Криптографические протоколы

Лекция 9

Криптографические протоколы

Протокол – это порядок действий, предпринимаемых двумя или более сторонами, предназначенный для решения определенной задачи

Криптографический протокол – это абстрактный или конкретный протокол, включающий набор криптографических алгоритмов. В основе протокола лежит набор правил, регламентирующих использование криптографических преобразований и алгоритмов в информационных процессах

Криптографические протоколы делятся на **примитивные** и **прикладные:**

- **Примитивные протоколы** - решают абстрактную задачу
- **Прикладные протоколы** - строятся на базе примитивных и используются для решения конкретных задач

Классификация протоколов

Разделение секрета: 2 из 2

Задача: необходимо разделить некоторый секрет **P** между **двумя** участниками таким образом, чтобы прочитать его могли только оба участника собравшись вместе

Решение:

Трент – это посредник, которому доверяют все участники

1. **Трент** генерирует строку случайных бит **R**, такой же длины, что и секрет **P**
2. **Трент** выполняет XOR над **P** и **R**, создавая **S**

$$\mathbf{S} = \mathbf{R} \oplus \mathbf{P}$$

3. **Трент** передает **Алисе R**, а **Бобу - S**
4. Чтобы получить сообщение, **Алисе** и **Бобу** нужно выполнить единственное действие:

$$\mathbf{R} \oplus \mathbf{S} = \mathbf{P}$$

Разделение секрета: N из N

Задача: необходимо разделить некоторый секрет **P** между **N** участниками таким образом, чтобы прочитать его могли только все участники, собравшись вместе

Решение:

1. **Трент** генерирует набор строк случайных бит $\mathbf{R_1}, \mathbf{R_2}, \ldots, \mathbf{R_{N-1}}$ такой же длины, что и секрет **P**
2. **Трент** выполняет XOR над **P** и всеми $\mathbf{R_i}$ $(1 \le i \le N-1)$, создавая **S**

$$\mathbf{S} = \mathbf{R_1} \oplus \mathbf{R_2} \oplus \ldots \oplus \mathbf{R_{N-1}} \oplus \mathbf{P}$$

3. **Трент** передает первому участнику **S**, второму участнику - $\mathbf{R_1}$, третьему - $\mathbf{R_2}$ и т.д.
4. Чтобы получить сообщение все участники должны выполнить единственное действие:

$$\mathbf{S} \oplus \mathbf{R_1} \oplus \mathbf{R_2} \oplus \ldots \oplus \mathbf{R_{N-1}} = \mathbf{P}$$

Разделение секрета: M из N

Задача: необходимо разделить некоторый секрет **P** между **N** участниками таким образом, чтобы прочитать его могли любые **M** (**M** < **N**) участников, собравшись вместе

Решение:

- Разделение по схеме **Шамира** (использование интерполяционных полиномов Лагранжа)
- Схема **Асмута-Блума**

Полином Лагранжа

- Пусть имеется некоторая исходная функция **f(x)**, с помощью которой определены **m** точек **(x_i, y_i)**
- Тогда можно подобрать полином степени **m-1**, который будет проходить через все точки и максимально близко описывать исходную функцию
- Интерполяционный полином **L(x)** определяется формулой:

$$f(x) \approx L(x) = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \cdots + a_2x^2 + a_1x + a_0 = \sum_{i=0}^{m-1} y_i l_i(x)$$

где: a_i – коэффициенты полинома Лагранжа

y_i – значения исходной функции в i-ой точке

$l_i(x)$ – базисные полиномы, определяемые по формуле:

$$l_i(x) = \prod_{j=0, j\neq i}^{m-1} \frac{x - x_j}{x_i - x_j}$$

x_i, x_j – значения аргумента в i-ой и j-ой точках

Полином Лагранжа

Пример подбора полинома Лагранжа:

- Исходная функция f(x) = sin (x^2)
- Точки исходной функции:
  - $X_0 = -2$, $f(x_0) = -0{,}7568$
  - $X_1 = -1$, $f(x_1) = -0{,}8415$
  - $X_2 = 0$, $f(x_2) = 0$
  - $X_3 = 1$, $f(x_3) = 0{,}8415$
  - $X_4 = 2$, $f(x_4) = 0{,}7568$
- Определение базисных полиномов:

$$l_0(x) = \prod_{j=1}^{4} \frac{x - x_j}{x_0 - x_j} = \frac{x - x_1}{x_0 - x_1} \cdot \frac{x - x_2}{x_0 - x_2} \cdot \frac{x - x_3}{x_0 - x_3} \cdot \frac{x - x_4}{x_0 - x_4} = \frac{x+1}{-2+1} \cdot \frac{x-0}{-2-0} \cdot \frac{x-1}{-2-1} \cdot \frac{x-2}{-2-2} = \frac{x+1}{-1} \cdot \frac{x}{-2} \cdot \frac{x-1}{-3} \cdot \frac{x-2}{-4} = \frac{1}{24} \cdot (x^4 - 2x^3 - x^2 + 2x)$$

$$l_1(x) = \prod_{j=0, j\neq 1}^{4} \frac{x - x_j}{x_1 - x_j} = \frac{x - x_0}{x_1 - x_0} \cdot \frac{x - x_2}{x_1 - x_2} \cdot \frac{x - x_3}{x_1 - x_3} \cdot \frac{x - x_4}{x_1 - x_4} = \frac{x+2}{-1+2} \cdot \frac{x-0}{-1-0} \cdot \frac{x-1}{-1-1} \cdot \frac{x-2}{-1-2} = \frac{x+2}{1} \cdot \frac{x}{-1} \cdot \frac{x-1}{-2} \cdot \frac{x-2}{-3} = \frac{1}{-6} \cdot (x^4 - x^3 - 4x^2 + 4x)$$

$$l_2(x) = \prod_{j=0, j\neq 2}^{4} \frac{x - x_j}{x_2 - x_j} = \frac{x - x_0}{x_2 - x_0} \cdot \frac{x - x_1}{x_2 - x_1} \cdot \frac{x - x_3}{x_2 - x_3} \cdot \frac{x - x_4}{x_2 - x_4} = \frac{x+2}{0+2} \cdot \frac{x+1}{0+1} \cdot \frac{x-1}{0-1} \cdot \frac{x-2}{0-2} = \frac{x+2}{2} \cdot \frac{x+1}{1} \cdot \frac{x-1}{-1} \cdot \frac{x-2}{-2} = \frac{1}{-4} \cdot (x^4 - 5x^2 + 4)$$

$$l_3(x) = \prod_{j=0, j\neq 3}^{4} \frac{x - x_j}{x_3 - x_j} = \frac{x - x_0}{x_3 - x_0} \cdot \frac{x - x_1}{x_3 - x_1} \cdot \frac{x - x_2}{x_3 - x_2} \cdot \frac{x - x_4}{x_3 - x_4} = \frac{x+2}{1+2} \cdot \frac{x+1}{1+1} \cdot \frac{x-0}{1-0} \cdot \frac{x-2}{1-2} = \frac{x+2}{3} \cdot \frac{x+1}{2} \cdot \frac{x}{1} \cdot \frac{x-2}{-1} = \frac{1}{-6} \cdot (x^4 + x^3 - 4x^2 - 4x)$$

