 1 \tag{3.16}$$

В свою очередь,

- соотношение (3.15) можно эквивалентным образом сформулировать так:

$$\begin{array}{l}\text{для каждого } j \in \{0,\ldots,k-1\}\\ a \in f_j^{-1}(1) \cup f_j^{-1}(-1) \cup \overline{f_{j+1}^{-1}(1)}\end{array} \tag{3.17}$$

- соотношение (3.16) можно эквивалентным образом сформулировать так:

$$a \in f_k^{-1}(1) \tag{3.18}$$

(3.17) можно переписать так:

$$a \quad \in \quad \bigcap_{j=0}^{k-1} \quad (\ f_j^{-1}(1) \cup f_j^{-1}(-1) \cup \overline{f_{j+1}^{-1}(1)} \ ) \tag{3.19}$$

Из сказанного выше следует, что если для числа n алгоритм может выдать ошибочный ответ, то

- имеет место (3.12), и

- вероятность выдачи ошибочного ответа в рассматриваемом случае равна отношению числа элементов в пересечении множеств

$$\bigcap_{j=0}^{k-1} \quad (\ f_j^{-1}(1) \cup f_j^{-1}(-1) \cup \overline{f_{j+1}^{-1}(1)} \ ) \tag{3.20}$$

и

$$f_k^{-1}(1) \tag{3.21}$$

к числу элементов множества (3.1).

Для оценки вероятности выдачи ошибочного ответа мы отдельно рассмотрим случаи, когда

$$|G_k| \neq 1 \tag{3.22}$$

и

$$|G_k| = 1 \tag{3.23}$$

1. Пусть имеет место (3.22).

   Поскольку f_k является эпиморфизмом вида

   $$f_k : G \to G_k$$

   то, согласно (2.23), имеет место неравенство

   $$|f_k^{-1}(1)| = |Ker(f_k)| \leq \frac{1}{2}|G|$$

   Следовательно, число элементов в пересечении (3.20) и (3.21) также не превосходит числа

   $$\frac{1}{2}|G| \tag{3.24}$$

   Поскольку (3.24) не превосходит половины от числа элементов множества (3.1), то, следовательно, в случае (3.22) вероятность выдачи ошибочного ответа не превосходит $1/2$.

2. Теперь рассмотрим случай (3.23).

   Заметим, что группа G не является одноэлементным множеством, т.к. она содержит по меньшей мере два элемента -

   $$1 \quad \text{и} \quad -1 \tag{3.25}$$

   Кроме того, группа G_0 $(= G^l)$ также не является одноэлементным множеством, поскольку, ввиду того, что число l – нечётное, то

   $$(-1)^l = -1$$

   и, следовательно, группа G_0 также содержит по меньшей мере два элемента (3.25).

   Таким образом, существует номер

   $$j_0 \in \{0, \ldots, k-1\}$$

такой, что

$$|G_{j_0}| \neq 1 \tag{3.26}$$

и

$$|G_{j_0+1}| = 1 \tag{3.27}$$

Докажем, что для данного номера j_0 число элементов в множестве

$$f_{j_0}^{-1}(1) \cup f_{j_0}^{-1}(-1) \cup \overline{f_{j_0+1}^{-1}(1)} \tag{3.28}$$

не превосходит числа (3.24).

В частности, число элементов в пересечении (3.20) и (3.21) в этом случае также не будет превосходить числа (3.24), и, поскольку (3.24) не превосходит половины от числа элементов множества (3.1), то, следовательно, в случае (3.23) вероятность выдачи ошибочного ответа также не будет превосходить $1/2$.

Из (3.27) следует, что

$$\overline{f_{j_0+1}^{-1}(1)} \;=\; \emptyset$$

Поэтому для доказательства того, что число элементов в множестве (3.28) не превосходит числа (3.24), достаточно доказать неравенство

$$|f_{j_0}^{-1}(1)| \;+\; |f_{j_0}^{-1}(-1)| \;\leq\; \frac{1}{2}|G| \tag{3.29}$$

Введём следующие обозначения:

$$\begin{array}{ll}
U \stackrel{\text{def}}{=} \mathbf{Z}_u^* & V \stackrel{\text{def}}{=} \mathbf{Z}_v^* \\
U_0 \stackrel{\text{def}}{=} U^l & V_0 \stackrel{\text{def}}{=} V^l \\
U_1 \stackrel{\text{def}}{=} U_0^2 & V_1 \stackrel{\text{def}}{=} V_0^2 \\
\ldots & \ldots \\
U_k \stackrel{\text{def}}{=} U_{k-1}^2 & V_k \stackrel{\text{def}}{=} V_{k-1}^2
\end{array}$$

В данных обозначениях равенство (2.92) имеет вид

$$G = U \times V$$

и для каждого $j \in \{0, \ldots, k-1\}$ имеет место равенство

$$G_j = U_j \times V_j$$

поскольку возведение в степень пар из $U \times V$ осуществляется покомпонентно.

Согласно определению отображения (2.71), которое индуцирует изоморфизм

$$G \to U \times V$$

элементу -1 группы G при данном изоморфизме соответствует в декартовом произведении $U \times V$ пара

$$(-1, -1) \tag{3.30}$$

(левая компонента которой понимается как $r_u(-1)$, а правая - как $r_v(-1)$).

Заметим, что в обоих группах U и V элементы -1 не совпадают с элементами 1.

Рассмотрим следующие случаи:

(a) $|U_{j_0}| = 1$ или $|V_{j_0}| = 1$.
В этом случае пара (3.30) не может принадлежать декартову произведению

$$U_{j_0} \times V_{j_0}$$

поэтому элемент -1 группы G (соответствующий данной паре) не принадлежит группе G_{j_0}, т.е.

$$f_{j_0}^{-1}(-1) = \emptyset$$

и, следовательно, второе слагаемое в левой части неравенства (3.29) равно нулю.

Таким образом, в данном случае для доказательства неравенства (3.29) достаточно доказать неравенство

$$|f_{j_0}^{-1}(1)| \quad \leq \quad \frac{1}{2}|G| \tag{3.31}$$

Неравенство (3.31) следует из (2.23): поскольку f_{j_0} является эпиморфизмом вида

$$f_{j_0} : G \to G_{j_0}$$

и $|G_{j_0}| > 1$, то на основании (2.23) имеет место неравенство

$$|f_{j_0}^{-1}(1)| = |Ker(f_{j_0})| \leq \frac{1}{2}|G|$$

(b) $|U_{j_0}| > 1$ и $|V_{j_0}| > 1$.

В этом случае

$$|Im(f_{j_0})| \;=\; |G_{j_0}| \;\geq\; 4$$

и, следовательно, на основании (2.22) можно заключить, что

$$|Ker(f_{j_0})| \leq \frac{1}{4}|G| \tag{3.32}$$

Ввиду того, что

- $f_{j_0}^{-1}(1) = Ker(f_{j_0})$, и
- множество $f_{j_0}^{-1}(-1)$ является одним из смежных классов по подгруппе $Ker(f_{j_0})$, т.е., в частности,

$$|f_{j_0}^{-1}(-1)| = |Ker(f_{j_0})|$$

то

$$|f_{j_0}^{-1}(1)| \;+\; |f_{j_0}^{-1}(-1)| \;=\; 2|Ker(f_{j_0})| \tag{3.33}$$

Искомое неравенство (3.29) следует из (3.33) и (3.32).

3.2.4 Уменьшение вероятности ошибки

Предложенный алгоритм можно немного модицифировать так, чтобы вероятность ошибочного ответа существенно уменьшилась. Данная модификация заключается в том, что вместо однократного применения описанного выше алгоритма к числу n мы применим его к числу n несколько раз, и в качестве окончательного ответа будем выдавать

- ответ

  $$n \text{ – простое} \tag{3.34}$$

  если при всех применениях данного алгоритма к числу n был получен ответ (3.34), и

- ответ

  $$n \text{ – составное} \tag{3.35}$$

  если при хотя бы одном из применений данного алгоритма к числу n был получен ответ (3.35).

Нетрудно видеть, что окончательный ответ является ошибочным только тогда, когда при каждом из применений то число a, которое выбиралось на шаге 4, было плохим.

Пусть символ d обозначает количество применений алгоритма проверки на простоту к числу n. Вероятность ошибки можно оценить как вероятность того, что при d независимых выборах чисел

$$a_1, \ldots, a_d \tag{3.36}$$

из множества (3.1) все выбранные числа будут плохими.

Обозначим символом p вероятность того, что при однократном выборе числа a из множества (3.1) данное число будет плохим. Как было установлено выше, $p \leq 1/2$.

Поскольку все числа из списка (3.36) порождаются независимо друг от друга, то вероятность того, что все они одновременно будут плохими, равна p^d.

Таким образом, вероятность ошибки модифицированного алгоритма не превосходит числа 2^{-d}.

3.3 Доказательства с нулевым разглашением

3.3.1 Понятие доказательства с нулевым разглашением

Иногда какой-либо агент (a) хочет путем диалога с другим агентом (b) убедить его в том, что он знает некоторый секрет s, которым может быть, например,

- криптографический ключ, или
- знание решения вычислительно сложной задачи,

таким образом, чтобы агент b

- убедился в том, что a знает секрет,
- но при этом не смог бы извлечь из сообщений, которые он получает от a в процессе данного диалога, никакой информации об этом секрете.

Диалог между агентами a и b, обладающий указанными выше свойствами, называется **доказательством с нулевым разглашением (Д0Р)** знания секрета s.

Как правило, Д0Р представляет собой распределённый вероятностный алгоритм, и состоит из нескольких раундов обмена сообщениями между a и b, где каждый раунд имеет вид

- $b \to a$: вопрос
- $a \to b$: ответ
- b проверяет правильность полученного ответа,

и если a даёт правильный ответ на каждый вопрос, который задаёт ему b, то b принимает решение, что a знает секрет.

Вопросы, которые b задаёт a, как правило, представляют собой выбранные случайным образом элементы некоторого множества. Задача поиска ответов на вопросы должна обладать следующими свойствами:

- если a действительно знает секрет, то он может за полиномиальное время найти правильный ответ на каждый вопрос, который он получит от b, и
- если a не знает секрета, то вероятность того, что ему удастся найти правильный ответ на произвольный вопрос от b за полиномиальное время, должна быть ограничена некоторым числом q, где $0 \leq q < 1$.

Из второго условия следует, что a сможет найти правильные ответы во всех n раундах, в каждом из которых b генерирует свой вопрос независимо от других раундов, с вероятностью, не превышающей q^n. Путём подходящего выбора числа n раундов можно добиться того, чтобы a, не зная секрета, мог убедить b в том, что он знает секрет (т.е. смог найти правильные ответы на все заданные ему вопросы), со сколь угодно малой вероятностью.

3.3.2 ДОР знания изоморфизма графов

Пусть задан граф G, и N – множество его вершин. Для каждой перестановки ξ множества N (т.е. биективного отображения $\xi : N \to N$) запись $\xi(G)$ обозначает граф с множеством вершин N, множество рёбер которого определяется следующим образом: для любых $i, j \in N$

$$G \text{ содержит ребро } i \to j \text{ тогда и только тогда, когда}$$
$$\xi(G) \text{ содержит ребро } \xi(i) \to \xi(j).$$

Будем называть графы G_1 и G_2 **изоморфными**, если $G_1 = \xi(G_2)$ для некоторой перестановки ξ. Запись $G_1 \sim G_2$ обозначает утверждение о том, что G_1 и G_2 изоморфны.

Если агент a знает доказательство того, что $G_1 \sim G_2$ (т.е. a знает перестановку σ, такую, что $G_1 = \sigma(G_2)$), то a может доказать b то, что

он знает такую перестановку σ при помощи Д0Р, раунд которого имеет следующий вид:

- $a \to b : \xi(G_1)$, где ξ – случайным образом порождённая перестановка на множестве вершин G_1,
- $b \to a : i \underset{r}{\in} \{1, 2\}$,
- $a \to b$: перестановка, доказывающая $\xi(G_1) \sim G_i$
  т.е. $a \to b : [\![i = 1]\!]\ \xi : \xi \circ \sigma$.

3.3.3 Д0Р знания существования гамильтонова цикла в графе

Гамильтонов цикл (ГЦ) в графе – это цикл в этом графе (т.е. путь, начало которого совпадает с его концом), который проходит через каждую вершину ровно один раз.

Если a знает некоторый ГЦ в графе G, то a может убедить в этом b при помощи Д0Р, раунд которого имеет следующий вид:

- a случайным образом порождает перестановку ξ на множестве вершин графа G,
- $a \to b : H$, где H – матрица, каждый элемент которой представляет собой результат вероятностного шифрования соответствующего элемента матрицы инцидентности графа $\xi(G)$ (который равен 0 или 1),
- $b \to a : i \underset{r}{\in} \{1, 2\}$,
- $a \to b : [\![i = 1]\!]\ \xi : \sigma$, где σ – список элементов матрицы H, результат расшифровки которых представляет собой ГЦ в графе $\xi(G)$.

3.3.4 Общая схема Д0Р

Приведённые выше Д0Р можно рассматривать как частные случаи общей схемы, позволяющей доказывать с нулевым разглашением знание решения некоторого класса задач.

Предполагается, что на множестве задач $\mathcal{P}$ из этого класса задана некоторая совокупность Ξ преобразований, обладающих следующими свойствами.

- Каждое преобразование $\xi : \mathcal{P} \to \mathcal{P}$ из множества Ξ сопоставляет каждой задаче $p \in \mathcal{P}$ задачу $\xi(p) \in \mathcal{P}$, изоморфную (в некотором смысле) задаче p.
- Если какой-либо агент знает решение σ задачи $p \in \mathcal{P}$, то $\forall \xi \in \Xi$ данный агент знает решение задачи $\xi(p)$. Будем обозначать это решение записью $\xi(\sigma)$.

Если агент a знает решение σ задачи p, то a может доказать агенту b то, что он знает σ, при помощи Д0Р, раунд которого имеет следующий вид:

- $a \to b : (\xi(p), h(\xi(\sigma)))$, где ξ – выбранное случайным образом преобразование из Ξ, h – ХФ,
- $b \to a : i \underset{r}{\in} \{1, 2\}$,
- $a \to b : [\![i = 1]\!]\ \xi : \xi(\sigma)$.

3.3.5 Д0Р знания дискретного логарифма

Пусть p – простое число, и $g, h \in \mathbf{Z}_p$. Ниже все вычисления и сравнения производятся по mod p.

Если агент a знает число $x \in \mathbf{Z}^*_{p-1}$, такое, что $g^x = h$, то он может доказать другому агенту b знание такого x при помощи следующего протокола:

- $a \to b : \{h_i := g^{x_i} \mid i = 1, \ldots, n\}$, где $x_1, \ldots, x_n \underset{r}{\in} \mathbf{Z}^*_{p-1}$,
- a и b совместно генерируют случайные биты $e_1, \ldots, e_n$,
- $i_1 :=$ min число из $\{1, \ldots, n\}$ со свойством $e_{i_1} = 1$,
- $a \to b : \{[\![e_i = 0]\!]\ x_i : y_i \mid i = 1, \ldots, n\}$, где
$$\forall\, i = 1, \ldots, n \quad y_i := (x_i - x_{i_1})_{p-1},$$
- $\forall\ i = 1, \ldots, n$ агент b проверяет истинность выражения
$$[\![e_i = 0]\!]\ (g^{x_i} = h_i) : (g^{y_i} = h_i h_{i_1}^{-1})$$
- $a \to b : z := (x - x_{i_1})_{p-1}$
- b принимает ответ, если $g^z = h h_{i_1}^{-1}$.

Можно доказать, что данный протокол является Д0Р, и если a не знает x со свойством $g^x = h$, то вероятность того, что b примет ответ, не превосходит 2^{-n}.

Глава 4

Криптографические примитивы

Криптографические примитивы – это математические конструкции и алгоритмы, которые

- используются в качестве элементарных компонентов при построении протоколов, и
- предназначены для обеспечения различных свойств безопасности протоколов.

К криптографическим примитивам относятся системы шифрования, хэш-функции, схемы разделения секрета, и др.

4.1 Системы шифрования

4.1.1 Симметричные системы шифрования

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

Блочные системы шифрования

В **блочных СШ** открытый текст перед шифрованием разбивается на блоки, 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:

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

Поточные системы шифрования

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** (называемые ниже просто **регистрами**). Регистр – это автономный автомат,

- состояниями которого являются битовые вектора из $\{0,1\}^n$, где n – фиксированное число,

- функция переходов $\delta : \{0,1\}^n \to \{0,1\}^n$ сопоставляет каждому состоянию $(q_1, \ldots, q_n) \in \{0,1\}^n$ состояние
$$(q_2, \ \ldots, \ q_n, \ \sum_{i=1}^{n} c_i q_i) \tag{4.1}$$
где $c_1, \ldots, c_n$ – фиксированные элементы $\{0,1\}$, операции выполняются по $\bmod 2$, и

- выходными сигналами которого являются 0 и 1.

Гамма, порождаемая регистром, представляет собой последовательность его выходных сигналов (которые он выдает в моменты 0, 1, и т.д.).

Для шифрования или дешифрования при помощи регистра его начальное состояние полагается равным ключу.

Существуют и другие способы порождения гаммы:

1. один из них заключается в использовании двух регистров: 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 $b_i = 0$,

2. другой способ заключается в том, что вместо регистров 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, $b \in R^n$.