$$l_4(x) = \prod_{j=0}^{3} \frac{x - x_j}{x_4 - x_j} = \frac{x - x_0}{x_4 - x_0} \cdot \frac{x - x_1}{x_4 - x_1} \cdot \frac{x - x_2}{x_4 - x_2} \cdot \frac{x - x_3}{x_4 - x_3} = \frac{x+2}{2+2} \cdot \frac{x+1}{2+1} \cdot \frac{x-0}{2-0} \cdot \frac{x-1}{2-1} = \frac{x+2}{4} \cdot \frac{x+1}{3} \cdot \frac{x}{2} \cdot \frac{x-1}{1} = \frac{1}{24} \cdot (x^4 + 2x^3 - x^2 - 2x)$$

Полином Лагранжа

- Определение интерполяционного полинома Лагранжа

$$L(x)=\sum_{i=0}^{m-1} y_i l_i(x)=\frac{y_0}{24}\cdot(x^4-2x^3-x^2+2x)+\frac{y_1}{-6}\cdot(x^4-x^3-4x^2+4x)+\frac{y_2}{-4}\cdot(x^4-5x^2+4)+\frac{y_3}{-6}\cdot(x^4+x^3-4x^2-4x)+\frac{y_4}{24}\cdot(x^4+2x^3-x^2-2x)=$$

$$\frac{-0.7568}{24}\cdot(x^4-2x^3-x^2+2x)+\frac{0.8415}{-6}\cdot(x^4-x^3-4x^2+4x)+\frac{0}{-4}\cdot(x^4-5x^2+4)+\frac{0.8415}{-6}\cdot(x^4+x^3-4x^2-4x)+\frac{-0.7568}{24}\cdot(x^4+2x^3-x^2-2x)=-0.3436x^4+1.1851x^2$$

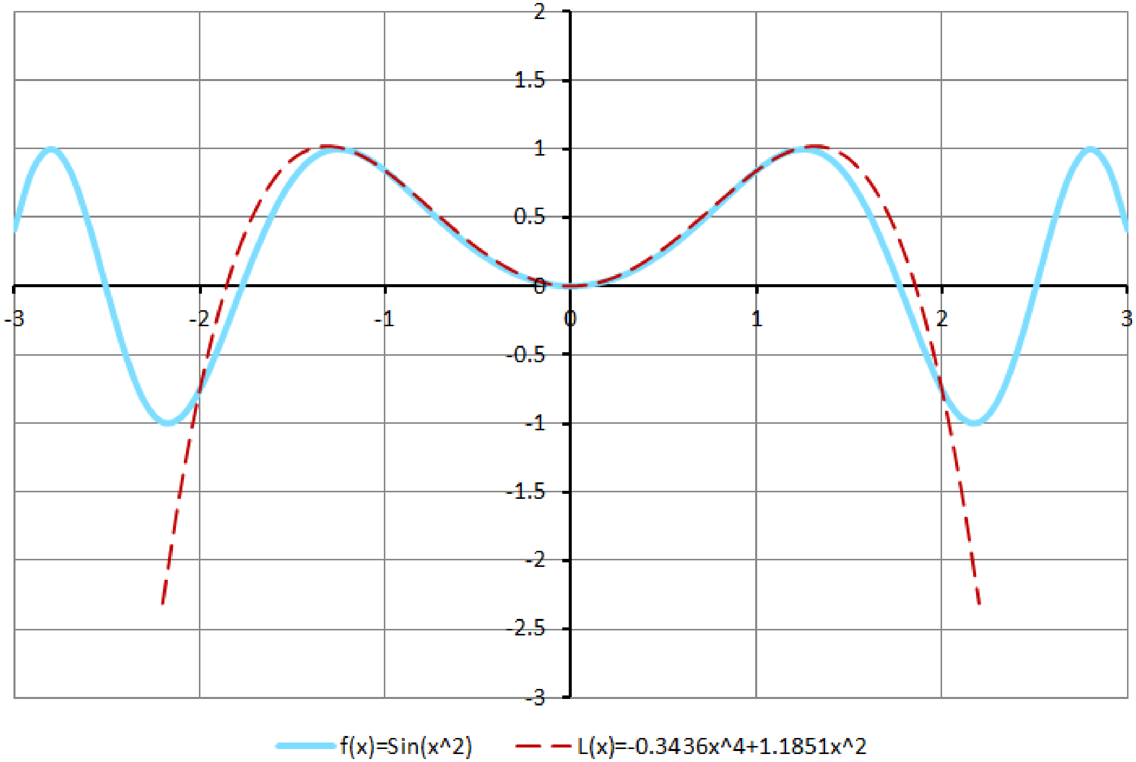


Схема Шамира

- В 1979 г. Ади Шамир предложил протокол разделения секрета с использованием полиномов, максимальная степень которых равна **m-1**. Для восстановления секрета используются формулы полинома Лагранжа
- Для разделения секрета **S**, восстанавливаемого с помощью **m** долей, используется полином степени **m-1** по модулю **p:**

$$f(x) = L(x) = (a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \cdots + a_2x^2 + a_1x + S) \bmod p$$

где: **f(x)** = **L(x)** – исходная функция и полином Лагранжа
$\mathbf{a_i}$ – целочисленные коэффициенты полинома Лагранжа
S = $\mathbf{a_0}$ – разделяемый секрет, закодированный в виде числа
p – простое число

- Коэффициенты полинома $\mathbf{a_i}$ выбираются произвольно, за исключением $\mathbf{a_0}$ = **S**
- Модуль **p** должен быть простым числом, большим секрета **S** и **n**
- Владелец секрета для $\mathbf{x_i}$ = **1..n** определяет значения полинома $\mathbf{y_i}$ = $\mathbf{f(x_i)}$ и передает пары $\mathbf{(x_i, y_i)}$ участникам
- Для восстановления секрета необходимо собрать **m** долей (пар $\mathbf{(x_i, y_i)}$) и найти значения коэффициентов интерполяционного полинома, включая секрет **S** = $\mathbf{a_0}$

Схема Шамира

Пример реализации протокола

- Секрет **S** = 11
- Количество долей, необходимых для восстановления секрета **m** = 3
- Общее количество долей **n** = 5

Процедура определения и распределения долей (выполняет владелец):

1. Выбор простого числа **p** (**p** > **n**, **p** > **S**): **p** = 59
2. Выбор произвольного многочлена степени **m**-1: $\mathbf{f(x)=a_2x^2} + \mathbf{a_1x} + \mathbf{S}$ (mod **p**)
3. Выбор произвольных констант $\mathbf{a_1}$ и $\mathbf{a_2}$: $\mathbf{a_1}$ = 23 и $\mathbf{a_2}$ = 10
4. Определение долей ($\mathbf{X_i}$, $\mathbf{Y_i}$), где $\mathbf{Y_i} = \mathbf{f(X_i)}$, $\mathbf{X_i} = \mathbf{i} + 1$

$$y_0 = (10*1^2 + 23*1 + 11) \bmod 59 = 44$$
$$y_1 = (10*2^2 + 23*2 + 11) \bmod 59 = 38$$
$$y_2 = (10*3^2 + 23*3 + 11) \bmod 59 = 52$$
$$y_3 = (10*4^2 + 23*4 + 11) \bmod 59 = 27$$
$$y_4 = (10*5^2 + 23*5 + 11) \bmod 59 = 22$$

5. Публикация **p** и распределение долей ($\mathbf{X_i}$, $\mathbf{Y_i}$) между участниками:

$$p = 59$$
$$(x_0, y_0) = (1, 44)$$
$$(x_1, y_1) = (2, 38)$$
$$(x_2, y_2) = (3, 52)$$
$$(x_3, y_3) = (4, 27)$$
$$(x_4, y_4) = (5, 22)$$

Схема Шамира

Процедура восстановления секрета:

1. Сбор m долей:

$$(x_1, y_1) = (2, 38)$$
$$(x_2, y_2) = (3, 52)$$
$$(x_4, y_4) = (5, 22)$$

2. Определение базисных полиномов

$$l_1(x) = \frac{x-3}{2-3} \cdot \frac{x-5}{2-5} = \frac{x-3}{-1} \cdot \frac{x-5}{-3} = \frac{1}{3} \cdot (x^2 - 8x + 15)$$
$$l_2(x) = \frac{x-2}{3-2} \cdot \frac{x-5}{3-5} = \frac{x-2}{1} \cdot \frac{x-5}{-2} = \frac{1}{-2} \cdot (x^2 - 7x + 10)$$
$$l_4(x) = \frac{x-2}{5-2} \cdot \frac{x-3}{5-3} = \frac{x-2}{3} \cdot \frac{x-3}{2} = \frac{1}{6} \cdot (x^2 - 5x + 6)$$

3. Определение полинома Лагранжа

$$L(x) = \left[\frac{38}{3} \cdot (x^2 - 8x + 15) + \frac{52}{-2} \cdot (x^2 - 7x + 10) + \frac{22}{6} \cdot (x^2 - 5x + 6)\right] mod\ 59$$
$$L(x) = \left[\frac{76}{6} \cdot (x^2 - 8x + 15) - \frac{156}{6} \cdot (x^2 - 7x + 10) + \frac{22}{6} \cdot (x^2 - 5x + 6)\right] mod\ 59$$
$$L(x) = \left[\frac{1}{6} \cdot (-58x^2 + 374x - 288)\right] mod\ 59$$

Схема Шамира

Процедура восстановления секрета (продолжение):

4. Определение обратного числа $\mathbf{b^{-1}}$ по модулю **p** (используется расширенный алгоритм Евклида):

$$\frac{1}{b} = \frac{1}{6}$$

$$b^{-1} = 10 \; [(6 * 10) \bmod 59 = 1]$$

5. Замена дробного множителя **1/b** на значение $\mathbf{b^{-1}}$:

$$L(x) = [10 * (-58x^2 + 374x - 288)] \bmod 59 = (-580x^2 + 3740x - 2880) \bmod 59$$

6. Приведение коэффициентов полинома и определение секрета **S**:

$$a_2 = -580 \bmod 59 = -49 \bmod 59 = 10$$

$$a_1 = 3740 \bmod 59 = 23$$

$$S = a_0 = -2880 \bmod 59 = -48 \bmod 59 = 11$$

$$L(x) = (10x^2 + 23x + 11) \bmod 59$$

Китайская теорема об остатках

Сущность китайской теоремы об остатках заключается в определении некоторого числа **S**' по набору его остатков $\mathbf{k_i}$ от деления на некоторые заданные взаимно простые числа $\mathbf{d_i}$

$$\begin{cases} S' \, mod \, d_1 = k_1 \\ S' \, mod \, d_2 = k_2 \\ \quad \ldots \\ S' \, mod \, d_m = k_m \end{cases}$$

Например, для трех пар ($\mathbf{d_i}$, $\mathbf{k_i}$) – (3, 1), (5, 3) и (8, 3) – таким числом является **S**' = 43

$$\begin{cases} 43 \, mod \, 3 = 1 \\ 43 \, mod \, 5 = 3 \\ 43 \, mod \, 8 = 3 \end{cases}$$

Чаще всего на практике для реализации используется алгоритм Гарнера: http://e-maxx.ru/algo/export_chinese_theorem

Схема Асмута-Блума

Пример реализации протокола

- Секрет **S** = 11
- Количество долей, необходимых для восстановления секрета **m** = 3
- Общее количество долей **n** = 5

Процедура определения и распределения долей (выполняет владелец):

1. Выбор простого числа **p** (**p** > **S**): **p** = 13
2. Выбор **n** взаимно простых чисел $\mathbf{d_i}$, удовлетворяющих трем условиям:
   - $\mathbf{d_i > p}$
   - $\mathbf{d_i < d_{i+1}}$
   - $\mathbf{d_1 * d_2 * \ldots * d_m < p * d_{n-m+2} * d_{n-m+3} * \ldots * d_n}$

$$d_i \in \{17, 20, 23, 29, 37\}$$
$$17 * 20 * 23 < 13 * 29 * 37$$
$$7820 < 13949$$

Схема Асмута-Блума

Процедура определения и распределения долей (продолжение):

3. Выбор произвольного числа **r**, удовлетворяющего условию: $r < \dfrac{\prod_{i=1}^{m} d_i - S}{p}$

$$r = 30$$

$$\left[30 < 600.7 = \frac{17 \cdot 20 \cdot 23 - 11}{13}\right]$$

4. Вычисление S' = S + r*p:

$$S' = 11 + 30 * 13 = 401$$

5. Определение долей ($\mathbf{d_i}$, $\mathbf{k_i}$), где $\mathbf{k_i}$ = **S'** mod $\mathbf{d_i}$:

$$k_1 = 401 \bmod 17 = 10$$
$$k_2 = 401 \bmod 20 = 1$$
$$k_3 = 401 \bmod 23 = 10$$
$$k_4 = 401 \bmod 29 = 24$$
$$k_5 = 401 \bmod 37 = 31$$

6. Публикация **p** и распределение долей ($\mathbf{d_i}$, $\mathbf{k_i}$) между участниками:

$$p = 13$$
$$(d_1, k_1) = (17, 10)$$
$$(d_2, k_2) = (20, 1)$$
$$(d_3, k_3) = (23, 10)$$
$$(d_4, k_4) = (29, 24)$$
$$(d_5, k_5) = (37, 31)$$