4.1.2 Асимметричные системы шифрования

Каждая асимметричная СШ предполагает использование пары ключей, связанной с каким-либо агентом, причем

- ключ для шифрования известен всем агентам, и
- ключ для дешифрования должен быть известен только тому агенту, с которым он связан.

Данные ключи обозначаются записями a^+ и a^- соответственно, где a – идентификатор агента, с которым связаны эти ключи.

Система шифрования RSA

Название системы шифрования RSA является аббревиатурой, связанной с фамилиями её создателей (Rivest, Shamir и Adleman). Криптографическая стойкость данной СШ (т.е. сложность нахождения по ШТ соответствующих им ОТ 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, удовлетворяющих условиям:

- p и q примерно одинаковы по размеру,
- НОД $(p-1, q-1)$ – небольшое число.

Ключи a^+ и a^- имеют следующий вид:

- $a^+ := \{n, e\}$, где $n := pq, \quad e \underset{r}{\in} \mathbf{Z}^*_{\varphi(n)}$.
- $a^- := \{d, p, q\}$, где $ed \underset{\varphi(n)}{=} 1$.

Для шифрования ОТ разбивается на блоки, каждый из которых можно рассматривать как двоичную запись числа из $\mathbf{Z}_n$. Каждый из этих блоков шифруется отдельно.

Ниже вычисления и сравнения выполняются по $\bmod n$.

- Шифрование: $a^+(m) \stackrel{\text{def}}{=} m^e$.
- Дешифрование: $a^-(u) \stackrel{\text{def}}{=} u^d$.

Докажем, что определенная таким образом СШ обладает свойством (1.1), т.е. $\forall\, x \in \mathbf{Z}_n$ верно равенство

$$(x^e)^d = x. \tag{4.2}$$

Т.к. $ed \underset{\varphi(n)}{=} 1$, то $\exists\, t \in \mathbf{Z} : ed = t\varphi(n) + 1$.

- Если $x \in \mathbf{Z}^*_n$, то $x^{\varphi(n)} = 1$ (т.к. $|\mathbf{Z}^*_n| = \varphi(n)$), поэтому

  $$(x^e)^d = x^{ed} = x^{t\varphi(n)+1} = (x^{\varphi(n)})^t x = 1^t x = x.$$

- Если $x \in \mathbf{Z}_n \setminus \mathbf{Z}^*_n$, то такой x имеет вид

  - pi (где $i = 0, \ldots, q-1$), или
  - qi (где $i = 0, \ldots, p-1$).

  Расмотрим, например, первый случай, т.е. $x = pi$, где $i = 0, \ldots, q-1$.

  Если $i = 0$, то $x = 0$, и равенство (4.2) верно.

  Если $i \neq 0$, то число pi взаимно просто с q. Разделим pi с остатком на q:

  $$pi = qj + r, \qquad \text{где } r \in \mathbf{Z}^*_q$$

  Порядок группы $\mathbf{Z}^*_q$ равен $q - 1$, поэтому

  $$r^{q-1} \underset{q}{=} 1$$

т.е. $(pi - qj)^{q-1} - 1 \underset{q}{=} 0$. Раскрывая $(pi - qj)^{q-1}$ по формуле бинома, получаем:

$$\begin{aligned}
&(pi)^{q-1} - 1 \underset{q}{=} 0 && \Rightarrow \\
&(pi)^{q-1} \underset{q}{=} 1 && \Rightarrow \\
&(pi)^{t(p-1)(q-1)} \underset{q}{=} 1 && \Rightarrow \\
&(pi)^{t\varphi(n)} \underset{q}{=} 1 && \Rightarrow \\
&((pi)^{t\varphi(n)} - 1) \underset{q}{=} 0 && \Rightarrow \\
&((pi)^{t\varphi(n)} - 1) = qk && \Rightarrow \\
&(pi)((pi)^{t\varphi(n)} - 1) = pqik && \Rightarrow \\
&x(x^{t\varphi(n)} - 1) \underset{n}{=} 0
\end{aligned}$$

откуда следует (4.2). ■

Нетрудно видеть, что СШ RSA обладает свойством (1.2).

Система шифрования Эль-Гамаля

Другим примером АСШ 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 $g \in \mathbf{Z}_p^*$
  (т.е. такой g, который порождает группу $\mathbf{Z}_p^*$).

Ниже все вычисления и сравнения - по mod p.
Ключи a^+ и a^- имеют следующий вид:

$$a^- := x \underset{r}{\in} \mathbf{Z}_p^*, \quad a^+ := \{p, g, y\}, \quad \text{где } y := g^x.$$

Для шифрования ОТ разбивается на блоки, каждый из которых можно рассматривать как двоичную запись числа из $\mathbf{Z}_p$. Каждый из этих блоков шифруется отдельно.

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

4.2 Хэш-функции

4.2.1 Пример построения хэш–функции

ХФ может быть построена на основе одношаговой сжимающей функции

$$\delta : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$$

Вычисление $h(x)$ производится следующим образом:

- строка x дополняется до размера, кратного n, и разбивается на блоки длины n:
$$x =: (x_1, \ldots, x_k)$$
- вычисляются строки $q_0, \ldots, q_k$, где
  - q_0 – некоторая фиксированная строка, и
  - $q_i := \delta(x_i, q_{i-1}) \quad (i = 1, \ldots, k)$
- $h(x) := q_k$.

Пример хорошей δ: $\delta(x_1, x_2) := x_1(x_2) \oplus x_2$, где $x_1(x_2)$ – ШТ, получаемый шифрованием ОТ x_2 на ключе x_1.

Пример плохой δ: $\delta(x_1, x_2) := k(x_1 \oplus x_2)$, где $k(x_1 \oplus x_2)$ – ШТ, получаемый шифрованием ОТ $x_1 \oplus x_2$ на ключе k.

4.2.2 Стандарт хэш–функции SHS

В этом пункте мы рассмотрим американский стандарт ХФ SHS (Secure Hash Standard), он был разработан в 1993 г. Этот стандарт основан на алгоритме MD4 Ривеста. ХФ, построенная по данному стандарту, преобразует строки длины $\leq 2^{64}$ в строки длины 160.

Вычисление значения $h(a_1, \ldots, a_n)$ производится следующим образом. Сначала строка $(a_1, \ldots, a_n)$ дополняется до строки u, размер которой делится на 512, и которая имеет вид

$$u = (a_1, \ldots, a_n, 1, 0, \ldots, 0, l_1, \ldots, l_{64})$$

где $(l_1, \ldots, l_{64})$ – двоичная запись n. Разобьем строку u на блоки $u_1, \ldots, u_k$ длины 512. Искомое значение $h(a_1, \ldots, a_n)$ равно состоянию, в которое перейдёт автомат без выхода (U, Q, q_0, δ) после поступления на его вход последовательности $u_1, \ldots, u_k$, где

- $U = \{0,1\}^{512}$ – множество входных сигналов

- $Q = \{0,1\}^{160}$ – множество состояний
- $q_0 \in Q$ - начальное состояние
- $\delta : Q \times U \to Q$ – отображение переходов

Отображение переходов данного автомата определяется следующим образом:

$$\delta(q, x) := q + g(q, x)$$

где сложение понимается следующим образом: слагаемые разбиваются на 5 слов $\in \{0,1\}^{32}$, и соответствующие слова складываются по mod 2^{32}. Выражение $g(q, x)$ обозначает состояние, в которое перейдёт автомат

$$(U', Q', q, \delta'), \quad \text{где} \begin{cases} U' = \{0,1\}^{32} \\ Q' = (\{0,1\}^{32})^5 \\ q - \text{начальное состояние} \\ \delta' : Q' \times U' \to Q' - \\ \quad \text{отображение переходов} \end{cases}$$

после поступления на его вход последовательности $(v_0, \ldots, v_{79})$, которая определяется следующим образом:

- последовательность $(v_0, \ldots, v_{15})$ представляет собой разбиение x на 16 слов длины 32
- для каждого $i = 16, \ldots, 79$

$$v_i := v_{i-3} \oplus v_{i-8} \oplus v_{i-14} \oplus v_{i-16}$$

Отображение переходов δ' определяется следующим образом. Пусть

- текущее состояние автомата имеет вид

$$(a_i, b_i, c_i, d_i, e_i) \in (\{0,1\}^{32})^5$$

- входной сигнал в текущий момент равен v_i.

Тогда состояние $(a_{i+1}, b_{i+1}, c_{i+1}, d_{i+1}, e_{i+1})$ данного автомата в следующий момент времени определяется следующим образом:

- $a_{i+1} := \Big(T^5(a_i) + f_i(b_i, c_i, d_i) + e_i + v_i + z_i\Big)_{2^{32}}$

- $b_{i+1} := a_i$
- $c_{i+1} := T^{30}(b_i)$
- $d_{i+1} := c_i$
- $e_{i+1} := d_i$

где

- T – циклический сдвиг влево на 1 бит
- $$f_i(b,c,d) = \\ = \begin{cases} (b \wedge c) \vee (\overline{b} \wedge d) & (i = 0, \ldots, 19) \\ b \oplus c \oplus d & (i = 20, \ldots, 39, 60, \ldots, 79) \\ (b \wedge c) \vee (b \wedge d) \vee (c \wedge d) & (i = 40, \ldots, 59) \end{cases}$$
  где $\vee$ и $\wedge$ – побитовые дизъюнкция и конъюнкция векторов, и $\overline{b}$ – побитовое инвертирование
- $z_0, \ldots, z_{79}$ – 32-битовые слова, которые имеют один и тот же вид для
  - $i = 0, \ldots, 19$
  - $i = 20, \ldots, 39$
  - $i = 40, \ldots, 59$
  - $i = 60, \ldots, 79$

4.3 Схемы разделения секрета

4.3.1 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,

- s может представлять собой вектор $s \in \{0,1\}^l$, и
- долями являются вектора $s_1, \ldots, s_n$ из $\{0,1\}^l$, такие, что выполняется соотношение
$$s = s_1 \oplus \ldots \oplus s_n$$

В данном случае 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.

4.3.2 (n, k)-пороговая СРС

Наиболее часто используются (n, k)-пороговые СРС (где $1 < k \leq n$), в которых секрет s распределяется между n агентами таким образом, что

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

Примеры (n, k)-пороговых СРС:

1. **СРС Шамира**.

   Пусть секрет s можно представить в виде элемента некоторого поля P. Для распределения секрета среди n агентов $a_1, \ldots, a_n$ выбирается многочлен $f \in P[x]$ степени $k - 1$:

   $$f = c_0 + c_1 x + \ldots + c_{k-1} x^{k-1}$$

   в котором $c_0 = s$.

   Для каждого из n агентов выбирается несекретный элемент $x_i \in P \setminus \{0\}$, причём элементы $x_1, \ldots, x_n$ различны.

   Доля агента a_i имеет вид $f(x_i) \quad (i = 1, \ldots, n)$.

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

   $$s = c_0 = f(0) = \sum_{i=1}^{k} f(y_i) \prod_{j \neq i} \frac{y_j}{y_j - y_i}$$

Если доли вычисляются и распределяются агентом s, которому агенты $a_1, \ldots, a_n$ не доверяют и хотят проверять правильность своих долей, то в том случае, когда $P = \mathbf{Z}_p$, это можно сделать следующим образом. Выбирается несекретный элемент $g \in P$, и выполняется

$$s \to \{a_1, \ldots, a_n\} : \{d_i := g^{c_i} \mid i = 0, \ldots, k-1\}.$$

Агент a_i проверяет правильность своей доли z_i следующим образом:

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

2. **СРС Blakley**.

   В этой СРС секрет s является элементом $\mathbf{R}^k$ (где $\mathbf{R}$ – множество действительных чисел).

   Каждая доля представляет собой уравнение $(k-1)$–мерной гиперплоскости, содержащей s.

   $\{s\}$ = пересечение любых k долей.

3. **СРС Карнина-Грини-Хеллмана**.

   В этой СРС секрет s является элементом $\mathbf{R}$.

   Выбираются вектора $\vec{v}_0, \ldots, \vec{v}_n \in \mathbf{R}^k$, такие, что ранг матрицы из любых k этих векторов равен k. Например,

   $$\vec{v}_i = (1, r_i, r_i^2, \ldots, r_i^{k-1})$$

   где $r_0, \ldots, r_n$ – различные действительные числа.

   Секрет s имеет вид

   $$s := \vec{v}_0 u^{\downarrow}, \qquad \text{где } u^{\downarrow} \underset{r}{\in} \mathbf{R}^k$$

   Доля i–го агента $(i = 1, \ldots, n)$ имеет вид $\vec{v}_i u^{\downarrow}$. Каждые k долей порождают систему линейных уравнений размера $k \times k$, неизвестными в которой являются коэффициенты вектора $u^{\downarrow}$.

4.3.3 Схемы разделения секрета с двумя группами

Предполагается, что совокупность агентов состоит из двух групп, и для вычисления секрета s нужно

- k_1 любых долей агентов из первой группы, и

- k_2 любых долей агентов из второй группы.

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

- для распределения секрета выбирается пара многочленов $f_1, f_2 \in P[x]$ степени $k_1 - 1$ и $k_2 - 1$ соответственно, причём $s = f_1(0)f_2(0)$,
- для каждого из n_i агентов i–й группы $(i = 1, 2)$ выбирается несекретный элемент $x_j^{(i)} \in P \setminus \{0\}$, причём элементы $x_1^{(i)}, \ldots, x_{n_i}^{(i)}$ различны,
- доли агентов i–й группы $(i = 1, 2)$ имеют вид $f_i(x_j^{(i)})$.

Многочлен, по которому вычисляется секрет, равен произведению $f_1 f_2$.

4.4 Метки времени

Иногда к передаваемым сообщениям добавляются **метки времени (МВ)**. Это делается

- для повышения доверия к подлинности получаемых сообщений и их источников, а также
- для противодействия атакам, основанным на повторной передаче уже переданных сообщений.

МВ в сообщениях могут быть ложными или поддельными. Поэтому в некоторых КП предусмотрено проставление МВ не самим отправителем, а доверенным посредником s. Например, проставление МВ на сообщение m может делаться следующим образом.

$$a \to s : m$$
$$s \to a : \langle m, t \rangle_s$$

Однако такой способ проставления МВ не помогает в том случае, когда s по сговору с a может проставлять ложные МВ. Некоторое противодействие атакам подобного типа может оказывать нижеследующий КП, в котором проставляемые МВ связываются друг с другом. Если агенту a необходимо проставить МВ на сообщение m, то он выполняет следующие действия.

$$a \to s : h(m), a$$
$$s \to a : \langle a, n, h_n, t_n, a_{n-1}, h_{n-1}, t_{n-1}, H_n \rangle_s$$

где h – ХФ, n – порядковый номер проставляемой МВ, и

- $h_n := h(m)$
- t_n – проставленная МВ
- a_{n-1} – агент, которому s проставил предыдущую МВ
- h_{n-1} – значение ХФ h на сообщении агента a_{n-1}
- t_{n-1} – предыдущая МВ
- $H_n := h(a_{n-1}, h_{n-1}, t_{n-1}, H_{n-1})$

После того, как s проставит $n+1$–ю МВ, он выполняет действие

$$s \to a : a_{n+1}$$

где a_{n+1} – следующий после a агент, обратившийся к s за проставлением МВ.

Если правильность МВ t_n вызывает сомнения, то

- a связывается с a_{n-1} и a_{n+1} для получения информации, подтверждающей правильность t_n,
- если правильность и этой информации подвергается сомнению, то a связывается с a_{n-2} и a_{n+2}, и т.д.

Можно усилить предыдущий КП, связывая текущую проставляемую МВ t_n с k предыдущими и k последующими МВ. В этом КП агент a_n хранит МВ $t_{n\pm i}$, где $i = 1, \ldots, k$.

В случае отсутствия доверенного посредника s для проставления МВ могут привлекаться обычные агенты. В этом случае проставление МВ на сообщении m агента a может иметь следующий вид:

- a выбирает случайным образом некоторое множество агентов $a_1, \ldots, a_k$, и
- посылает каждому из них $h(m)$ с просьбой прислать ему $\langle h(m), t_i \rangle_{a_i}$, где t_i – показания часов агента a_i в текущий момент времени.

МВ сообщения m в этом случае имеет вид

$$\Big(\langle h(m), t_1 \rangle_{a_1}, \ldots, \langle h(m), t_k \rangle_{a_k} \Big).$$

4.5 Закрытая передача

Задача закрытой передачи сообщения m от агенту a агенту b заключается в том, что a должен передать b сообщение, которое мы будем обозначать записью $[m]^r$, где r – сообщение, называемое **маской**, причем должны быть выполнены следующие условия:

- алгоритм вычисления $[m]^r$ по паре (m, r) общеизвестен и имеет полиномиальную сложность
- нельзя найти $m' \neq m$, такое, что $\exists\, r' : [m']^{r'} = [m]^r$, за полиномиальное время
- нельзя извлечь m из $[m]^r$ за полиномиальное время.

Если m – большая строка, то $[m]^r$ можно определить, например, как $h(m, r)$, где h – ХФ.

Если m состоит из одного бита, то $[m]^r$ можно определить другим образом. В данном случае сообщение $[m]^r$ называется **блобом**, а процедура передачи блоба $[m]^r$ от a к b называется **вручением бита** (**bit committment**) и состоит из следующих действий:

- a и b выбирают простое p, и примитивный элемент $g \in \mathbf{Z}_p^*$, все вычисления выполняются по mod p
- $b \to a : x \underset{r}{\in} \mathbf{Z}_p^*$
- $a \to b : [b]^{x,y} := x^b g^y$, где $y \underset{r}{\in} \mathbf{Z}_{p-1}$.

Если агент a умеет $\forall\, x \underset{r}{\in} \mathbf{Z}_p^*$ раскрывать блоб $[b]^{x,y}$ заданным образом за полиномиальное время, т.е. умеет вычислять числа y_0, y_1, такие, что

$$[0]^{x,y_0} = [1]^{x,y_1}$$

т.е. $g^{y_0} = xg^{y_1}$, или $g^{y_0 - y_1} = x$, то, следовательно, a умеет за полиномиальное время решать задачу дискретного логарифмирования (вычислять $\log_g x$), что, по предположению, невозможно.

4.6 Совместная генерация строки

Если агенты a и b используют одну и ту же АСШ, и хотят совместно сгенерировать строку $u \in \{0,1\}^n$ (например для использования её в качестве общего ключа ССШ), причем вклад a и b в формирование каждого бита этой строки должен быть одинаковым, то они могут сделать это следующим образом:

- a генерирует u_a, b генерирует u_b, где $u_a, u_b \in \{0,1\}^n$
- $b \to a : r$ (где r – нонс)
- $a \to b : a^+(u_a, r)$
- $b \to a : u_b$
- $a \to b : a^-$
- a и b вычисляют искомую строку u как $u_a \oplus u_b$.

Пересылка от a к b строки u_a в зашифрованном виде необходима потому, что если b будет знать строку u_a до того, как он сгенерирует свою строку u_b, то он может сгенерировать её так, чтобы искомая строка $u = u_a \oplus u_b$ обладала желательными для него свойствами, что нарушает условие равноправия a и b при генерации u.

Если строка u, которую должны сгенерировать a и b, должна состоять из одного бита, то метод генерации может быть иным. Если дополнительно предположить, что используемая агентами a и b АСШ удовлетворяет условию коммутирования операций шифрования и дешифрования (например, этому условию удовлетворяет RSA), то можно сгенерировать u путем выполнения следующих действий:

- $b \to a : \{b^+(r_0), b^+(r_1)\}$, где r_0 и r_1 – нонсы, и последний бит нонса r_i равен i $(i = 0, 1)$
- $a \to b : a^+b^+(r_i)$, где $i \underset{r}{\in} \{0,1\}$
- $b \to a : b^-a^+b^+(r_i) \quad (= a^+(r_i))$
- $a \to b : r_i$,
  искомая строка u состоит из последнего бита нонса r_i
- a и b открывают друг другу закрытые ключи a^-, b^-.

Совместную генерацию однобитовой строки u можно произвести и следующим образом:

- $a \to b : y := h(x)$, где h – ХФ, $x \underset{r}{\in} \{0,1\}^*$
- $b \to a : y \underset{r}{\in} \{0,1\}$
- $a \to b : x$
- искомая строка $u \stackrel{\text{def}}{=}$ (последний бит x) $\oplus\, y$.

Глава 5

Протоколы аутентификации

5.1 Понятие протокола аутентификации

Протоколы аутентификации (ПА) предназначены для решения задачи **аутентификации** (т.е. доказательства подлинности) агентов, ключей, сообщений, времени создания сообщений, сеансов связи, и т.д.

В этой главе мы рассматриваем 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,
- а также может принимать участие доверенный посредник, которого мы будем обозначать символом s, в этом случае мы будем предполагать, что a и b имеют ключи ССШ k_a и k_b для связи с s (ключ k_a известен только a и s, а ключ k_b известен только b и s).

ПА агентов включают в себя следующие два класса:

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

Некоторые ПА агентов предназначены для одновременного решения двух задач:

- аутентификации агентов, и
- выработки ими (или передачи им от s) нового сеансового ключа ССШ, который эти агенты могут использовать для организации сеанса шифрованной связи друг с другом после завершения работы ПА.

5.2 Протоколы односторонней аутентификации

Как правило, аутентификация агента a заключается в доказательстве им другому агенту b того, что a знает некоторое секретное значение a^-. Главная особенность этого доказательства заключается в том, что a должен убедить b в том, что он знает a^-, не раскрывая при этом значение a^-. Если после завершения этого доказательства у b не появляется новой информации о том, в каком диапазоне может содержаться значение a^-, то такое доказательство называется **доказательством с нулевым разглашением (Д0Р)**.

5.2.1 Простейшие протоколы односторонней аутентификации

1. с использованием ССШ и МВ:

   $$a \to b : k(b, t)$$

   в качестве секретного значения a^- здесь выступает ключ k ССШ, который не должен быть известен никому кроме a и b

2. с использованием ЭП и МВ:

   $$a \to b : \langle b, t \rangle_a$$

   в качестве секретного значения a^- здесь выступает закрытый ключ ЭП агента a.

5.2.2 Протоколы аутентификации с использованием паролей

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

Простейший ПА с использованием пароля состоит из пересылки одного сообщения:

$$a \to b : a, \text{пароль}.$$

Для защиты от перехвата паролей можно использовать ПА

$$a \to b : \ a, r, h(\text{пароль}, r) \quad (\text{где } r - \text{нонс}).$$

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

1. 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. a и b используют алгоритм, который по каждому паролю p_i вырабатывает ключ ССШ k_i. i–й сеанс аутентификации a с b имеет вид

   $$a \to b : p_i, k_i(p_{i+1}).$$

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

   a выбирает нонс r, и генерирует последовательность паролей $p_0 := r$, $p_1 := h(p_0)$, $\ldots$, $p_n := h(p_{n-1})$.

   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 $h(p_{n-i}) = p_{n-i+1}$.

5.2.3 Вопросно-ответные протоколы односторонней аутентификации

Работа вопросно-ответного протокола односторонней аутентификации обычно состоит из нескольких раундов, в каждом из которых

- a посылает b доказательство знания секрета a^-, и
- b проверяет это доказательство и либо принимает его, либо не принимает.

a проходит аутентификацию только в том случае, если в каждом из раундов b принимает доказательство, которое ему прислал a. Как правило, в каждом раунде

- b посылает a некоторый вопрос, и
- доказательство знания секрета a^-, которое в этом раунде a посылает b, представляет собой ответ на этот вопрос.

Для защиты от атаки с повторной передачей (**replay**) можно включать в вопросы и ответы нонсы (т.е. случайные строки) или метки времени. В этом случае при проверке ответа производится дополнительная проверка того, что

- нонс, содержащийся в ответе, совпадает с нонсом, содержащемся в вопросе, или
- метка времени в ответе принадлежит заданному промежутку $[t_{min}, t_{max}]$.

Однораундовые протоколы односторонней аутентификации

В этом пункте мы приводим примеры простейших однораундовых протоколов односторонней аутентификации.

1. с использованием ССШ и нонса:
   - $b \to a : r$
   - $a \to b : k(b, r)$
2. с использованием ЭП и нонса:
   - $b \to a : r$
   - $a \to b : \langle b, r \rangle_a$
3. с использованием АСШ и нонса:
   - $b \to a : r$
   - $a \to b : a, a^-(r)$
4. с использованием ХФ и нонсов:
   - $b \to a : r_b$
   - $a \to b : r_a, h(a, r_a, r_b)$
5. с использованием АСШ, ХФ и нонса
   - $b \to a : \ h(r), b, a^+(a, r)$
   - $a \to b : r$

Протокол аутентификации Шнорра

Параметры (открытые):

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

$a^- := s \underset{r}{\in} \mathbf{Z}_q^*$, $a^+ = v := (g^{-s})_p$.

Протокол Шнорра состоит из следующих действий:

- $a \to b : x := (g^z)_p$, где $z \underset{r}{\in} \mathbf{Z}_q^*$
- $b \to a : e \underset{r}{\in} \mathbf{Z}_q^*$, где $|e| < 100$,
- $a \to b : y := (z + se)_q$
- b принимает ответ, если $x \underset{p}{=} g^y v^e$.

Предположим, что a не знает s, но хочет послать в качестве y то значение, которое примет b, тогда

- если a заранее знает значение e, которое ему пришлёт b во втором действии, то в качестве x он может послать значение $g^z v^e$, и в качестве y – значение z,
- если же a не знает e заранее, то для того, чтобы послать то значение y, которое примет b, он должен уметь вычислять для каждого возможного e значение $y = (z + se)_q$, что равносильно знанию $s \underset{q}{=} (y - z)e^{-1}$.

Можно доказать, что данный ПА является Д0Р.

Протокол Шаума

Chaum

n – открытое число.

$a^- = s$, $a^+ = v := (g^s)_n$, где $g \in \mathbf{Z}_n^*$ – порождающий элемент.

Протокол состоит из t раундов, каждый из которых имеет следующий вид:

- $a \to b : x := (g^z)_n$, где $z \underset{r}{\in} \{1, \ldots, \varphi(n)\}$,
- $b \to a : e \underset{r}{\in} \{0, 1\}$,

- $a \to b : y := (z + se)_{\varphi(n)}$
- b принимает y, если $g^y \underset{n}{=} xv^e$.

a проходит аутентификацию, если b принимает ответ a в каждом раунде. Если a не знает s, то вероятность того, что a пройдёт аутентификацию, равна 2^{-t}.

Если a не знает s, но хочет успешно пройти аутентификацию, то, в том случае когда ещё до выполнения какого-либо раунда a заранее знает значение e, которое ему пришлёт b во втором действии, то в этом раунде в качестве x он может послать $g^z v^{-e}$, и в качестве y – значение z.

Модификации протокола Шаума:

1. $a^- = (s_1, \ldots, s_l)$, $a^+ = (v_1, \ldots, v_l)$, где

$$\forall\, i = 1 \ldots, l \quad v_i := (g^{s_i})_n$$

где $g \in \mathbf{Z}_n^*$ – порождающий элемент.

Протокол состоит из одного раунда, который имеет следующий вид:

- $a \to b : x := (g^z)_n$, где $z \underset{r}{\in} \{1, \ldots, \varphi(n)\}$,
- $b \to a : (e_1, \ldots, e_l)$, где $\forall\, i = 1, \ldots, l \quad e_i \underset{r}{\in} \{0, 1\}$,
- $a \to b : y := (z + \sum\limits_{i=1}^{l} s_i e_i)_{\varphi(n)}$
- b принимает y, если $g^y \underset{n}{=} x \prod\limits_{i=1}^{l} v_i^{e_i}$.

2. $a^- = (s_1, \ldots, s_l)$, $a^+ = v := (\prod\limits_{i=1}^{l} g_i^{s_i})_n$, где $g_1, \ldots, g_l$ – элементы $\mathbf{Z}_n^*$ большого порядка.

Протокол состоит из t раундов, каждый из которых имеет следующий вид:

- $a \to b : (x_1, \ldots, x_l)$, где

$$\forall\, i = 1, \ldots, l \quad x_i := (g_i^{z_i})_n, \quad z_i \underset{r}{\in} \{1, \ldots, \varphi(n)\}$$

- $b \to a : e \underset{r}{\in} \{0, 1\}$,
- $a \to b : \{y_i := (z_i + s_i e)_{\varphi(n)} \mid i = 1, \ldots, l\}$
- b принимает y, если $\prod\limits_{i=1}^{l} g_i^{y_i} \underset{n}{=} (\prod\limits_{i=1}^{l} x_i) v^e$.

3. $a^- = s$, $a^+ = (v_1, \ldots, v_l)$, где

$$\forall\, i = 1, \ldots, l \quad v_i := (g_i^s)_n$$

Протокол состоит из t раундов, каждый из которых имеет следующий вид:

- $a \to b : \{x_i := (g_i^z)_n \mid i = 1, \ldots, l\}$, где $z \underset{r}{\in} \{1, \ldots, \varphi(n)\}$,
- $b \to a : e \underset{r}{\in} \{0, 1\}$,
- $a \to b : y := (z + se)_{\varphi(n)}$
- b принимает y, если $\forall\, i = 1, \ldots, l \quad g_i^y \underset{n}{=} x_i v_i^e$.

Протокол Фиата-Шамира

n – открытое число вида pq, где p, q – секретные простые числа, $|p|, |q| \geq 512$.

Все операции выполняются по mod n.

$a^- = s \underset{r}{\in} \mathbf{Z}_n^*$, $a^+ = v := s^2$.

Протокол состоит из t раундов, каждый из которых имеет следующий вид:

- $a \to b : x := z^2$, где $z \underset{r}{\in} \mathbf{Z}_n \setminus \{0\}$,
- $b \to a : e \underset{r}{\in} \{0, 1\}$,
- $a \to b : y := zs^e$
- b принимает y, если $y^2 = xv^e$.

a проходит аутентификацию, если b принимает ответ a в каждом раунде.

Если a не знает a^-, но хочет пройти аутентификацию, то для каждого значения $e \in \{0, 1\}$, которое b пришлёт a, агент a должен уметь вычислить значение y_e, которое он пошлет b в качестве y. Из условия принятия y следует, что $y_0^2 = x$ и $y_1^2 = xv$, поэтому $y_1^2 = y_0^2 v$. Можно доказать, что знание пары ненулевых чисел y_0, y_1, удовлетворяющих равенству $y_1^2 = y_0^2 v$, равносильно возможности за полиномиальное время вычислить $\sqrt{v}$ в $\mathbf{Z}_n$ для $n = pq$, где p и q – простые числа, что является вычислительно сложной задачей в том случае, когда числа p и q неизвестны.

Если же ещё до выполнения раунда a заранее знает значение e, которое ему пришлёт b во втором действии, то для того, чтобы пройти аутентификацию, в качестве x он может послать значение z^2v^{-e}, и в качестве y – значение z.

Таким образом, если a не знает s, то вероятность успешного ответа a в одном раунде равна $1/2$, и, поскольку все раунды независимы, то вероятность того, что a не ошибётся во всех раундах, равна произведению вероятностей успешного ответа a в одном раунде, т.е. 2^{-t}.

Можно доказать, что данный протокол является Д0Р.

ПА Фиата-Шамира можно модифицировать, заменив в нём возведение в квадрат на возведение в степень l, где $l \geq 2$ – открытое число. Соответствующий ПА называется **ПА Гиллу-Кискате**. В нём $a^+ = v := (s^l)^{-1}$, и каждый раунд имеет следующий вид:

- $a \to b : x := z^l$, где $z \underset{r}{\in} \mathbf{Z}_n^*$,
- $b \to a : e \underset{r}{\in} \{0, \ldots, l-1\}$,
- $a \to b : y := zs^e$,
- b принимает y, если $x = y^l v^e$.

Данный ПА тоже является Д0Р.

Протокол Фейге-Фиата-Шамира

n – открытое число вида pq, где p, q – секретные простые числа, $p \underset{4}{=} q \underset{4}{=} 3$ (т.е. n – число Блюма), $|p|, |q| \geq 512$.

Все операции выполняются по $\bmod n$.

- $a^- = (s_1, \ldots, s_l)$, где $\forall\, i = 1, \ldots, l \;\; s_i \underset{r}{\in} \mathbf{Z}_n^*$,
- $a^+ = (v_1, \ldots, v_l)$, где $\forall\, i = 1, \ldots, l$

$$v_i := \pm(s_i^2)^{-1}, \quad \pm \underset{r}{\in} \{+, -\},$$

  причем числа $v_1, \ldots, v_l$ различны.

Протокол состоит из t раундов, каждый из которых имеет следующий вид:

- $a \to b : x := \pm z^2$, где $z \underset{r}{\in} \mathbf{Z}_n \setminus \{0\}$, $\pm \underset{r}{\in} \{+, -\}$
- $b \to a : (e_1, \ldots, e_l) \underset{r}{\in} \{0, 1\}^l$

- $a \to b : y := z s_1^{e_1} \dots s_l^{e_l}$
- b принимает y, если $x = \pm y^2 v_1^{e_1} \dots v_l^{e_l}$.

a проходит аутентификацию, если b принимает ответ a в каждом раунде.

Если a не знает a^-, но хочет пройти аутентификацию, то для того, чтобы послать в качестве y значение, которое примет b,

- если a заранее знает кортеж $(e_1, \dots, e_l)$, который ему пришлёт b во втором действии, то в качестве x он может послать значение $\pm z^2 v_1^{e_1} \dots v_l^{e_l}$, и в качестве y – значение z,
- если же a не знает $(e_1, \dots, e_l)$ заранее, то для того, чтобы послать то значение y, которое примет b, он должен уметь вычислять значения функции $\sqrt{u}$ в $\mathbf{Z}_n$ для $n = pq$, где p и q – простые числа, что является вычислительно сложной задачей в том случае, когда числа p и q ему неизвестны.

Вероятность того, что a, не зная a^-, успешно пройдёт аутентификацию, не превосходит 2^{-lt}.

Можно доказать, что данный протокол является Д0Р.

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

5.2.4 Односторонняя аутентификация с передачей сеансового ключа

Простейшие протоколы

В этом пункте мы представляем три протокола односторонней аутентификации и передачи сеансового ключа с использованием АСШ, ЭП и МВ. Все эти протоколы состоят из одной пересылки сообщения.