Схема Асмута-Блума

Процедура восстановления секрета:

1. Сбор m долей:

$$(d_2, k_2) = (20, 1)$$
$$(d_3, k_3) = (23, 10)$$
$$(d_5, k_5) = (37, 31)$$

2. Вычисление произведения **D** взаимно простых чисел $\mathbf{d_j}$:

$$D = 20 * 23 * 37 = 17020$$

3. Вычисление сомножителей $\mathbf{D_j = D / d_j}$:

$$D_1 = 17020 / 20 = 851$$
$$D_2 = 17020 / 23 = 740$$
$$D_3 = 17020 / 37 = 460$$

4. Вычисление обратных чисел $\mathbf{D_j^{-1}}$ по модулям d_j (для вычисления используется расширенный алгоритм Евклида):

$$D_1^{-1} = 11 \ [(851 * 11) \bmod 20 = 1]$$
$$D_2^{-1} = 6 \ [(740 * 6) \bmod 23 = 1]$$
$$D_3^{-1} = 7 \ [(460 * 7) \bmod 37 = 1]$$

Схема Асмута-Блума

Процедура восстановления секрета (продолжение):

5. Вычисление $\mathbf{S'} = (\Sigma\ \mathbf{k_j} * \mathbf{D_j} * \mathbf{D_j}^{-1}) \bmod \mathbf{D}$:

$$S' = (1*851*11 + 10*740*6 + 31*460*7) \bmod 17020 = 153581 \bmod 17020 = 401$$

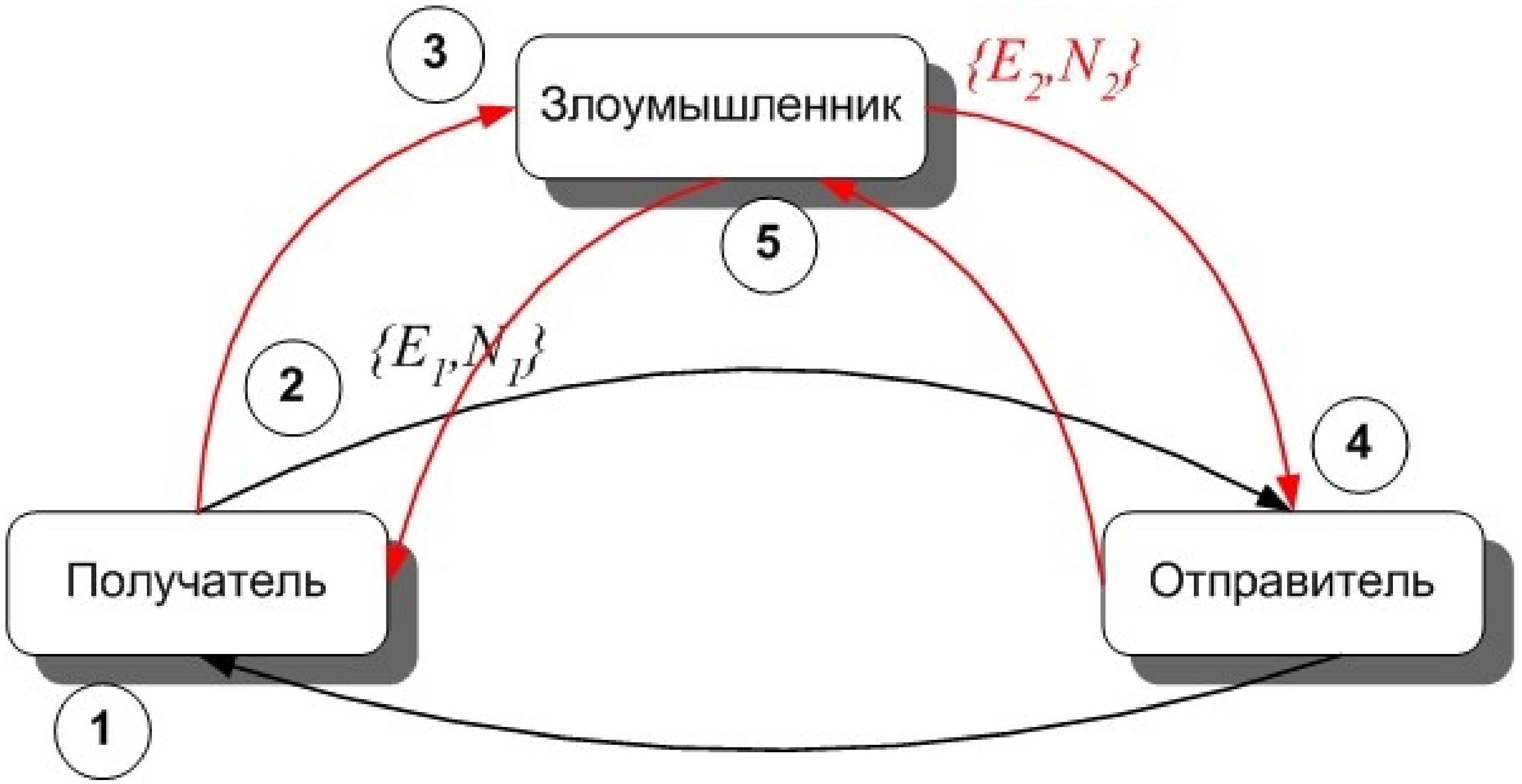
6. Определение секрета $\mathbf{S} = \mathbf{S'} \bmod \mathbf{p}$:

$$S = 401 \bmod 13 = 11$$

Другие схемы разделения секрета

1. Разделение секрета **без владельца**. Участники могут создать секрет и разделить его на доли так, что никто из них не узнает секрета, пока они совместно его не восстановят
2. Разделение секрета **без раскрытия долей** при восстановлении. В этом смысле протокол представляет собой нечто среднее между разделением секрета и тайными многосторонними вычислениями
3. Разделение секрета с **возможностью проверки корректности** отдельных долей. Каждый из участников независимо от других может проверить корректность своей доли без восстановления секрета
4. Разделение секрета с возможностью **блокирования восстановления секрета**. Каждый из участников получает две доли: «да» и «нет». Если при восстановлении секрета число долей «нет» превышает некоторое пороговое значение, то его восстановление невозможно, даже, если количества долей «да» достаточно
5. Разделение секрета с возможностью **блокирования долей**. Если после распределения долей, некоторые из участников теряют доверие, то можно блокировать их доли
6. Разделение секрета с возможностью **выявления фальшивых долей**. При восстановлении секрета возможно выявление участников, предоставивших фальшивые доли
7. Схема **группового разделения секрета**. Секрет распределяется среди участников, объединенных в **k** групп. Для восстановления секрета необходимо собрать в каждой группе нужное количество долей. Т.е. имеет место $((m_1, n_1), (m_2, n_2), ..., (m_k, n_k))$ - пороговая схема