1. $a \to b : b^+\langle b, k, t\rangle_a$
2. $a \to b : \langle b, b^+(a, k), t\rangle_a$
3. $a \to b : b^+(k, t), \langle b, k, t\rangle_a$

Протокол Wide Mouth Frog

- $a \to s : a, k_a(b, k, t_a)$
- $s \to b : k_b(a, k, t_b)$

Вопросно-ответный протокол

- $b \to a : r$
- $a \to b : k(b, k', r)$ (или $k' \oplus h(b, k, r)$)

где k и k' – старый и новый сеансовые ключи.

Протокол Отвея-Риса

- $a \to b : a, b, k_a(a, b, r, r_a), r$
- $b \to s : a, b, k_a(a, b, r, r_a), k_b(a, b, r, r_b), r$
- $s \to b : k_a(k, r_a), k_b(k, r_b), r$
- $b \to a : k_a(k, r_a), r$

5.3 Протоколы двусторонней аутентификации

5.3.1 Простейшие протоколы двусторонней аутентификации

1. с использованием ССШ и нонсов
   - $b \to a : r_b$
   - $a \to b : k(b, r_a, r_b)$
   - $b \to a : k(r_a, r_b)$
2. с использованием АСШ и нонсов (протокол Нидхэма-Шрёдера)
   - $a \to b : b^+(a, r_a)$
   - $b \to a : a^+(r_a, r_b)$
   - $a \to b : b^+(r_b)$
3. с использованием ЭП и нонсов
   - $b \to a : r_b$
   - $a \to b : \langle b, r_a, r_b \rangle_a$
   - $b \to a : \langle a, r_a, r_b \rangle_b$
4. с использованием ХФ и нонсов

- $b \to a : r_b$
- $a \to b : r_a, h(a, r_a, r_b)$
- $b \to a : h(b, r_a)$

5.3.2 Протоколы двусторонней аутентификации с передачей сеансового ключа

Простейшие протоколы

1. с использованием ССШ и нонсов

   - $a \to b : r_a$
   - $b \to a : k(r_a, r_b)$
   - $a \to b : k(b, k', r_b)$ (или $k' \oplus h(b, k, r_b)$)

   где k и k' – старый и новый сеансовые ключи,

2. с использованием МВ, ЭП, АСШ и ССШ

   - $a \to b : \langle b^+(k) \rangle_a, k(t_a)$
   - $b \to a : k(t_b)$

Протокол Отвея-Риса

- $a \to b : a, b, k_a(a, b, r, r_a), r$
- $b \to s : a, b, k_a(a, b, r, r_a), k_b(a, b, r, r_b), r$
- $s \to b : k_a(k, r_a), k_b(k, r_b), r$
- $a \to b : k_a(k, r_a), k(r_a, r_b)$
- $b \to a : k(r_a)$

Протокол Yahalom

- $a \to b : a, r_a$
- $b \to s : b, k_b(a, r_a, r_b)$
- $s \to a : k_a(b, k, r_a, r_b), k_b(a, k)$
- $a \to b : k_b(a, k), k(r_b)$

Протокол Ву-Лама

- $a \to b : b^+(a, r_a)$
- $b \to s : a, b, s^+(r_a)$
- $s \to b : b^+\langle a, b, k, r_a\rangle_s$
- $b \to a : a^+(\langle a, b, k, r_a\rangle_s, r_b)$
- $a \to b : k(r_b)$

Протокол Нидхэма-Шрёдера

- $a \to s : a, b, r_a$
- $s \to a : k_a(b, k, k_b(k, a, t), r_a)$
- $a \to b : k_b(a, k)$
- $b \to a : k(r_b)$
- $a \to b : k(r_b - 1)$

Протокол Ньюмана-Стабблбайна

- $a \to b : \ a, \ r_a$
- $b \to s : b, r_b, k_b(a, r_a, t)$
- $s \to a : k_a(b, k, r_a, t), k_b(a, k, t), r_b$
- $a \to b : k_b(a, k, t), k(r_b)$

Данный протокол устойчив по отношению к атакам, основанным на десинхронизации часов (которая может произойти из-за сбоя системы или саботажа).

После того, как сеанс связи завершён, a и b могут ещё раз сделать взаимную аутентификацию (для предотвращения атаки с повторной передачей):

- $a \to b : k_b(a, k, t), r'_a$
- $b \to a : k(r'_a), r'_b$
- $a \to b : k(r'_b)$

Протокол Kerberos

Прототип:

- $a \to s:\ a,\ b,\ r$
- $s \to a : k_a(b, k, r, l), k_b(a, k, l)$
- $a \to b : k_b(a, k, l), k(a, r', t)$
- $b \to a : k(r', t)$

где k_a, k_b – ключи ССШ-связи s с a и b, $l =$ время действия ключа k.

Kerberos предполагает работу с несколькими доверенными посредниками: s (authentication server) и s_1, ..., s_n (tickets grant servers). $\forall\, i = 1, \ldots, n$ k_{a_i}, k_{b_i} и k_{s_i} – ключи ССШ-связи s_i с a, b и s соответственно, k_a, k_b – ключи ССШ-связи s с a и b.

- $a \to s : a, s_i, r$
- $s \to a : k_a(s_i, k_{a_i}, r, l_1), k_{s_i}(a, k_{a_i}, l_1)$
- $a \to s_i : k_{s_i}(a, k_{a_i}, l_1), k_{a_i}(a, t_1), b, r'$
- $s_i \to a : k_{a_i}(b, r', k, l_2), k_{b_i}(a, k, l_2)$
- $a \to b : k_{b_i}(a, k, l_2), k(a, r'', t_2)$
- $b \to a : k(r'', t_2)$

Глава 6

Электронная подпись

6.1 Протоколы ЭП, получаемые из протоколов аутентификации

6.1.1 Протокол ЭП Шнорра

Параметры (открытые):

- простые числа p и q, где $|p| \geq 512$, $|q| \geq 140$, $q \mid p-1$,
- элемент $g \in \mathbf{Z}_p^*$, такой, что $\mathrm{ord}(g) = q$.
- h – ХФ вида $h : \{0,1\}^* \to \{0,1\}^t$.

$a^- := s \underset{r}{\in} \mathbf{Z}_q^*$, $a^+ = v := \underset{p}{} g^{-s}$.
$\langle m \rangle_a^s := (e, y)$, где

$$e := h(m, (g^z)_p), \quad y := (z + se)_q, \quad z \underset{r}{\in} \mathbf{Z}_q^*.$$

Проверка подлинности: $h(m, (g^y v^e)_p) = e$.

6.1.2 Протокол ЭП Фиата-Шамира

Этот ПЭП получается из ПА Фиата-Шамира заменой последовательности случайных битов $e_1, \ldots, e_t$, которые b посылает a в каждом из t раундов после получения x_i от a $(i = 1, \ldots, t)$, на префикс длины t строки $h(m, x_1, \ldots, x_t)$, где h – ХФ.

Параметры:

- n – открытое число вида pq, где p, q – секретные простые числа, $|p|, |q| \geq 512$,

- h – ХФ.

Все операции выполняются по mod n.

$a^- = s \underset{r}{\in} \mathbf{Z}_n^*$, $a^+ = v := s^2$.

Для вычисления $\langle m \rangle_a^s$ агент a генерирует

$$z_1, \ldots, z_t, \quad \text{где } \forall\, i = 1, \ldots, t \quad z_i \underset{r}{\in} \mathbf{Z}_n \setminus \{0\}.$$

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

- $e = (e_1, \ldots, e_t)$ – первые t битов числа

$$h(m, x_1, \ldots, x_t), \quad \text{где } \forall\, i = 1, \ldots, t \quad x_i := z_i^2,$$

- $y = (y_1, \ldots, y_t)$, где $\forall\, i = 1, \ldots, t \quad y_i := z_i s^{e_i}$.

Проверка подлинности:

$$e = \text{ первые } t \text{ бит } h(m, u_1, \ldots, u_t),$$

где $\forall\, i = 1, \ldots, t \quad u_i := y_i^2 v^{-e_i}$.

6.1.3 Протокол ЭП Фейге-Фиата-Шамира

Этот ПЭП получается из ПА Фейге-Фиата-Шамира заменой последовательности случайных битовых кортежей

$$(e_{11}, \ldots, e_{1l}), \ldots, (e_{t1}, \ldots, e_{tl})$$

которые b посылает a в каждом из t раундов после получения x_i от a $(i = 1, \ldots, t)$, на префикс длины tl строки $h(m, x_1, \ldots, x_t)$, где h – ХФ.

Параметры:

- n – открытое число вида pq, где p, q – секретные простые числа Блюма, $|p|, |q| \geq 512$,
- h – ХФ.

Все операции выполняются по mod n.

- $a^- = (s_1, \ldots, s_l)$, где $\forall\, i = 1, \ldots, l \;\; s_i \underset{r}{\in} \mathbf{Z}_n^*$,
- $a^+ = (v_1, \ldots, v_l)$, где $\forall\, i = 1, \ldots, l \quad v_i := (s_i^2)^{-1}$, причем числа $v_1, \ldots, v_l$ различны.

Для вычисления $\langle m\rangle_a^s$ агент a генерирует

$$z_1, \ldots, z_t, \quad \text{где } \forall\, i = 1, \ldots, t \quad z_i \underset{r}{\in} \mathbf{Z}_n.$$

$\langle m\rangle_a^s := (e, y)$, где

- $e = (e_{11}, \ldots, e_{1l}, \ldots, e_{t1}, \ldots, e_{tl})$ – первые tl битов числа

$$h(m, x_1, \ldots, x_t), \quad \text{где } \forall\, i = 1, \ldots, t \quad x_i := z_i^2,$$

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

Проверка подлинности:

$$e = \text{ первые } tl \text{ бит } h(m, u_1, \ldots, u_t),$$

где $\forall\, i = 1, \ldots, t \quad u_i := y_i^2 v_1^{e_{i1}} \ldots v_l^{e_{il}}$.

Вероятность обмана не превосходит 2^{-tl}.

Рекомендуется брать $l = 9, t = 8$.

На этот ПЭП м.б. атаки, основанные на парадоксе дней рождения.

6.1.4 Протокол ЭП Гиллу-Кискате

Этот ПЭП получается из ПА Гиллу-Кискате заменой случайного значения $e \underset{r}{\in} \{0, \ldots, l-1\}$, которое генерирует b после получения $x := z^l$ от a, на $h(m, x)$, где h – ХФ с множеством значений $\mathbf{Z}_l$.

Параметры:

- $n := pq$ – открытое число, где p, q – секретные простые числа, $|p|, |q| \geq 512$,
- $l \geq 2$ – открытое число.

Все вычисления и сравнения - по mod n.

$a^- = s \underset{r}{\in} \mathbf{Z}_n^*, \;\; a^+ = v := (s^l)^{-1}$.

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

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

6.2 Другие протоколы ЭП

6.2.1 Протокол ЭП DSA

DSA = Digital Signature Algorithm.

Параметры:

- p, q – простые числа, где $q \mid p-1$, $|p| \geq 512$ и делится на 64, $|q|$ – примерно 160,
- $g \in \mathbf{Z}_p^*$ – элемент порядка q,
- h – ХФ со значениями в $\mathbf{Z}_q^*$.

$a^- = x \underset{r}{\in} \mathbf{Z}_q$, $a^+ = y :\underset{p}{=} g^x$.

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

Проверка подлинности: $s_1 \underset{q}{=} (g^{u_1} y^{u_2})_p$, где

$$u_1 :\underset{q}{=} h(m)u, \quad u_2 :\underset{q}{=} s_1 u, \quad u :\underset{q}{=} s_2^{-1}.$$

6.2.2 Протокол ЭП ГОСТ

Параметры, a^-, a^+ – те же, что и у DSA, $|q| = 256$.

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

Проверка подлинности: $s_1 \underset{q}{=} (g^{u_1} y^{u_2})_p$, где

$$u_1 :\underset{q}{=} s_2 u, \quad u_2 :\underset{q}{=} -s_1 u, \quad u :\underset{q}{=} h(m)^{-1}.$$

6.2.3 Протокол ЭП Эль-Гамаля

Параметры:

- p – открытое простое число,
- $g \in \mathbf{Z}_p^*$ – примитивный элемент,
- h – ХФ со значениями в $\mathbf{Z}_p$.

$a^- = x \underset{r}{\in} \mathbf{Z}_{p-1}$, $a^+ = y := g^x$.

$\langle m \rangle_a^s := (s_1, s_2)$, где $\begin{cases} s_1 := g^z, \quad z \underset{r}{\in} \mathbf{Z}_{p-1}^* \\ s_2 := (h(m) - xs_1)z^{-1}. \end{cases}$

Проверка подлинности: $y^{s_1} s_1^{s_2} = g^{h(m)}$.

6.2.4 Обобщённый протокол ЭП

Излагаемый в настоящем пункте протокол ЭП является обобщением протоколов Шнорра, DSA и Эль-Гамаля.

Параметры:

- p, q – большие простые числа, причем $q \mid p - 1$,
- $g \in \mathbf{Z}_p^*$ – элемент порядка q,
- h – ХФ со значениями в $\mathbf{Z}_q^*$.

$a^- = x \underset{r}{\in} \mathbf{Z}_q$, $a^+ = y := (g^x)_q$.
$\langle m \rangle_a^s := (r, s_1, s_2, s_3)$, где

- $r := (g^z)_p$ $(z \underset{r}{\in} \mathbf{Z}_q^*)$, и
- $\{s_1, s_2, s_3\}$ совпадает с одним из перечисляемых ниже множеств:

$$\{\pm r', \pm s, h(m)\}, \{\pm r' h(m), \pm s, 1\}, \{\pm r' h(m), \pm s h(m), 1\},$$
$$\{\pm r' h(m), \pm r' s, 1\}, \{\pm s h(m), \pm r' s, 1\},$$

  где допускается любой выбор знаков, $r' := (r)_q$, и s ищется из уравнения $s_1 z \underset{q}{=} s_2 + s_3 x$.

Проверка подлинности: $r^{s_1} \underset{p}{=} g^{s_2} \cdot y^{s_3}$.

6.3 Стираемая электронная подпись

Иногда агент a хочет, чтобы

- подлинность его ЭП $\langle m \rangle_a^s$ могла быть доказана (или опровергнута) только с его участием, и

- если a доказал (или опроверг) подлинность $\langle m \rangle_a^s$ совместно с каким-либо агентом b, то b не мог бы использовать запись этого рассуждения для того, чтобы доказать (или опровергнуть) подлинность $\langle m \rangle_a^s$ другим агентам.

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

Доказательство или опровержение подлинности определяемых в этом параграфе стираемых ЭП производится при помощи интерактивных протоколов, где под **интерактивным протоколом (ИП)** распознавания 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.

6.3.1 Протокол Шаума–Антверпена стираемой ЭП

Параметры:

- p, q – простые числа, где $q \mid p-1$,
- $g \in \mathbf{Z}_p^*$ – элемент порядка q.

Ниже все вычисления и сравнения – по mod p.

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

$a^- = x \underset{r}{\in} \mathbf{Z}_q^*$, $a^+ = g^x$, $\langle m \rangle_a^s := m^x$.

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

- $b \to a : y := z^u (a^+)^v$, где $u, v \underset{r}{\in} \mathbf{Z}_q$,
- $a \to b : h := y^{(x^{-1})}$,
- $b : [\![h = m^u g^v]\!]$ утверждение верно.

Отметим, что данный протокол не обеспечивает нулевого разглашения a^-.

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

Оценим вероятность того, что в результате работы данного протокола для некоторой неподлинной ЭП $z \neq m^x$ сообщения m в результате работы протокола агент b примет решение, что ЭП z является подлинной:

- элемент $m_1 \stackrel{\text{def}}{=} z^{(x^{-1})}$ удовлетворяет равенству $z = m_1^x$, следовательно, значение y, которое a получит от b, равно $m_1^{xu} g^{xv}$,
- поскольку a может вычислить x^{-1}, то, следовательно, a может вычислить $(m_1^{xu} g^{xv})^{x^{-1}} = m_1^u g^v$,

поэтому, если a сможет убедить b в подлинности неверной подписи z сообщения m (т.е. пошлет ему $h = m^u g^v$), то значит a также сможет вычислить

$$\frac{m_1^u g^v}{m^u g^v} = \frac{m_1^u}{m^u} = \left(\frac{m_1}{m}\right)^u.$$

a не знает значение u (которое для него закрыто случайным неизвестным множителем $(a^+)^v$), поэтому вероятность того, что a может послать правильное h, равна вероятности того, что он правильно угадает u, эта вероятность равна $1/d$, где d – порядок элемента m_1/m в группе $\mathbf{Z}_p^*$.

Таким образом, искомая вероятность $\leq 1/d$. ■

6.3.2 Протокол Шаума стираемой ЭП

Параметры:

- p – большое простое число,
- $g \in \mathbf{Z}_p^*$ – примитивный элемент.

Ниже все вычисления и сравнения – по mod p.

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

$a^- = x \underset{r}{\in} \mathbf{Z}_p^*$, $a^+ = g^x$, $\langle m \rangle_a^s := m^x$.

1. Протокол подтверждения подлинности $\langle m \rangle_a^s$ (т.е. доказательство утверждения $z = m^x$, где $z \in \mathbf{Z}_p^*$):
   - $b \to a : y := m^u g^v$, где $u, v \underset{r}{\in} \mathbf{Z}_{p-1}$,

- $a \to b : (h_1, h_2)$, где $h_1 := yg^w$, $h_2 := h_1^x$, $w \underset{r}{\in} \mathbf{Z}_{p-1}$,
- $b \to a : (u, v)$,
- $a \to b : [\![y = m^u g^v]\!]\ w$,
- $b : \left[\!\!\left[\begin{array}{l} h_1 = yg^w \\ h_2 = z^u (a^+)^{v+w} \end{array} \right]\!\!\right]$ утверждение верно.

Отметим, что данный протокол не обеспечивает нулевого разглашения a^-.

Стираемость этой ЭП обосновывается тем, что агент b может самостоятельно

- выбрать произвольное $w \underset{r}{\in} \mathbf{Z}_{p-1}$,
- вычислить значения yg^w и $z^u (a^+)^{v+w}$, и
- предъявить их в качестве значений h_1 и h_2, которые ему якобы переслал агент a.

Оценим вероятность того, что в результате работы данного протокола для некоторой неподлинной ЭП z сообщения m в результате работы протокола агент b примет решение, что ЭП z является подлинной:

- элемент $m_1 \overset{\text{def}}{=} z^{(x^{-1})}$ удовлетворяет равенству $z = m_1^x$, следовательно, если a сможет убедить b в подлинности неверной подписи z сообщения m, то значение h_2 удовлетворяет условию
$$h_2 = z^u (a^+)^{v+w} = (m_1^x)^u (g^x)^{v+w} = (m_1^u g^{v+w})^x$$
- поскольку a может вычислить x^{-1}, то, следовательно, a может вычислить $(h_2)^{x^{-1}} = m_1^u g^{v+w}$, откуда следует, что a может вычислить
$$\frac{(h_2)^{x^{-1}}}{h_1} = \frac{m_1^u g^{v+w}}{yg^w} = \frac{m_1^u g^{v+w}}{m^u g^v g^w} = \frac{m_1^u}{m^u} = \left(\frac{m_1}{m}\right)^u.$$

До того, как a посылает b значение h_2, a не знает значение u (которое для него закрыто случайным неизвестным множителем g^v), поэтому a может послать правильное h_2 только если он правильно угадает u, это может произойти с вероятностью $1/d$, где d – порядок элемента m_1/m в группе $\mathbf{Z}_p^*$.

Таким образом, искомая вероятность $\leq 1/d$. ■

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

   - $b \to a : \left(\begin{array}{l} y_1 := m^u g^v \\ y_2 := z^u (a^+)^v \end{array} \right)$, где $u \underset{r}{\in} \mathbf{Z}_k, v \underset{r}{\in} \mathbf{Z}_{p-1}$,
   - $a \to b : [w]^r$, где r – случайная маска, и $w \in \mathbf{Z}_k$ – решение уравнения
     $$\frac{y_1^x}{y_2} = \left(\frac{m^x}{z} \right)^w, \tag{6.1}$$
     которое ищется перебором,
     (u – одно из решений этого уравнения)
   - $b \to a : v$
   - $a \to b : \left[\!\!\left[\begin{array}{l} y_1 = m^w g^v \\ y_2 = z^w (a^+)^v \end{array} \right]\!\!\right] r$
   - $b : [\![w = u]\!]$ утверждение верно.

   Обоснуем корректность данного протокола:

   - если утверждение $z \neq m^x$ истинно, то $w \underset{d}{=} u$,
   - если утверждение $z \neq m^x$ ложно (т.е. в действительности $z = m^x$), то любой элемент $\mathbf{Z}_k$ является решением уравнения (6.1), вероятность того что $w = u$, равна $1/d$.

6.3.3 Протокол ГОСТ стираемой ЭП

Параметры:

- p, q – большие простые числа, $q \mid p - 1$,
- $g \in \mathbf{Z}_p^*$ – элемент порядка q,

Ниже все вычисления и сравнения (кроме специально оговоренных) – по $\bmod p$.

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

$$a^- = (x_1, x_2), \quad \text{где } x_1, x_2 \underset{r}{\in} \mathbf{Z}_q, \ \ x_1 \neq x_2$$
$$a^+ = (g_1, g_2), \quad \text{где } g_1 := g^{x_1}, \ g_2 := g^{x_2}.$$

$$\langle m\rangle_a^s := (s_1, s_2), \text{ где } \begin{cases} s_1 := g^u, \ u \underset{r}{\in} \mathbf{Z}_q, \\ s_2 := \underset{q}{} x_1 s_1 + m x_2 u. \end{cases}$$

1. Протокол подтверждения подлинности $\langle m\rangle_a^s$ (т.е. доказательство утверждения
$$(z_1, z_2) = \langle m\rangle_a^s, \quad \text{где } z_1, z_2 \in \mathbf{Z}_p^* \tag{6.2}$$
   - $b \to a : y := z_1^u g^v$, где $u, v \underset{r}{\in} \mathbf{Z}_q$
   - $a \to b : (h_1, h_2), \quad$ где $h_1 := y g^w, \ h_2 := h_1^{x_2}, \ w \underset{r}{\in} \mathbf{Z}_q$
   - $b \to a : (u, v)$
   - $a \to b : [\![y = z_1^u g^v]\!] \ w$
   - $b : \left[\!\!\left[\begin{array}{l} h_1 = z_1^u g^{v+w} \\ h_2 = \gamma^u g_2^{v+w} \end{array} \right]\!\!\right]$ утверждение верно, где
$$\gamma := z_1^{x_2} = g^{z_2 m^{-1}} g_1^{-z_1 m^{-1}}$$

2. Протокол опровержения подлинности $\langle m\rangle_a^s$, т.е. доказательство утверждения
$$(z_1, z_2) \neq \langle m\rangle_a^s, \quad \text{где } z_1, z_2 \in \mathbf{Z}_p^* \tag{6.3}$$
   - $b \to a : (y_1, y_2, w)$, где $i \underset{r}{\in} \{0, 1\}$, и
     - $(y_1, y_2) = \{i = 0\}\ (g^v, g_2^v) : (z_1^v, \gamma^v), \ v \underset{r}{\in} \mathbf{Z}_q$
     - $w \underset{r}{\in} \mathbf{Z}_q \setminus \{1\}$,
   - $a \to b : [w^j]^r$, где
     - $j := (y_1^{x_2} = y_2)\,?\,0 : 1$, и
     - r – случайная маска,
   - $b \to a : v$
   - $a \to b : [\![(y_1, y_2) = (g^v, g_2^v) \text{ или } (z_1^v, \gamma^v)]\!] \ r$
   - $b : [\![i = j]\!]$ утверждение верно.

   Данный протокол повторяется k раз. Агент b считает утверждение (6.3) верным, если если в каждом из k сеансов этого протокола b принимает решение, что (6.3) верно. Вероятность того, что утверждение (6.3) верно, а b принял решение, что оно неверно, равна 2^{-k}.

6.4 ЭП, подтверждаемая уполномоченными агентами

Ещё один вид ограничений на возможность проверки ЭП заключается в том, чтобы подлинность ЭП какого-либо агента a могли доказывать только агенты из заданного множества M_a. Агенты, входящие в M_a, называются **уполномоченными** агентами (**designated confimers**). В этом пункте мы рассмотрим случай, когда M_a имеет вид $\{a, c\}$.

Параметры:

- p, p_1, p_2 – большие простые числа,
- $g \in \mathbf{Z}_p^*$ – элемент порядка $p-1$,
- h – ХФ.

Ниже все вычисления и сравнения (кроме специально оговоренных) – по mod p.

$$c^- = x \underset{r}{\in} \mathbf{Z}_p^*,$$
$$a^- = (p_1, p_2), \quad a^+ = (p, g, \eta, n), \qquad \text{где } \eta = g^x, n = p_1 p_2.$$

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

$$s_1 = g^y, \quad s_2 = \eta^y, \quad s_3 = (h(m) \oplus h(s_1, s_2))_n^{1/3}, \quad y \underset{r}{\in} \mathbf{Z}_p^*$$

(поскольку агент a знает разложение n на простые множители, то он может быстро вычислять функцию $(x)_n^{1/3}$).

1. Доказательство подлинности ЭП $\langle m \rangle_a^s$ агентом a:
   - $b \to a : z := g^u \eta^v$, где $u, v \underset{r}{\in} \mathbf{Z}_p^*$
   - $a \to b : (d, e)$, где $d := g^w$, $e := (zd)^y$, $w \underset{r}{\in} \mathbf{Z}_p^*$
   - $b \to a : (u, v)$
   - $a \to b : [\![g^u \eta^v = z]\!] \, w$
   - $b : \left[\!\!\left[\begin{array}{l} g^w = d \\ e/(s_1^w) = {s_1}^u {s_2}^v \\ h(m) \oplus h(s_1, s_2) \underset{n}{=} {s_3}^3 \end{array} \right]\!\!\right]$ принимает ЭП.

   Если b попытается убедить другого участника в подлиннности ЭП $\langle m \rangle_a^s$ путём показа записи исполнения этого протокола, то это ему не удастся: он мог её и подделать.

2. Доказательство подлинности ЭП $\langle m \rangle_a^s$ уполномоченным агентом c:

   - $b \to c : z := g^u s_1^v$, где $u, v \underset{r}{\in} \mathbf{Z}_p$
   - $c \to b : (d, e)$, где $d := g^w$, $e := (zd)^x$, $w \underset{r}{\in} \mathbf{Z}_p^*$
   - $b \to c : (u, v)$
   - $c \to b : [\![g^u s_1^v = z]\!]\ w$
   - $b : \left[\!\!\left[\begin{array}{l} g^w = d \\ e/\eta^w = \eta^u s_2^v \\ h(m) \oplus h(s_1, s_2) \underset{n}{=} {s_3}^3 \end{array} \right]\!\!\right]$ принимает ЭП.

На базе этого ПЭП и схемы разделения секрета можно построить такой ПЭП, в котором доказывать подлинность ЭП a могут любые m агентов из заданной совокупности $\{c_1, \ldots, c_n\}$.

6.5 Слепая ЭП

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

Один из протоколов слепой ЭП имеет следующий вид.

Параметры:

- p, q – секретные большие простые числа,
- число $n \stackrel{\text{def}}{=} pq$ открыто,
- h – ХФ со значениями в $\mathbf{Z}_n$.

Ниже все вычисления и сравнения – по mod n.

$$a^+ = e \underset{r}{\in} \mathbf{Z}_{\varphi(n)}^*, \quad a^- = d \in \mathbf{Z}_{\varphi(n)}^*, \quad \text{где } ed \underset{\varphi(n)}{=} 1.$$

Обмен сообщениями между a и b имеет следующий вид:

- $b \to a : u := h(m) x^e$, где $x \underset{r}{\in} \mathbf{Z}_n^*$
- $a \to b : v := u^d$
- b вычисляет искомое значение $\langle m \rangle_a^s := v x^{-1}$.

Нетрудно видеть, что

$$\langle m\rangle_a^s = vx^{-1} = (u^d)x^{-1} = ((h(m)x^e)^d)x^{-1} =$$
$$= h(m)^d x^{ed}x^{-1} = h(m)^d xx^{-1} = h(m)^d.$$

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

6.6 Протоколы совместной ЭП

6.6.1 Понятие протокола совместной ЭП

Иногда требуется, чтобы сообщение m было одновременно подписано несколькими агентами $a_1, \ldots, a_k$. Соответствующая ЭП называется **совместной ЭП**, обозначается записью $\langle m\rangle_{a_1\ldots a_k}^s$, и может иметь, например, следующий вид:

$$\langle m\rangle_{a_1\ldots a_k}^s := (\langle m\rangle_{a_1}^s, \ldots, \langle m\rangle_{a_k}^s) \tag{6.4}$$

(каждый из агентов вычисляет свою компоненту совместной ЭП независимо от других), или (с использованием RSA)

$$\langle m\rangle_{a_1\ldots a_k}^s := a_k^- \ldots a_1^-(m) \tag{6.5}$$

т.е. агенты вносят свой вклад в создание совместной ЭП по очереди: каждый из агентов (a_i) подписывает результат работы предыдущих агентов $(a_{i-1}, \ldots, a_1)$.

В некоторых случаях совместная ЭП должна удовлетворять дополнительным условиям: например,

1. должна быть обеспечена возможность раздельной проверки корректности вклада в совместную ЭП каждого из агентов $a_1, \ldots, a_k$ (нетрудно видеть, что (6.4) удовлетворяет этому условию, а (6.5) – нет), или

2. если совместная ЭП не может быть создана по причине того, что некоторые агенты отказались вносить свой вклад в её создание, в то время как другие агенты $(a_{i_1}, \ldots, a_{i_l})$ свой вклад уже внесли, то по результату работы агентов $a_{i_1}, \ldots, a_{i_l}$ должно быть невозможно идентифицировать этих агентов.

6.6.2 Примеры протоколов совместной ЭП

Приведём ещё два протокола совместной ЭП.

1. Первый протокол имеет следующие параметры:

   - n, v – открытые натуральные числа,
   - h – ХФ со значениями в $\mathbf{Z}_v$.

   Ниже все вычисления и сравнения – в $\mathbf{Z}_n$.
   $\forall\, i = 1, \ldots, k \quad a_i^- = x_i \in \mathbf{Z}_n^*, \quad a_i^+ = y_i := (x_i^v)^{-1}$.
   $\langle m \rangle_{a_1 \ldots a_k}^s := (s_1, s_2)$, где

   $$s_1 := h(m, r_1^v \ldots r_k^v), \quad s_2 := r_1 x_1^{s_1} \ldots r_k x_k^{s_1},$$
   $$\forall\, i = 1, \ldots, k \quad r_i \underset{r}{\in} \mathbf{Z}_n \setminus \{0\}.$$

   Проверка подлинности: $s_1 = h(m, s_2^v y_1^{s_1} \ldots y_k^{s_1})$.

2. Второй протокол называется протоколом совместной ЭП Брикелла-Ли-Якоби. Он имеет следующие параметры:

   - p – простое число,
   - $g \in \mathbf{Z}_p^*$ – элемент порядка $p - 1$,
   - l – большое натуральное число,
   - h – ХФ.

   Ниже все вычисления и сравнения – в $\mathbf{Z}_p$.
   $\forall\, i = 1, \ldots, k \quad a_i^- = \{x_{i1}, \ldots, x_{il}\}, \quad a_i^+ = \{y_{i1}, \ldots, y_{il}\}$, где $y_{ij} = g^{-x_{ij}} \ (j = 1, \ldots, l)$.

   В вычислении ЭП $\langle m \rangle_{a_1 \ldots a_k}^s$ принимает участие доверенный посредник s, с которым агенты $a_1, \ldots, a_k$ могут обмениваться сообщениями с использованием АСШ. Протокол вычисления $\langle m \rangle_{a_1 \ldots a_k}^s$ имеет следующий вид:

   - $a_i \to s : s^+(v_i)$, где $v_i := g^{u_i}, u_i \underset{r}{\in} \mathbf{Z}_{p-1}$
   - $s \to \{a_1, \ldots, a_k\} : v := v_1 \ldots v_k$
   - $a_i \to s : w_i$, где
     $w_i := (u_i + \sum\limits_{b_j = 1} x_{ij})_{p-1}, \quad b_j =$ j-й бит $g^{h(m, v, id_1, \ldots, id_k)}$,
     где $id_1, \ldots, id_k$ – идентификаторы агентов $a_1, \ldots, a_k$

- s вычисляет $\langle m \rangle^s_{a_1 \ldots a_k} = w := (\sum\limits_{i=1}^{k} w_i)_{p-1}$.

Проверка подлинности: $g^w \prod\limits_{i=1}^{k} \prod\limits_{b_j=1} y_{ij} = v$.

Можно доказать, что данный протокол обеспечивает нулевое разглашение, и выполнено второе условие из пункта 6.6.1.

Глава 7

Генерация и передача ключей

Симметричные и асимметричные СШ имеют существенно разные сложностные и криптографические характеристики:

- скорость шифрования в ССШ существенно более высокая, чем в АСШ (примерно в 1000 раз), но
- стойкость ключей, используемых в ССШ, существенно ниже.

Оптимальный режим использования этих СШ заключается в их комбинированном функционировании: для шифрования основной переписки используются ССШ, но ключи шифрования в этих СШ периодически обновляются, и для зашифрованной пересылки обновленных ключей используются АСШ.

Ключи, используемые в АСШ, имеют гораздо более высокую криптографическую стойкость чем ключи в ССШ, их иногда называют **долговременными ключами**, потому что они могут использоваться в течение продолжительного времени. Ключи, используемые в ССШ, как правило, используются в течение одного сеанса связи, их принято называть **сеансовыми ключами**. После завершения сеанса связи они уничтожаются и заменяются на новые сеансовые ключи. Генерация и обновление сеансовых ключей осуществляется в соответствии с протоколами, некоторые из которых излагаются в настоящей главе.

7.1 Протокол Диффи–Хеллмана и его обобщения

Излагаемые в этом пункте протоколы генерации ключей основаны на известном протоколе Диффи–Хеллмана генерации секретного ключа по

открытому каналу связи. Во всех этих протоколах предполагается, что p – простое число, и $g \in \mathbf{Z}_p^*$ – примитивный элемент. Все вычисления - в $\mathbf{Z}_p$.