Атака «Человек-в-середине»



Протокол «Держась за руки»

Данный протокол относится к **протоколам распределения ключей** и позволяет предотвратить атаку «**Человек-в-середине**»

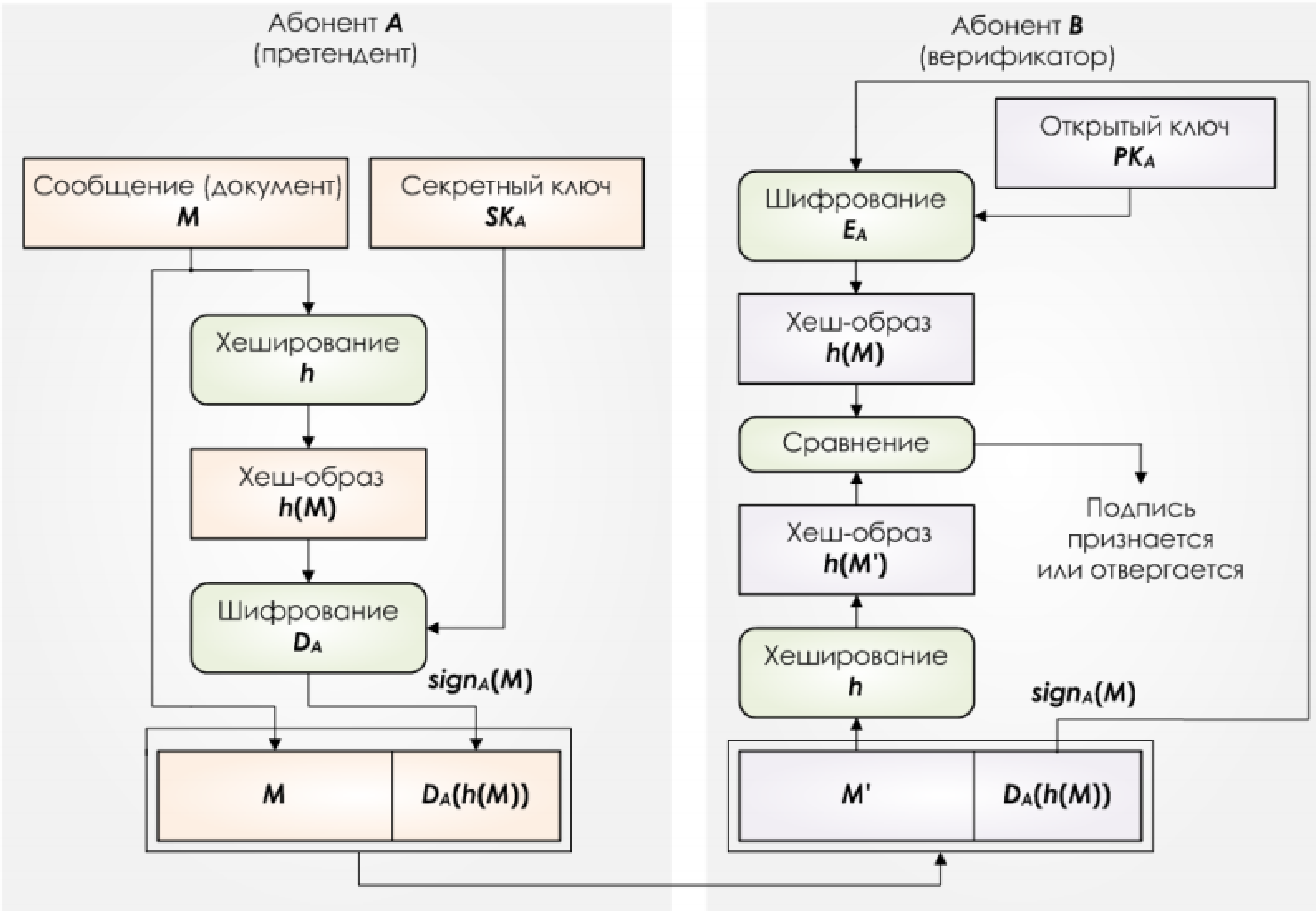
1. **Алиса** посылает **Бобу** свой открытый ключ $\mathbf{ОК_А}$
2. **Боб** посылает **Алисе** свой открытый ключ $\mathbf{ОК_Б}$
3. **Алиса** зашифровывает свое сообщение открытым ключом **Боба**:
$$\mathbf{C_А = F(M_А, ОК_Б)}$$
4. **Алиса** отправляет **Бобу** половину зашифрованного сообщения: $\mathbf{C_А^{1/2}}$
5. **Боб** зашифровывает свое сообщение открытым ключом **Алисы**:
$$\mathbf{C_Б = F(M_Б, ОК_А)}$$
6. **Боб** отправляет ей половину зашифрованного сообщения: $\mathbf{C_Б^{1/2}}$
7. **Алиса** отправляет **Бобу** 2 половину зашифрованного сообщения: $\mathbf{C_А^{2/2}}$
8. **Боб** складывает две части сообщения **Алисы** и расшифровывает его с помощью своего закрытого ключа:
$$\mathbf{M_А = F^{-1}(C_А^{1/2} + C_А^{2/2}, ЗК_Б)}$$
9. **Боб** отправляет **Алисе** вторую половину своего зашифрованного сообщения $\mathbf{C_Б^{2/2}}$
10. **Алиса** складывает две части сообщения **Боба** и расшифровывает его с помощью своего закрытого ключа:
$$\mathbf{M_Б = F^{-1}(C_Б^{1/2} + C_Б^{2/2}, ЗК_А)}$$

Протокол «Держась за руки»

- **Мэллори** может подменить открытые ключи **Алисы** и **Боба** своим ключом на этапах **(1)** и **(2)**
- Но теперь, перехватив половину сообщения **Алисы** на этапе **(4)**, он не сможет расшифровать ее своим закрытым ключом и снова зашифровать открытым ключом **Боба**. Он может создать совершенно новое сообщение и отправить половину его **Бобу**
- Перехватив половину сообщения **Боба Алисе** на этапе **(6)**, Мэллори столкнется с этой же проблемой. Он не сможет расшифровать ее своим закрытым ключом и снова зашифровать открытым ключом **Алисы**. Ему придется создать совершенно новое сообщение и отправить половину его Алисе
- К тому времени, когда он перехватит вторые половины настоящих сообщений на этапах **(7)** и **(9)**, подменять созданные им новые сообщения будет слишком поздно

Примечание:
Применение ЭЦП в протоколе обмена сеансовым ключом также позволяет избежать вскрытия «человек-в-середине»