7.1.1 Протокол Диффи-Хеллмана

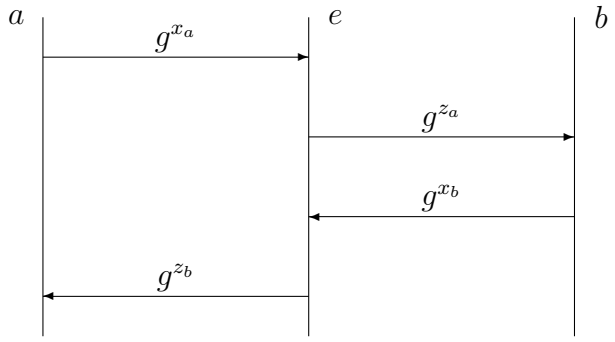
В этом протоколе участвуют два агента – a и b, их задача – создать секретный СК k для связи друг с другом, изначально не имея никакой секретной информации.

Протокол имеет следующий вид:

- $a \to b : y_a := g^{x_a}$, где $x_a \underset{r}{\in} \mathbf{Z}_{p-1}$
- $b \to a : y_b := g^{x_b}$, где $x_b \underset{r}{\in} \mathbf{Z}_{p-1}$

после чего a вычисляет ключ k как $y_b^{x_a}$, а b – как $y_a^{x_b}$.

Этот протокол уязвим к атаке активным противником e, который может перехватывать пересылаемые сообщения и заменять их на свои. Атака заключается в том, что противник e участвует одновременно в двух сеансах протокола Диффи-Хеллмана, и в диалоге с b он выдает себя за a, а в диалоге с a – за b, данные сеансы можно изобразить диаграммой



(т.е. противник e подменяет g^{x_a} и g^{x_b} на g^{z_a} и g^{z_b} соответственно, где $z_a, z_b \underset{r}{\in} \mathbf{Z}_{p-1}$). После этого он вычисляет два ключа: $k_a := (g^{x_a})^{z_b}$ (для переписки с a) и $k_b := (g^{x_b})^{z_a}$ (для переписки с b).

Можно определить аналогичный протокол создания общего секретного ключа и для большего числа агентов. Например, для трех агентов соответствующий протокол выглядит следующим образом:

- $a \to b : y_a := g^{x_a}$, где $x_a \underset{r}{\in} \mathbf{Z}_{p-1}$
- $b \to c : y_b := g^{x_b}$, где $x_b \underset{r}{\in} \mathbf{Z}_{p-1}$

- $c \to a : y_c := g^{x_c}$, где $x_c \underset{r}{\in} \mathbf{Z}_{p-1}$
- $a \to b : z_c := y_c^{x_a}$,
- $b \to c : z_a := y_a^{x_b}$,
- $c \to a : z_b := y_b^{x_c}$,

после чего a, b, c вычисляют $k = z_b^{x_a} = z_c^{x_b} = z_a^{x_c}$.

7.1.2 Протоколы STS и MTI

Излагаемые в этом пункте протоколы представляют собой добавление к протоколу Диффи-Хеллмана средств, предотвращающих описанную выше атаку. Данные протоколы решают ту же задачу, что протокол Диффи-Хеллмана.

1. **Протокол STS** (его название является аббревиатурой от словосочетания Station-To-Station) имеет вид
   - $a \to b : y_a := g^{x_a}$, где $x_a \underset{r}{\in} \mathbf{Z}_{p-1}$
   - $b \to a : (y_b, k_b(\langle y_a, y_b \rangle_b))$, где
     $$y_b := g^{x_b}, \quad k_b := y_a^{x_b}, \quad x_b \underset{r}{\in} \mathbf{Z}_{p-1}$$
   - $a \to b : k_a(\langle y_a, y_b \rangle_a)$, где $k_a := (y_b)^{x_a} \ \ (= k_b)$.
2. **Протокол MTI** (его название является аббревиатурой от фамилий его создателей – Мацумото, Такашима, Имаи) предполагает, что агенты a и b используют следующие закрытые и открытые ключи:
   $$a^- = z_a, \ \ a^+ = g^{z_a}, \quad \text{где } z_a \in \mathbf{Z}_{p-1}$$
   $$b^- = z_b, \ \ b^+ = g^{z_b}, \quad \text{где } z_b \in \mathbf{Z}_{p-1}$$

   Протокол имеет следующий вид:
   - $a \to b : y_a := g^{x_a}$, где $x_a \underset{r}{\in} \mathbf{Z}_{p-1}$
   - $b \to a : y_b := g^{x_b}$, где $x_b \underset{r}{\in} \mathbf{Z}_{p-1}$
   - a вычисляет $k_a = y_b^{z_a}(b^+)^{x_a}$
   - b вычисляет $k_b = y_a^{z_b}(a^+)^{x_b} \quad (= k_a)$.

7.1.3 Генерация ключа несколькими агентами

В этом пункте рассматривается протокол, решающий ту же задачу, которую решает протокол Диффи-Хеллмана, для случая n агентов, которых мы будем обозначать записями a_0, ..., a_{n-1}. В описании этого протокола операции, указанные в индексах этих агентов, выполняются по mod n.

Протокол имеет следующий вид:

- $a_i \to \{a_{i-1}, a_{i+1}\}: \quad y_i := g^{x_i}$, где $x_i \underset{r}{\in} \mathbf{Z}_{p-1}$
- $a_i \to \{a_0, \ldots, a_{n-1}\}: \quad u_i := \left(\frac{y_{i+1}}{y_{i-1}}\right)^{x_i}$
- $\forall\, i = 0, \ldots, n-1 \quad a_i$ вычисляет

$$k_i := y_{i-1}^{nx_i} u_i^{n-1} u_{i+1}^{n-2} \ldots u_{i+n-2}^{1}.$$

Нетрудно доказать, что $k_0 = \ldots = k_{n-1} = g^{\sum\limits_{i=0}^{n-1} x_i x_{i+1}}$.

7.2 Обновление сеансового ключа

Агенты a и b имеют общий сеансовый ключ k, и хотят заменить его на новый сеансовый ключ k'.

7.2.1 Обновление без аутентификации

Можно обновлять СК по одному из следующих протоколов:

1. $a \to b : k(k', b, t)$
2. $a \to b : k' \oplus h(k, b, t)$
3. $a \to b : b^{+}(k', a, t)$

7.2.2 Обновление с аутентификацией

Для обновления СК можно также использовать следующие протоколы:

1. С односторонней аутентификацией:

   (a)
   - $b \to a : r$
   - $a \to b : k(k', b, r)$

(b)
- $b \to a : b, r$
- $a \to b : k(k', r)$
- $b \to a : k'(r)$

2. С двусторонней аутентификацией (Andrew Secure RPC Handshake):

   - $b \to a : b, k(r)$
   - $a \to b : k(r+1, r')$
   - $b \to a : k(r'+1)$
   - $a \to b : k(k', r, r'')$

   Если последнее сообщение будет иметь вид $k(k', r'')$, то на данный КП будет возможна атака путем посылки уже переданного сообщения (replay attack).

7.2.3 Протоколы EKE обновления сеансового ключа

Для решения задачи обновления СК можно также использовать протоколы **EKE** (Encrypted Key Exchange). В обоих протоколах производится взаимная аутентификация агентов, для чего a и b создают нонсы r_a и r_b соответственно.

1. В первом протоколе a создает новый сеансовый ключ k' и передает его b. Агенты используют общую АСШ.

   Протокол имеет следующий вид:

   - $a \to b : k(b^+(k'))$
   - $b \to a : k'(r_b)$
   - $a \to b : k'(r_a, r_b)$
   - $b \to a : k'(r_a)$

2. Во втором протоколе a и b порождают новый сеансовый ключ k' совместно. Все используемые ими значения являются элементами $\mathbf{Z}_p$, где p – простое число, и все вычисления выполняются в $\mathbf{Z}_p$. Ниже $g \in \mathbf{Z}_p^*$ – примитивный элемент, и

$$a^- = x_a \underset{r}{\in} \mathbf{Z}_{p-1}, \quad a^+ = y_a := g^{x_a},$$
$$b^- = x_b \underset{r}{\in} \mathbf{Z}_{p-1}, \quad b^+ = y_b := g^{x_b}$$

   Протокол имеет следующий вид:

- $a \to b : a, y_a$
- $b \to a : k(y_b), k'(r_b)$, где $k' := {y_a}^{x_b}$
- $a \to b : k'(r_a, r_b)$
- $b \to a : k'(r_a)$

Можно заменить последние 2 действия на

- $a \to b : k'(x_a, r_a)$
- $b \to a : k'(x_a, x_b, r_b)$

и в качестве нового СК использовать $h(r_a) \oplus h(r_b)$, где h – ХФ.

7.3 Создание общего ключа с использованием нескольких открытых каналов связи

В этом пункте рассматривается задача создания агентами a и b общего ключа при следующих условиях:

- a и b могут использовать n открытых каналов связи (будем считать, что эти каналы занумерованы числами $1, \ldots, n$),
- некоторые из этих каналов контролируются активным противником, который может искажать передаваемые сообщения,
- число t каналов, которые контролируются противником, не превосходит $\frac{n-1}{2}$,
- номера контролируемых каналов неизвестны.

Один из способов создать общий ключ в этой ситуации заключается в следующем.

1. a и b выбирают простое число $p > n$, так, чтобы размер двоичной записи p был больше размера создаваемого ключа (т.е. ключ, который должен быть создан, можно было бы рассматривать как элемент $\mathbf{Z}_p$).
2. a генерирует
   - ключ $k \in \mathbf{Z}_p$, и
   - полином $f \in \mathbf{Z}_p[x]$ степени t, такой, что $k = f(0)$.

3. $\forall\, i = 1, \ldots, n$ по каналу номер i происходит передача

$$a \to b : f(i).$$

4. Обозначим записями $m_1, \ldots, m_n$ те сообщения, которые b получил по каналам $1, \ldots, n$ соответственно. Возможно, что некоторые из этих сообщений отличаются от соответствующих сообщений, которые посылал a. По условию, число неправильно переданных сообщений не превосходит t. Из неравенства $t \leq \frac{n-1}{2}$ следует, что число правильно переданных сообщений $\geq t + 1$.

5. b ищет полином $g \in \mathbf{Z}_p[x]$ степени t, такой, что

$$\forall\, i = 1, \ldots, n \quad g(i) = m_i \tag{7.1}$$

   Поскольку полином степени t однозначно определяется своими значениями в $t + 1$ точке, то в том случае, когда b может построить полином g со свойством (7.1), имеет место равенство $g = f$. В этом случае b может вычислить $k := g(0)$.

6. Если b не может построить полином g со свойством (7.1), то выполняются следующие действия.

   - b посылает a по всем каналам сообщение

$$\{(i, m_i) \mid i = 1, \ldots, n\}.$$

     Возможно, что сообщения, которые получит a по разным каналам, будут различными, но количество сообщений, переданных без искажений, будет $\geq \frac{n+1}{2}$. Поэтому a может установить, какие именно сообщения переданы с искажениями.

   - a посылает b по всем каналам сообщение

$$\{i \mid f(i) \neq m_i\} \tag{7.2}$$

   - b определяет, какие из присланных сообщений (7.2) получены без искажений (их будет $\geq \frac{n+1}{2}$), удаляет из списка $m_1, \ldots, m_n$ неверные значения, и строит полином g по оставшимся значениям.

7.4 Распределение ключевой информации

Для уменьшения объёма хранимой и передаваемой информации при генерации ключей используются схемы распределения **ключевой информации (КИ)**.

Задача распределения КИ заключается в следующем. Имеются некоторая СШ Σ и n агентов

$$a_1, \ldots, a_n \quad (n > 2). \tag{7.3}$$

Каждый из этих агентов a_i должен получить КИ $\Im_i$, используя которую, он может сгенерировать ключ k_{ij} СШ Σ для связи с каждым другим агентом a_j из (7.3).

В этом параграфе мы рассмотрим одну из схем распределения КИ, которая называется **схемой Блома**.

Пусть P – поле, элементами которого можно представлять ключи СШ Σ. Мы будем отождествлять каждый такой ключ с соответствующим ему элементом поля P.

Сопоставим каждому агенту a_i из (7.3) элемент $r_i \neq 0$ поля P, так, чтобы все элементы $r_1, \ldots, r_n$ были различны. Элементы $r_1, \ldots, r_n$ несекретны.

Для создания КИ используется матрица A над P вида

$$A = \begin{pmatrix} a_{00} & \ldots & a_{0m} \\ \ldots & \ldots & \ldots \\ a_{m0} & \ldots & a_{mm} \end{pmatrix} \quad (1 \leq m < n). \tag{7.4}$$

КИ $\Im_i$ представляет собой строку

$$(1 \; r_i \; \ldots \; r_i^m)\, A. \tag{7.5}$$

$\forall\, i, j \in \{1, \ldots, n\}$ ключ k_{ij} вычисляется по формуле

$$k_{ij} := (1 \; r_i \; \ldots \; r_i^m)\, A \begin{pmatrix} 1 \\ r_j \\ \ldots \\ r_j^m \end{pmatrix}. \tag{7.6}$$

Теорема 1. Раскрытие всех ключей любых m агентов из совокупности (7.3) не даёт возможности вычислить ключи k_{ij}, где i и j – номера агентов, ключи которых не раскрыты.

Доказательство. Мы можем считать, что те агенты, все ключи которых раскрыты, имеют номера от 1 до m, т.е. известны ключи k_{ij} $\forall\, i \in \{1, \ldots, m\}$, $\forall\, j \in \{1, \ldots, n\}$.

Пусть $i_0, j_0 \in \{m+1, \ldots, n\}$. Докажем, что $\forall\, d \in P$ существует матрица A вида (7.4) со следующими свойствами:

- значения ключей агентов с номерами от 1 до m, вычисляемые по этой матрице в соответствии с правилом (7.6), совпадают с известными значениями, и
- значение ключа $k_{i_0 j_0}$, вычисляемое по этой матрице в соответствии с правилом (7.6), равно d.

Поскольку значения k_{ij} удовлетворяют (7.6), то

$$K = PAQ \tag{7.7}$$

где символы K, P, Q обозначают матрицы

$$K = \begin{pmatrix} k_{11} & \ldots & k_{1m} & k_{1j_0} \\ \ldots & \ldots & \ldots & \ldots \\ k_{m1} & \ldots & k_{mm} & k_{mj_0} \\ k_{i_0 1} & \ldots & k_{i_0 m} & k_{i_0 j_0} \end{pmatrix}$$

$$P = \begin{pmatrix} 1 & r_1 & \ldots & r_1^m \\ \ldots & \ldots & \ldots & \ldots \\ 1 & r_m & \ldots & r_m^m \\ 1 & r_{i_0} & \ldots & r_{i_0}^m \end{pmatrix}$$

$$Q = \begin{pmatrix} 1 & \ldots & 1 & 1 \\ r_1 & \ldots & r_m & r_{j_0} \\ \ldots & \ldots & \ldots & \ldots \\ r_1^m & \ldots & r_m^m & r_{j_0}^m \end{pmatrix}$$

Т.к. P и Q обратимы, то (7.7) эквивалентно равенству

$$A = P^{-1} K Q^{-1} \tag{7.8}$$

Следовательно, определив матрицу A по формуле (7.8), где все элементы матрицы K имеют заданные значения (в частности, $k_{i_0 j_0} = d$), получим, что определённая таким образом матрица A обладает вышеупомянутыми свойствами. ■

Теорема 2.

Пусть схема распределения КИ между n агентами обладает свойством, изложенным в формулировке теоремы 1, и ключи, генерируемые этой схемой, состоят из l битов.

Тогда размер (в битах) КИ, которую должен иметь каждый агент, не меньше, чем $l(m+1)$. ■

Отметим, что размер КИ в схеме Блома равен $l(m+1)$, т.е. эта схема обеспечивает наиболее экономное распределение КИ, обладающее свойством, изложенным в формулировке теоремы 1.

7.5 Квантовая передача ключей

В протоколах квантовой передачи ключей используются два канала связи: обычный и квантовый.

- **Обычный канал связи (ОКС)** является открытым, и противник может прослушивать все сообщения, пересылаемые по этому каналу (но не может менять их порядок, а также делать вставки или удаления). Факт чтения противником того или иного сообщения невозможно обнаружить.
- По **квантовому каналу связи (ККС)** пересылаются фотоны. При прослушивании фотонов в их параметры с большой вероятностью вносятся искажения.

7.5.1 Обработка информации в ККС

Каждому фотону, передаваемому по ККС, можно сопоставить параметр, называемый **углом поляризации**. Мы будем предполагать, что углы поляризации фотонов, передаваемых по ККС, могут иметь одно из следующих четырёх значений: 0, 45, 90 и 135 градусов. Мы будем говорить, что фотоны с такими углами поляризации имеют тип $-$, $/$, $|$ и $\backslash$ соответственно.

Для определения типа фотона используется специальный прибор, называемый **фильтром**. Из законов квантовой механики следует, что если фотон с углом поляризации α, попадает на фильтр, расположенный под углом β, то этот фотон

- пройдёт через фильтр с вероятностью $\cos^2(\alpha - \beta)$
- поглотится с вероятностью $\sin^2(\alpha - \beta)$

Мы будем говорить, что фильтр

- имеет тип $+$, если он расположен под углом 0 градусов, и
- имеет тип $\times$, если он расположен под углом 45 градусов.

Фильтр называется **совместимым** с фотоном, если

- фильтр имеет тип $+$, и фотон имеет тип $-$ или $|$, или
- фильтр имеет тип $\times$, и фотон имеет тип $/$ или $\backslash$.

Определение типа фотона при помощи фильтра производится следующим образом.

- Если фильтр имеет тип $+$, то
  - если измеряемый фотон прошёл через этот фильтр, то мы будем считать, что тип этого фотона равен $-$, и
  - если измеряемый фотон поглотился этим фильтром, то мы будем считать, что тип этого фотона равен $|$.
- Если фильтр имеет тип $\times$, то
  - если измеряемый фотон прошёл через этот фильтр, то мы будем считать, что тип этого фотона равен $/$, и
  - если измеряемый фотон поглотился этим фильтром, то мы будем считать, что тип этого фотона равен $\backslash$.

Из всего сказанного выше следует, что тип фотона определяется фильтром правильно в том и только в том случае, когда этот фильтр совместим с измеряемым фотоном.

7.5.2 Протокол квантовой передачи ключа

Протокол квантовой передачи ключа от агента a к агенту b состоит из двух частей:

1. передача по ККС от a к b случайным образом сгенерированной цепочки фотонов заранее оговоренной длины n, и
2. обсуждение по ОКС.

Агент a выполняет следующие действия:

- генерирует случайным образом слово ω длины n в алфавите $\{-, |, /, \backslash\}$ (например, ω может иметь вид $| \; | \; / \; - \; - \; \backslash \; - \; | \; - \; / \;)$
- создаёт цепочку из n фотонов, причём тип каждого фотона совпадает с соответствующей компонентой слова ω, и
- посылает эту цепочку фотонов агенту b.

Агент b для обработки полученной цепочки фотонов выполняет следующие действия.

- b генерирует случайным образом слово ψ длины n в алфавите $\{+, \times\}$.

  Например, ψ может иметь такой вид:

$$\psi = \quad \times \; + \; + \; \times \; \times \; \times \; + \; \times \; + \; +$$

- b создаёт цепочку из n фильтров типа $+$ и $\times$, причём тип каждого фильтра совпадает с соответствующей компонентой слова ψ.
- b измеряет полученную цепочку фотонов при помощи своей цепочки фильтров, причём каждый полученный фотон измеряется соответствующим ему фильтром (отметим, что количество фильтров, совместимых с соответствующими им фотонами из полученной цепочки, в среднем равно $n/2$).

  Результатом измерений является слово ω' длины n в алфавите $\{-, |, /, \backslash\}$.

Далее все пересылки осуществляются по ОКС

1. $b \to a : \psi$
2. $a \to b$: номера фильтров, совместимых с соответствующими им фотонами из ω
   (в нашем примере - это 2, 6, 7, 9)
3. a оставляет в ω только те компоненты, номера которых a послал b в предыдущем действии,
   в нашем примере это будет выглядеть так:

$$* \; | \; * \; * \; * \; \backslash \; - \; * \; - \; *$$

   (символ $*$ обозначает удаляемые компоненты)
4. b делает то же самое с компонентами ω'

5. a заменяет

   - компоненты $-$ и $/$ в оставшейся части последовательности ω на бит 1, и
   - компоненты $|$ и $\backslash$ – на бит 0.

   Получившуюся последовательность битов обозначим символом k.

   В нашем примере k имеет вид

   $$(0, 0, 1, 1)$$

6. b делает те же операции со своей оставшейся частью последовательности ω'. Получившуюся последовательность битов обозначим символом k'.

Затем проверяется, было ли подслушивание в ККС:

1. $b \to a$: последовательность пар вида $(i, k'[i])$, где
   - $k'[i] = i$–й бит k',
   - количество пар в посылаемой последовательности приблизительно равно $1/3$ длины k',
   - номера тех битов k', которые включаются в посылаемую последовательность, выбираются случайным образом.
2. $a \to b : \begin{cases} 0, \text{если среди принятых пар есть пара} \\ \quad (i, k'[i]), \text{такая, что } k'[i] \neq k[i] \\ 1, \text{иначе} \end{cases}$

Если a послал b значение 1, то это означает, что подслушивания в ККС не было, и в качестве искомого ключа агенты a и b могут использовать последовательности из тех компонентов k и k', которые не участвовали в последних проверках.

Если же a послал b значение 0, то это означает, что в ККС имело место подслушивание.

Этот КП можно улучшить так, чтобы даже в случае обнаружения подслушивания a и b могли бы сгенерировать общий закрытый ключ.

Глава 8

Протоколы голосования

8.1 Понятие протокола голосования

Задача **голосования** заключается в том, что несколько агентов должны совместно выбрать решение из некоторого множества возможных решений. Каждый агент заполняет свой бюллетень, отражающий решение этого агента. Совместное решение вырабатывается путём обработки всех бюллетеней.

Например, в качестве данного решения может выступать избрание некоторого лица на какую-либо должность, в этом случае

- каждый бюллетень представляет собой список возможных кандидатов,
- заполнение бюллетеня агентом заключается в том, что агент оставляет в своём бюллетене только того кандидата, которого он хотел бы видеть избранным на эту должность,
- обработка бюллетеней заключается в подсчёте голосов, поданных за каждого кандидата,
- избранным считается тот кандидат, за которого подано наибольшее количество голосов.

Часто при процедуре голосования важно обеспечить конфиденциальность решений, принимаемых агентами (и некоторые другие условия, связанные с контролем правильности подсчёта голосов). Данная задача решается при помощи специальных КП, которые называются **КП голосования**.

Наиболее часто используются такие КП голосования, в которых

- каждый агент отсылает свой бюллетень некоторому доверенному агенту, называемому **Центральной Избирательной Комиссией (ЦИК)**, которого мы будем обозначать ниже символом s,
- ЦИК обрабатывает полученные от агентов бюллетени, и публикует результат голосования.

Условия на процедуру голосования, которые должен обеспечить КП голосования, могут иметь, например, следующий вид.

1. Голосовать могут только те агенты, которые имеют на это право (такие агенты называются **избирателями**).
2. Каждый избиратель может голосовать только один раз (т.е. может послать только один бюллетень).
3. Невозможно установить, за кого проголосовал каждый избиратель.
4. Невозможно использовать дубликат заполненного бюллетеня.
5. Невозможно изменить результат голосования каждого избирателя.
6. Каждый избиратель может проверить, что его бюллетень учтён.
7. Всем известно, кто участвовал в голосовании.

8.2 Примеры протоколов голосования

1. Простой КП голосования:
   - каждый избиратель a_i создаёт свой бюллетень v_i, шифрует его и посылает s:
   $$a_i \to s : s^+(v_i)$$
   - s вычисляет результат и публикует его.

   В данном КП не выполняются почти все вышеперечисленные условия.
2. Другой КП:
   - каждый избиратель a_i создаёт свой бюллетень v_i, подписывает его, шифрует, и посылает s:
   $$a_i \to s : s^+\langle v_i \rangle_{a_i}$$

- s вычисляет результат и публикует его.

В данном КП условия 1 и 2 выполняются, условие 3 не выполняется, условие 4 выполняется, и т.д.

8.3 КП голосования с использованием слепой ЭП

Предположим, что целью голосования является выбор одной из двух альтернатив. Каждый избиратель a_i должен иметь два бюллетеня (каждый из которых соответствует некоторой альтернативе), которые подписаны s. Избиратель a_i должен выбрать один из этих бюллетеней, и послать его s. Победившей считается та альтернатива, за которую подано больше голосов.

В излагаемом ниже КП взаимодействие избирателя a_i с s осуществляется следующим образом.

1. a_i создает 10 троек вида $(v_j^{(1)}, v_j^{(2)}, r_j)$ $(j = 1, \ldots, 10)$, где
   - $v_j^{(1)}$ – j-й бюллетень агента a_i, соответствующий первой альтернативе,
   - $v_j^{(2)}$ – j-й бюллетень агента a_i, соответствующий второй альтернативе,
   - r_j – нонс, который в данном случае рассматривается как уникальный номер a_i.
2. $a_i \to s : \{\left(\xi_j v_j^{(1)},\ \xi_j v_j^{(2)},\ \xi_j r_j\right) \mid j = 1, \ldots, 10\}$, где ξ_j – маскирующие множители, причём операция умножения на маскирующий множитель коммутирует с функцией вычисления ЭП $\langle m \rangle_s$.
3. s проверяет в своей базе данных, что раньше от a_i не было сообщений, и заносит имя a_i в свою базу данных.
4. s выбирает 9 сообщений из полученных 10, просит у a_i их маскирующие множители, и открывает их.
5. Если все сообщения корректны, s подписывает в невскрытом корртеже $\left(\xi v^{(1)},\ \xi v^{(2)},\ \xi r\right)$ все 3 компоненты, и посылает результат a_i:

$$s \to a_i : \left(\langle \xi v^{(1)} \rangle_s,\ \langle \xi v^{(2)} \rangle_s,\ \langle \xi r \rangle_s\right)$$

6. a_i удаляет ξ, выбирает нужный $\langle v^{(\cdot)} \rangle_s$, и

$$a_i \overset{\circ}{\to} s : s^+\Big(\langle v^{(\cdot)} \rangle_s,\ \langle r \rangle_s\Big)$$

($\overset{\circ}{\to}$ – анонимная посылка)

7. s проверяет уникальность нонса (если нонс новый, то заносит его в свою БД), после чего вычисляет результат, и публикует

   - результат голосования, и
   - нонсы голосовавших, вместе с волеизъявлением каждого нонса.

Недостаток: нечестный s может фальсифицировать выборы, генерируя бюллетени и посылая их самому себе.

8.4 Голосование с ЦИК + ЦУР

Для того чтобы противодействовать возможной нечестности со стороны ЦИК, можно использовать второго доверенного посредника, называемого **Центральным Управлением Регистрации (ЦУР)**, и обозначаемого записью s'. Необходимым условием корректности нижеследующего КП является отсутствие обмена информацией между ЦУР и ЦИК.

Взаимодействие между a_i, s и s' осуществляется следующим образом. В каждой из нижеследующих пересылок пересылаемые сообщения должны быть подписаны и зашифрованы открытым ключом получателя.

1. $a_i \to s'$: просьба дать регистрационный номер
2. $s' \to a_i : r_i$ (случайный рег. номер)
3. $s' \to s : \mathcal{R} :=$ список всех выданных рег. номеров
4. $a_i \to s : (ID_i, r_i, v_i)$, где ID_i – случайный идентификатор избирателя a_i, v_i – бюллетень избирателя a_i
5. s проверяет: $r_i \in \mathcal{R}$? Если это верно, то
   - $\mathcal{R} := \mathcal{R} \setminus \{r_i\}$
   - $\Im := \Im \sqcup \{ID_i\}$ (в начале работы $\Im = \emptyset$)
6. после получения всех бюллетеней s публикует результат, и список записей вида $(ID_i,\ v_i)$.
7. s' публикует список зарегистрированных a_i.

8.5 Улучшенный КП голосования

Данный КП удовлетворяет условиям 1 - 6, а также обладает следующим свойством: если избиратель a_i обнаружит, что его бюллетень был заполнен или обработан неправильно, то он может переголосовать.

Подготовка:

1. s публикует список всех агентов, имеющих право голосовать
2. $a_i \to s:$ намерение голосовать
3. s публикует список избирателей, собирающихся принять участие в выборах
   (это мешает s включать поддельные бюллетени)

Голосование (ниже $\overset{\circ}{\to}$ – анонимная передача):

1. $s \overset{\circ}{\to} a_i : ID_i$ (идентификационный номер)
2. $a_i \overset{\circ}{\to} s: \left(ID_i,\ a_i^+(ID_i, v_i)\right)$
3. s публикует $a_i^+(ID_i, v_i)$ (это позволяет a_i проверить, что s корректно учёл его v_i)
4. $a_i \to s : ID_i, a_i^-$
5. s расшифровывает бюллетени и обрабатывает их
6. s публикует результаты голосования, и все
$$v_i,\ a_i^+(ID_i, v_i)$$
7. Если a_i обнаружил, что его v_i учтён неверно, то
$$a_i \to s: \left(ID_i,\ a_i^+(ID_i, v_i),\ a_i^-\right)$$
8. Если a_i хочет изменить выбор с v_i на v_i', то
$$a_i \to s: \left(ID_i,\ a_i^+(ID_i, v_i'),\ a_i^-\right)$$

Если на этапе 1 голосования s обнаруживает, что два избирателя получили одинаковые ID, то s

- генерирует новый ID',

- выбирает одного из избирателей (=: a_i) с этим ID, и
- публикует $ID', a_i^+(ID, v_i)$

a_i узнаёт о путанице, и повторно отсылает свой v_i (этап 2 голосования) с новым идентификационным номером ID'.

Недостатки: преступная ЦИК может

- воспользоваться бюллетенями избирателей, которые зарегистрировались, но не голосовали
- безнаказанно "потерять" некоторые бюллетени

8.6 Выборы без ЦИК

Для простоты мы рассмотрим случай, когда в голосовании участвуют 4 избирателя: a_1, a_2, a_3, a_4. Все избиратели используют одну и ту же асимметричную ШС.

1. $\forall\, i \;\; a_i \to a_1 : m_i$, где

$$\begin{aligned} m_i &:= a_1^+(u_{i1}, r_{i1}) \\ u_{i1} &:= a_2^+(u_{i2}, r_{i2}) \\ u_{i2} &:= a_3^+(u_{i3}, r_{i3}) \\ u_{i3} &:= a_4^+(u_{i4}, r_{i4}) \\ u_{i4} &:= a_1^+(v_{i1}) \\ v_{i1} &:= a_2^+(v_{i2}) \\ v_{i2} &:= a_3^+(v_{i3}) \\ v_{i3} &:= a_4^+(v_i, r_{i5}) \end{aligned}$$

   (где v_i – бюллетень избирателя a_i)

2. $a_1 \to a_2 : \{u_{i1} \mid i = 1, \ldots, 4\}$
3. $a_2 \to a_3 : \{u_{i2} \mid i = 1, \ldots, 4\}$,
4. $a_3 \to a_4 : \{u_{i3} \mid i = 1, \ldots, 4\}$
5. $a_4 \to a_1 : \{u_{i4} \mid i = 1, \ldots, 4\}$
6. $a_1 \to \{a_2, a_3, a_4\} : \{\langle v_{i1} \rangle_{a_1} \mid i = 1, \ldots, 4\}$

7. $a_2 \to \{a_1, a_3, a_4\} : \{\langle v_{i2} \rangle_{a_2} \mid i = 1, \ldots, 4\}$

8. $a_3 \to \{a_1, a_2, a_4\} : \{\langle v_{i3} \rangle_{a_3} \mid i = 1, \ldots, 4\}$

9. $a_4 \to \{a_1, a_2, a_3\} : \{\langle a_i, r_{i5} \rangle_{a_4} \mid i = 1, \ldots, 4\}$

Каждый раз, когда избиратель получает набор сообщений, одной из компонентов которых является нонс, он проверяет наличие среди полученных сообщений такого, в котором присутствует его нонс.

8.7 Числовой протокол голосования

В данном протоколе участвуют n избирателей $a_1, \ldots, a_n$ и доверенный посредник s. Участники протокола выбирают

- простое число p, и
- примитивные элементы $g_1, g_2 \in \mathbf{Z}_p^*$.

Все вычисления и сравнения выполняются по mod p.

Для каждого избирателя a_i $(i = 1, \ldots, n)$

- голосом избирателя a_i является число $x_i \in \mathbf{Z}_p$, и
- его избирательный бюллетень имеет вид $y_i \stackrel{\text{def}}{=} g_2^{x_i}$.

Доверенный посредник s использует закрытый ключ $x \underset{r}{\in} \mathbf{Z}_p^*$ и открытый ключ $y \stackrel{\text{def}}{=} g_1^x$.

Результат голосования представляет собой число

$$z \stackrel{\text{def}}{=} \sum_{i=1}^{n} x_i.$$

Протокол имеет следующий вид:

- $a_i \to s : (u_i, v_i)$, где $u_i := g_1^{r_i}$, $v_i := y^{r_i} y_i$, $r_i \underset{r}{\in} \mathbf{Z}_{p-1}$
- s ищет z перебором из уравнения

$$g_2^z = \prod_{i=1}^{n} \frac{v_i}{u_i^x}. \tag{8.1}$$

Отметим, что (8.1) равносильно равенству $u^x = v$, где

$$u := \prod_{i=1}^{n} u_i, \quad v := \frac{\prod\limits_{i=1}^{n} v_i}{g_2^z}.$$

Таким образом, проверка правильности вычисления z сводится к доказательству истинности равенства $u^x = v$.

Для интерактивного доказательства истинности равенства $u^x = v$ (напомним, что x – секретное значение, открытым является $y = g_1^x$) можно использовать **протокол Шаума-Педерсена**, один раунд которого имеет следующий вид:

- $s \to b : (h_g, h_u) := (g^i,\ u^i)$, где $i \underset{r}{\in} \mathbf{Z}_{p-1}$
- $b \to s : j \underset{r}{\in} \mathbf{Z}_{p-1}$
- $s \to b : k := \ i + jx$
- $b : \left[\!\!\left[\begin{array}{l} g^k = h_g y^j \\ u^k = h_u v^j \end{array} \right]\!\!\right]$ доказательство принимается.