Протокол ЭЦП

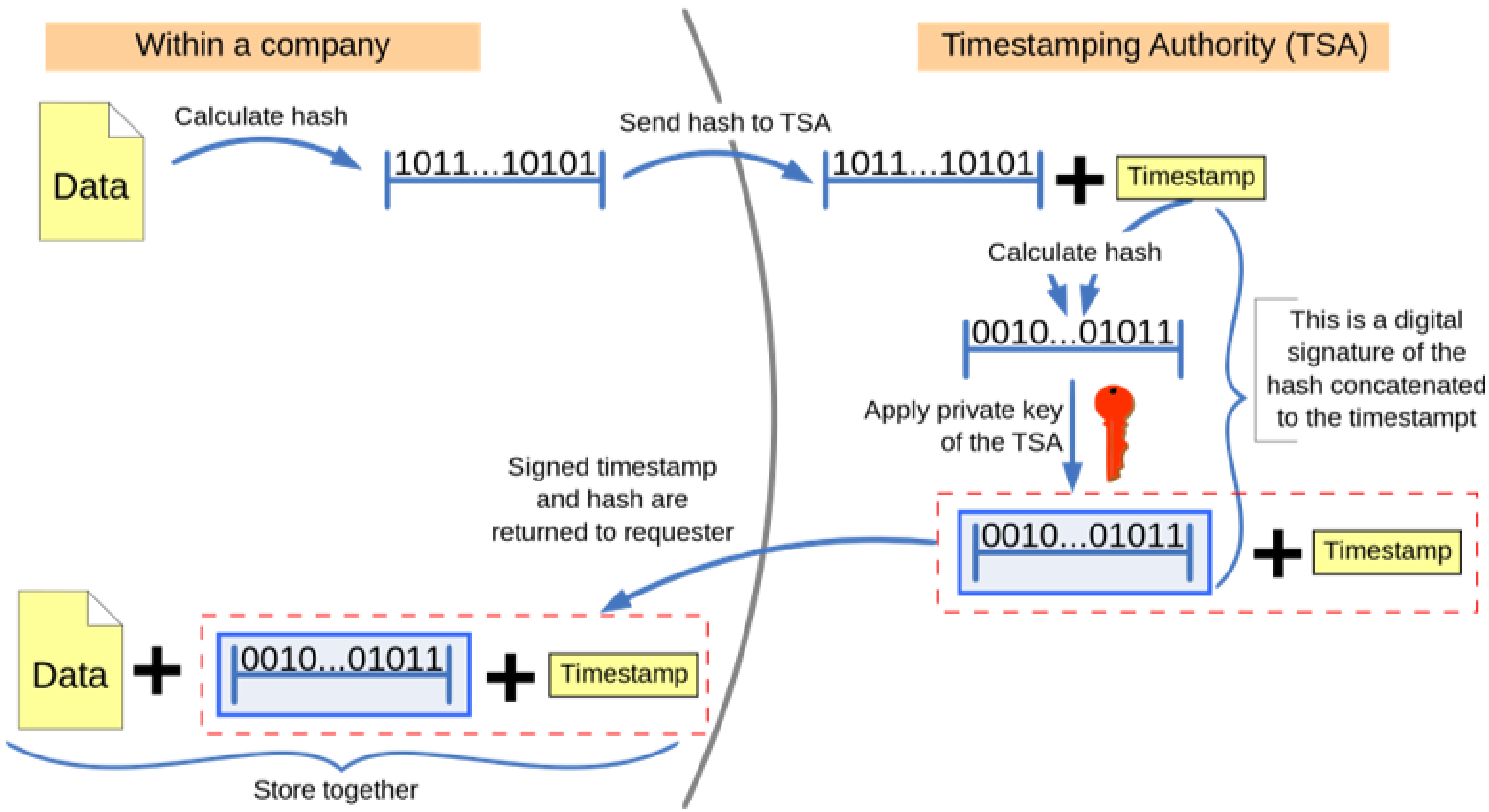


Использование меток времени

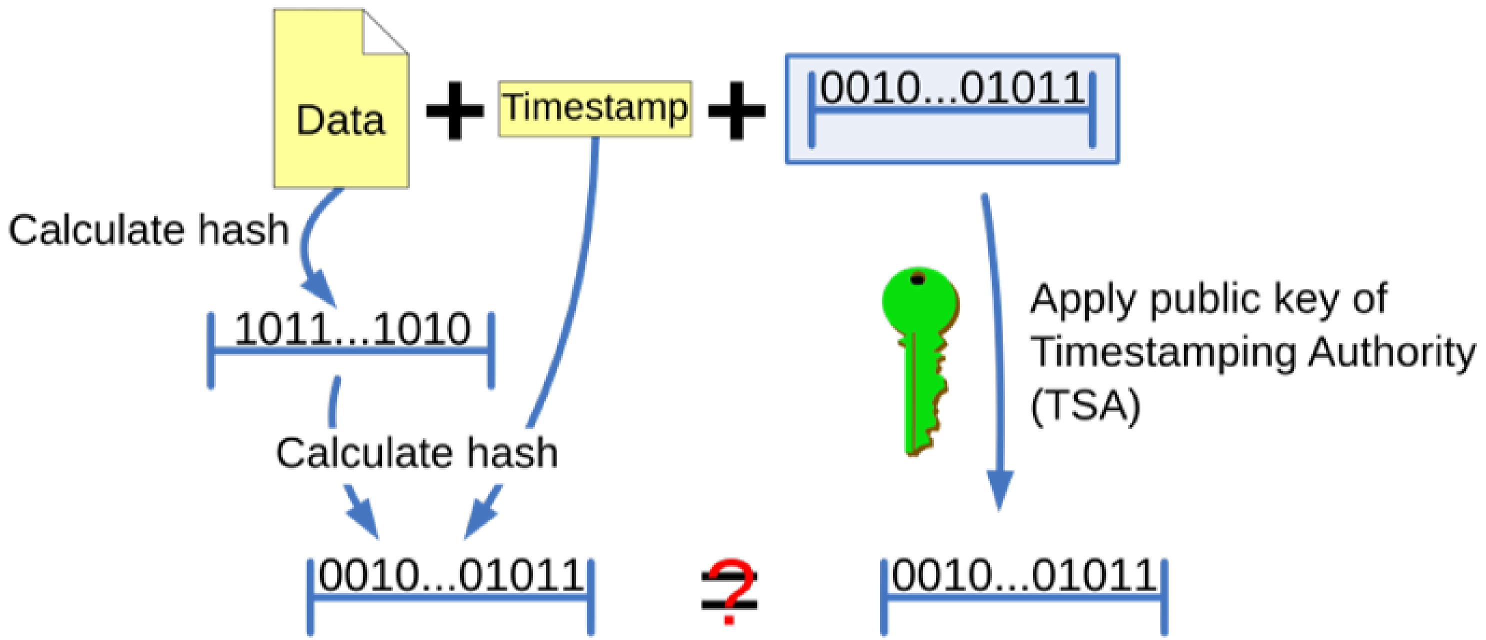
Временна́я метка (также **метка времени** или **timestamp** с англ. — «временна́я печать») — это последовательность символов или закодированной информации, показывающей, когда произошло определённое событие. Обычно показывает дату и время (иногда с точностью до долей секунд)