Глава 9

Электронная коммерция

9.1 Протоколы электронной коммерции

В протоколах **электронной коммерции (ПЭК)** некоторые взаимодействия между участниками имеют коммерческий характер. Сообщения, передаваемые в таких взаимодействиях, являются электронными аналогами бумажных денег и называются **электронными банкнотами (ЭБ)**.

Свойства, которыми могут обладать ЭБ, могут иметь, например, следующий вид.

1. По ЭБ невозможно вычислить тех участников, которые использовали её в своих платежах (анонимность).

2. ЭБ можно передавать другим участникам, которые могут использовать её по своему усмотрению.

3. ЭБ невозможно использовать более одного раза путём изготовления дубликата.

Ниже в этой главе некоторые участники ПЭК обозначаются специальными символами:

- символом b обозначается банк, и
- символом t обозначается продавец.

9.2 Примеры ПЭК

- $a \to b : (\xi_1 m_1, \ldots, \xi_k m_k)$, где

  - $m_1, \ldots, m_k$ – переводы на одну и ту же сумму,
  - $\xi_1, \ldots, \xi_k$ – маскирующие множители (маскировка коммутирует с ЭП $\langle m \rangle_b$)

- b выбирает $i \in \{1, \ldots, k\}$, снимает маскировку у всех переводов кроме i-го, проверяет вскрытые переводы, после чего
  - $b \to a : \langle \xi_i m_i \rangle_b$
  - b списывает со счёта a соотв. сумму.

  (в результате b не сможет определить, что $m_i \in a$)

- $a \to t : \langle m_i \rangle_b$ (платёж)
- $t \to b : \langle m_i \rangle_b$ (депонирование)
- $b \to t :$ деньги.

Недостаток: можно сделать копию ЭБ и использовать её. Чтобы этого не было, надо заменить первое действие на

$$a \to b : \xi_1(m_1, r_1), \ldots, \xi_k(m_k, r_k)$$

Во 2-м действии b проверяет уникальность всех нонсов у демаскированных переводов. Когда в 4-м действии b получает $\langle m_i, r_i \rangle_b$, он проверяет, депонировалась ли ЭБ с r_i.

Однако этот КП не может определить мошенника.

Можно заменить действие 4 на

$$t \to b : \langle m_i, r_i \rangle_b, r'$$

где нонс r' генерируется совместно a и t. Когда b получает это сообщение, он проверяет наличие r_i и r' в своей БД:

- если r' есть в БД банка, то присланное сообщение является не законной ЭБ, а копией, которую сделал t,
- если r_i есть, а r' нет, то присланное сообщение является не законной ЭБ, а копией, которую сделал a.

9.3 ПЭК, распознающий жулика

1. $a \to b : \{\xi_i(m_i, r_i, h(\alpha_{i1}), h(\alpha_{i2})) \mid i = 1, \ldots, k\}$, где
   - $m_1, \ldots, m_k$ – переводы на одну и ту же сумму,
   - $\xi_1, \ldots, \xi_k$ – маскировка, коммутирует с ЭП,
   - α_{i1} и α_{i2} – доли секрета: $id_a = \alpha_{i1} \oplus \alpha_{i2}$,
   - h – ХФ.
2. b выбирает один из переводов, вскрывает остальные и проверяет их (в том числе, просит a сообщить α_{i1} и α_{i2} у вскрытых переводов), после чего подписывает оставшийся перевод и посылает его a:
   - $b \to a : \langle \xi(m, r, h(\alpha_1), h(\alpha_2)) \rangle_b$
   - b списывает со счёта a соотв. сумму.
3. $a \to t : \langle m, r, h(\alpha_1), h(\alpha_2) \rangle_b$ (ЭБ для платежа)
4. $t \to a : e := (e_1, \ldots, e_k) \underset{r}{\in} \{1, 2\}^k$
5. $a \to t : a_e := (\alpha_{1e_1}, \ldots, \alpha_{ke_k})$
   (t принимает платёж, если среди $\{h(\alpha_{ie_i}) \mid i = 1, \ldots, k\}$ есть третья или четвёртая компонента полученной ЭБ)
6. $t \to b : (\langle m, r, h(\alpha_1), h(\alpha_2) \rangle_b, a_e)$ (депонирование)
7. b проверяет: $r \in \mathcal{R}$?
   - Если $r \notin \mathcal{R}$, то $\mathcal{R} := \mathcal{R} \cup \{r\}$, $b \to t$: деньги
   - Если $r \in \mathcal{R}$, то b сравнивает a_e в ЭБ и в своей БД.
     - Если они совпадают, то ЭБ скопировал t.
     - Если они не совпадают, то ЭБ скопировал a.
       Т.к. она уже использовалась при покупке у другого продавца t', который послал a список $e' \neq e$, то $\exists\, i$: одному из продавцов t, t' a послал α_{i1}, а другому – α_{i2}.
       b получает $id_a = \alpha_{i1} \oplus \alpha_{i2}$.

Глава 10

Другие протоколы

10.1 Неопределённая передача

Излагаемый в этом пункте протокол решает следующую задачу: агент a должен послать агенту b пару m_1, m_2 сообщений (в зашифрованном виде), но

- b должен иметь возможность получить только одно из этих сообщений, и
- a не должен знать, какое именно из этих сообщений будет получено агентом b.

Передачу подобного типа будем обозначать записью

$$a \to b : \tfrac{1}{2}\{m_1, m_2\}. \tag{10.1}$$

Предполается, что a и b используют общую АСШ, a имеет две пары (e_1, d_1), (e_2, d_2) ключей шифрования и соответствующих им ключей дешифрования в этой СШ, и b имеет пару (e, d) аналогичных ключей.

КП, реализующий передачу (10.1), имеет следующий вид:

- $a \to b : e_1, e_2$
- $b \to a : e_i(e)$, где $i \underset{r}{\in} \{1, 2\}$
- $a \to b : \Big(d_1(e_i(e))(m_1),\ d_2(e_i(e))(m_2)\Big)$
- $a \to b : d_1, d_2$ (для проверки честности)

Данный протокол используется как составная часть других протоколов.

10.2 Протокол заказного письма

В этом пункте излагается протокол, предназначенный для доставки от агента a агенту b сообщения m, причём должны быть выполнены следующие условия:

- b должен получить возможность прочитать m только после того, как он пришлёт a расписку в том, что он получил сообщение от a,
- b должен послать расписку в получении m только после того, как он удостоверится в возможности прочитать m.

Описываемый ниже протокол для решения этой задачи называется **протоколом заказного письма**. Первое действие данного протокола заключается в том, что a посылает b сообщение m, зашифрованное на ключе k, а затем

- постепенно пересылает b информацию, по которой можно построить ключ k,
- так же постепенно получая от него информацию, по которой можно построить искомую расписку (будем обозначать её символом v).

Протокол имеет следующий вид:

1. $a \to b : k(m)$
2. a создает n пар ключей ССШ $\{(\alpha_i^l, \alpha_i^r) \mid i = 1, \ldots, n\}$, где
$$\forall\, i = 1, \ldots, n \quad k = \alpha_i^l \oplus \alpha_i^r$$
3. $a \to b : \{\left(\alpha_i^l(0), \alpha_i^r(0)\right) \mid i = 1, \ldots, n\}$
4. b создает
   - n пар ключей ССШ $\{(\beta_i^l, \beta_i^r) \mid i = 1, \ldots, n\}$, и
   - n пар вида (v_i^l, v_i^r), где $\forall\, i = 1, \ldots, n$
   $$v_i^l = wb_i^l, \quad v_i^r = wb_i^r, \quad v = b_i^l \oplus b_i^r$$
   (w – строка, известная обоим агентам)
5. $b \to a : \{\left(\beta_i^l(v_i^l), \beta_i^r(v_i^r)\right) \mid i = 1, \ldots, n\}$
6. $\forall\, i = 1, \ldots, n \quad a \to b : \frac{1}{2}\{\alpha_i^l, \alpha_i^r\}$

7. $\forall\, i = 1, \ldots, n \quad b \to a : \frac{1}{2}\{\beta_i^l, \beta_i^r\}$
8. a и b пытаются расшифровать компоненты полученных пар, в каждой паре будет расшифрована только одна компонента
9. $\forall\, i = 1, \ldots, N$ (где N – длина ключа в используемой ССШ), $\forall j = 1, \ldots, n$
   - $a \to b : \left(i\text{-й бит } \alpha_j^l,\ i\text{-й бит } \alpha_j^r\right)$
   - $b \to a : \left(i\text{-й бит } \beta_j^l,\ i\text{-й бит } \beta_j^r\right)$

   (биты пересылаются в зашифрованном виде)
10. a и b расшифровывают оставшиеся половины полученных пар.

10.3 Протокол подписания контракта

Протокол подписания контракта (ППК) используется в тех случаях, когда участники a и b, которые не доверяют друг другу, хотят совместно подписать некоторое сообщение (называемое **контрактом**). Простое решение этой задачи, заключающееся в том, что

- один из участников подписывает контракт, после чего
- подписанный контракт пересылается другому участнику, который тоже подписывает этот контракт

не устраивает обоих участников, так как они не исключают возможности того, что партнёр может не подписать полученный контракт.

10.3.1 Протокол подписания контракта с доверенным посредником

Если для решения данной задачи можно привлечь доверенного посредника s, то соответствующий протокол имеет простой вид:

1. $a \to s : \langle m \rangle_a \qquad (m$ – это контракт)
2. $s \to b$: извещение, что он имеет $\langle m \rangle_a$
3. $b \to a : \langle m \rangle_b$
4. $a \to s$: извещение, что он имеет $\langle m \rangle_b$
5. $s \to b : \langle m \rangle_a$

10.3.2 Протокол подписания контракта без доверенного посредника

ППК без доверенного посредника имеет следующий вид:

1. a создает
   - n пар ключей ССШ $\{(\alpha_i^l, \alpha_i^r) \mid i = 1, \ldots, n\}$, и
   - n пар сообщений вида (u_i^l, u_i^r), где $\forall\, i = 1, \ldots, n$

     $$u_i^l = wa_i^l, \quad u_i^r = wa_i^r, \quad \langle m \rangle_a = a_i^l \oplus a_i^r$$

     (w – строка, известная обоим агентам)
2. b создает
   - n пар ключей ССШ $\{(\beta_i^l, \beta_i^r) \mid i = 1, \ldots, n\}$, и
   - n пар сообщений вида (v_i^l, v_i^r), где $\forall\, i = 1, \ldots, n$

     $$v_i^l = wb_i^l, \quad v_i^r = wb_i^r, \quad \langle m \rangle_b = b_i^l \oplus b_i^r$$
3. $a \to b : \{\left(\alpha_i^l(u_i^l), \alpha_i^r(u_i^r)\right) \mid i = 1, \ldots, n\}$
4. $b \to a : \{\left(\beta_i^l(v_i^l), \beta_i^r(v_i^r)\right) \mid i = 1, \ldots, n\}$

После этого выполняются шаги 6-10 протокола заказного письма.

10.4 Ограниченная передача секретов

10.4.1 Задача ограниченной передачи секретов

В этом пункте излагаются протоколы, решающие задачу **ограниченной передачи секретов (ОПС)**, которая заключается в следующем. У агента a имеются секретные битовые строки $m_1, \ldots, m_k$. Каждый из агентов $b_1, b_2, \ldots$

- хочет получить одну из этих строк, и
- не хочет сообщать a номер той строки, которую он хочет получить.

Протоколы для решения данной задачи обозначаются записью ANDOS (All-Or-Nothing Disclosure of Secrets).

В излагаемых ниже протоколах предполагается, что агенты $a, b_1, \ldots$ используют СШ RSA с таким модулем n, что секретные строки $m_1, \ldots, m_k$ можно рассматривать как двоичные записи чисел из $\mathbf{Z}_n$.

10.4.2 Протокол для честных агентов

Если все агенты $b_1, \ldots$ честны, то для решения задачи ОПС можно использовать излагаемый ниже протокол.

Пусть b_1 хочет получить секрет m_{i_1}, b_2 хочет m_{i_2}, и т.д.

Ниже все вычисления - в $\mathbf{Z}_n$, где n – модуль СШ RSA.

1. a создает пару (e, d) ключей шифрования и дешифрования соответственно в выбранной СШ RSA
2. $a \to b_j : (e, b_1, \ldots, b_k)$, где $\forall\, i = 1, \ldots, k \quad b_i := m_i^e$
3. $b_j \to a : b := b_{i_j} x^e \quad (x \underset{r}{\in} \mathbf{Z}_n^*)$
4. $a \to b_j : c := b^d = (b_{i_j} x^e)^d = b_{i_j}^d x^{ed} = (m_{i_j}^e)^d x^{ed} = m_{i_j} x$
5. b вычисляет $m_{i_j} := cx^{-1}$

Если агенты $b_1, \ldots$ нечестны и могут обмениваться друг с другом информацией, то, используя данный КП, они могут узнать больше секретов, чем им положено (соблюдая условие, что ни один из них не должен знать, какие секреты получили другие). Поэтому для ОПС нечестным агентам нужно использовать другие КП.

10.4.3 Протокол для нечестных агентов

В этом пункте излагается протокол ОПС для случая, когда число агентов, желающих получить секреты, равно двум (b_1 хочет m_{i_1}, b_2 хочет m_{i_2}). В описании этого протокола числа из $\mathbf{Z}_n$ представляются битовыми векторами одинаковой длины, которые рассматриваются как двоичные записи этих чисел (возможно дополненные слева нулями до необходимой длины), и символ $\oplus$ обозначает побитовое сложение (mod 2) этих векторов.

Протокол имеет следующий вид:

1. a создаёт 2 пары $(e_1, d_1), (e_2, d_2)$ ключей шифрования и дешифрования соответственно в выбранной СШ RSA
2. - $a \to b_1 : e_1$
   - $a \to b_2 : e_2$
3. - $b_1 \to b_2 : (u_1, \ldots, u_k)$, где $\forall\, i = 1, \ldots, k \quad u_i \underset{r}{\in} \mathbf{Z}_n$
   - $b_2 \to b_1 : (v_1, \ldots, v_k)$, где $\forall\, i = 1, \ldots, k \quad v_i \underset{r}{\in} \mathbf{Z}_n$

4.
   - $b_1 \to b_2 : v := v_{i_1} \oplus e_1(v_{i_1})$
   - $b_2 \to b_1 : u := u_{i_2} \oplus e_2(u_{i_2})$

5.
   - $b_1 \to a : (u'_1, \ldots, u'_k)$, где $\forall\, i = 1, \ldots, k \;\; u'_i := u_i \oplus u$ $(\Rightarrow \;\; u'_{i_2} = e_2(u_{i_2}))$
   - $b_2 \to a : (v'_1, \ldots, v'_k)$, где $\forall\, i = 1, \ldots, k \;\; v'_i := v_i \oplus v$ $(\Rightarrow \;\; v'_{i_1} = e_1(v_{i_1}))$

6.
   - $a \to b_1 : (m_1 \oplus v''_1, \ldots, m_k \oplus v''_k)$, где $v''_i = d_1(v'_i)$ $(\Rightarrow \;\; v''_{i_1} = v_{i_1})$
   - $a \to b_2 : (m_1 \oplus u''_1, \ldots, m_k \oplus u''_k)$, где $u''_i = d_2(u'_i)$ $(\Rightarrow \;\; u''_{i_2} = u_{i_2})$

7.
   - b_1 вычисляет $m_{i_1} := (m_{i_1} \oplus v''_{i_1}) \oplus v_{i_1}$
   - b_2 вычисляет $m_{i_2} := (m_{i_2} \oplus u''_{i_2}) \oplus u_{i_2}$

Данный протокол имеет уязвимость: a и b_1, действуя совместно, могут найти i_2.

Можно обобщить этот КП до КП ОПС нескольким агентам. Например, для трёх агентов КП ОПС имеет вид

- a создаёт 3 пары $(e_1, d_1), (e_2, d_2), (e_3, d_3)$ ключей шифрования и дешифрования соответственно в выбранной СШ RSA
- $a \to b_1 : e_2, e_3, \quad a \to b_2 : e_1, e_3, \quad a \to b_3 : e_1, e_2,$
- и т.д.

Литература

[1] **Шнайер Б.:** Прикладная криптография. – М.: Триумф, 2002.

[2] **Черемушкин А.В.:** Криптографические протоколы. Основные свойства и уязвимости. Учебное пособие. – М.: Изд. центр "Академия", 2009. – 272 с.

[3] **Menezes A.J., van Oorschot P.C., Vanstone S.A.:** Handbook of applied cryptography. – Boca Raton, New York, London, Tokyo: CRC Press, 1997.

[4] **Stinson D.R.:** Cryptography, theory and practice. – London etc., CRC Press, 1995.

[5] **Мао В.:** Современная криптография: теория и практика. – М.: Вильямс, 2005.

[6] **Столлингс В.:** Криптография и защита сетей: принципы и практика, 2-е издание. – М.: Вильямс, 2001.

[7] **Столингс В.:** Основы защиты сетей. Приложения и стандарты. – М.: Вильямс, 2002.