Защищённая метка времени — это метка, выданная при свидетелях. **Trusted third party** (**TTP**) ведёт себя как timestamping authority (**TSA**). Это используется для подтверждения существования определённых данных до определённого момента времени (контракты, данные исследования, медицинские записи и т. п.) без возможности дописывания задним числом. Сложные TSA могут использоваться для повышения надёжности и уменьшения уязвимости

Создание метки времени



Проверка метки времени



Лягушка с открытым ртом

Протокол "**Лягушка с открытым ртом**" (**Wide-Mouth Frog** англ.) — простейший протокол управления симметричными ключами. Он позволяет двум абонентам установить общий сессионный ключ для защищенного общения между собой

Описание протокола:

1. **Алиса** генерирует сеансовый ключ **K**
2. **Алиса** шифрует конкатенацию метки времени ($\mathbf{T_A}$), идентификатора Боба (**B**) и ключа **K** с помощью некоторого общего ключа, известного **Алисе** и **Тренту**
3. **Алиса** передает свой идентификатор и зашифрованное значение **Тренту**: { $\mathbf{A, E_A (T_A, B, K)}$}
4. **Трент** расшифровывает совместным с **Алисой** ключом пакет, выбирает оттуда сгенерированный **Алисой** случайный сеансовый ключ **K** и составляет конкатенацию из новой метки времени, идентификатора Алисы и сеансового ключа, после чего шифрует её общим с **Бобом** ключом и передаёт ему: { $\mathbf{E_B (T_T, A, K)}$}
5. После этого **Боб** расшифровывает пакет данных общим с **Трентом** ключом и может использовать сгенерированный **Алисой** случайный сеансовый ключ для передачи данных

Лягушка с открытым ртом

Недостатки алгоритма:

- **Трент** имеет доступ ко всем ключам
- Значение сеансового ключа **K** полностью определяется **Алисой**, то есть она должна быть достаточно компетентной для генерации хороших ключей.
- Может **дублировать** сообщения, во время действия временной метки
- **Алиса** не знает существует ли **Боб**
- Все данные проходят через **Трента**, который в данном случае выполняет активную роль

Примечание:

- В 1995 и в 1997 годах были описаны алгоритмы атаки на протокол с помощью подмены ключей
- В 1997 году **Гэвин Лоу** предложил модифицированный алгоритм

Протокол Нидхема-Шрёдера: SK

Протокол The Needham-Schroeder shared-key был предложен в 1978 году и лежит в основе протоколов **Kerberos** и **Otway-Rees**

Описание протокола:

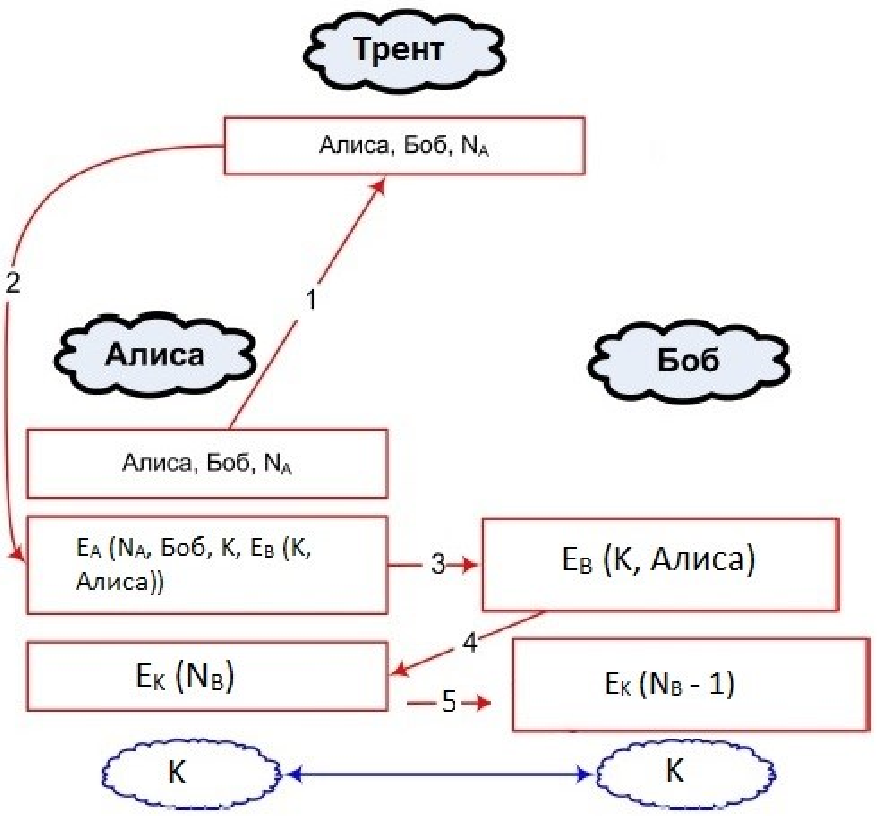
1. **Алиса** выбирает число $\mathbf{N_A}$, **Боб** выбирает число $\mathbf{N_B}$
2. **Алиса** формирует сообщение $\mathbf{M_0} = \{ \mathbf{A, B, N_A} \}$ и отправляет **Тренту**
3. **Трент** формирует сообщение, состоящее из 2-х частей:
   - **1 часть:** $\mathbf{N_A}$, идентификатор **Боба - B** и новый ключ **K**
   - **2 часть:** шифр от ключа **K** и идентификатора **Алисы – A**

   $$\mathbf{M_1 = E_A (N_A, B, K, E_B (K, A))}$$
4. **Алиса** расшифровывает сообщение и найдя в нем $\mathbf{N_A}$, убеждается, что поговорила с **Трентом**. Вторую часть она прочитать не может и просто переправляет ее **Бобу**: $\mathbf{M_2 = E_B (K, A)}$
5. **Боб** расшифровывает сообщение, извлекает из него ключ **K**, помещает в него свое число $\mathbf{N_B}$, зашифрованное ключом **K**:

   $$\mathbf{M_3 = E_K (N_B)}$$
6. **Алиса** получает сообщение, извлекает из него $\mathbf{N_B}$, уменьшает его на 1, шифрует ключом **K** и отправляет обратно Бобу: $\mathbf{M_4 = E_K (N_B} -1)$
7. **Боб** проверяет значение **K**-1, проверяя тем самым, что это **Алиса**
8. **Алиса** и **Боб** владеют общим ключом **K** для шифрования

Протокол Нидхема-Шрёдера: SK

Протокол на **симметричных ключах** использует удостоверяющий центр для того чтобы удостовериться в подлинности абонентов



Протокол Kerberos

Протокол был предложен в 1980 году, в 1989 вышла 5-а версия

Описание протокола:

1. **Алиса** выбирает число $\mathbf{N_A}$
2. **Алиса** формирует сообщение $\mathbf{M_0}$ = { **A, B,** $\mathbf{N_A}$ } и отправляет **Тренту**
3. **Трент** формирует сообщение, состоящее из 2-х частей:
   - **1 часть:** $\mathbf{N_A}$, **B,** ключ **K,** период валидности **t**
   - **2 часть:** шифр от ключа **K, A** и **t**

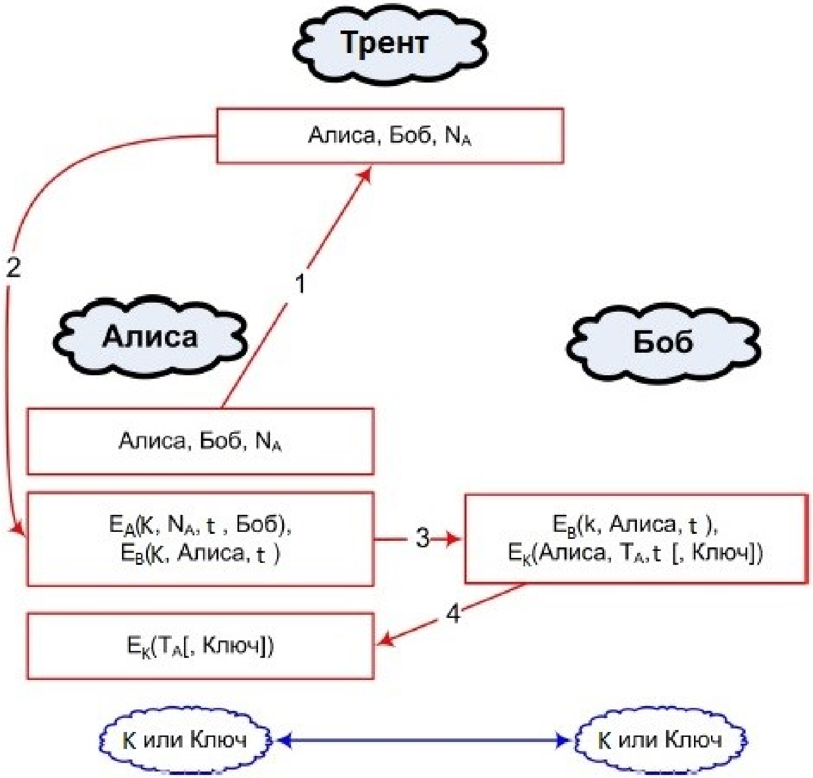
$$M_1 = E_A(N_A, B, K, t), E_B(K, A, t))$$

4. **Алиса** расшифровывает сообщение и отправляет **Бобу** сообщение из 2-х частей: $M_2 = E_B(K, A, t), E_K(A, T_A, t)$
5. **Боб** принимает сообщение, извлекает из него ключ **K** и используя его расшифровывает вторую часть, после чего отправляет ей метку времени, зашифрованную ключом **K**:

$$M_3 = E_K(T_A)$$

6. **Алиса** удостоверяется, что **Боб** — это **Боб**. Здесь применимы следующие рассуждения: **Боб** мог расшифровать сообщение от **Алисы** с меткой времени, только если он знал ключ **K**. А ключ **K** он мог узнать, только если знает $\mathbf{E_B}$. А так как это секретный ключ **Боба** и **Трента**, то приславший сообщение **Алисе** — **Боб**
7. **Алиса** и **Боб** используют ключ **K**

Протокол Kerberos



Протокол Нидхема-Шрёдера: PK

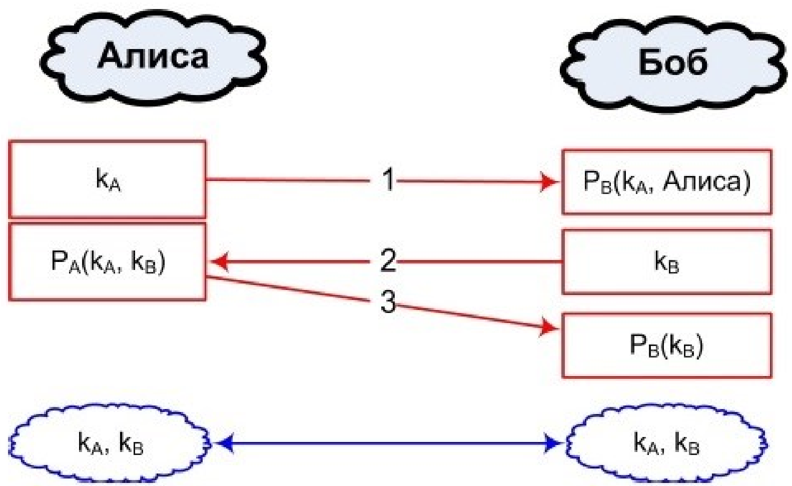
Протокол The Needham-Schroeder public-key был предложен в 1978 году

Описание протокола:

1. **Алиса** выбирает свою часть ключа, $\mathbf{k_A}$, и формирует сообщение **Бобу**, в которое кладет свой идентификатор **A** и $\mathbf{k_A}$. Все сообщение шифруется публичным ключом Боба $\mathbf{P_B}$ и отправляется ему же:
$$\mathbf{M_0 = P_B (A, k_A)}$$
2. **Боб** расшифровывает сообщение и теперь знает, что с ним хочет поговорить **Алиса**, и для общения она хочет использовать ключ $\mathbf{k_A}$. **Боб** выбирает свою часть ключа, $\mathbf{k_B}$, и отправляет **Алисе** сообщение, состоящее из двух ключей $\mathbf{k_A}$ и $\mathbf{k_B}$, зашифрованное открытым ключом **Алисы**. Тем самым **Боб** подтверждает **Алисе**, что получил часть её ключа $\mathbf{K_A}$: $\mathbf{M_1 = P_A (k_A, k_B)}$
3. Теперь очередь **Алисы** доказать **Бобу**, что она — **Алиса**. Чтобы это сделать, она должна уметь расшифровывать сообщения, зашифрованные ключом $\mathbf{P_A}$. С чем она прекрасно справляется — она расшифровывает сообщение от **Боба**, забирает оттуда $\mathbf{k_A}$ и отправляет **Бобу** сообщение, содержащее его ключ $\mathbf{k_B}$: $\mathbf{M_2 = P_B (k_B)}$
4. В результате на этапе сообщения $\mathbf{M_1}$ **Алиса** уверена, что **Боб** — это **Боб**, и **Боб** знает весь ключ. А на этапе сообщения $\mathbf{M_2}$ **Боб** уверен, что разговаривал с **Алисой**, и она знает весь ключ

Протокол Нидхема-Шрёдера: PK

Протокол на **асимметричных ключах не** использует удостоверяющий центр для того чтобы удостовериться в подлинности абонентов



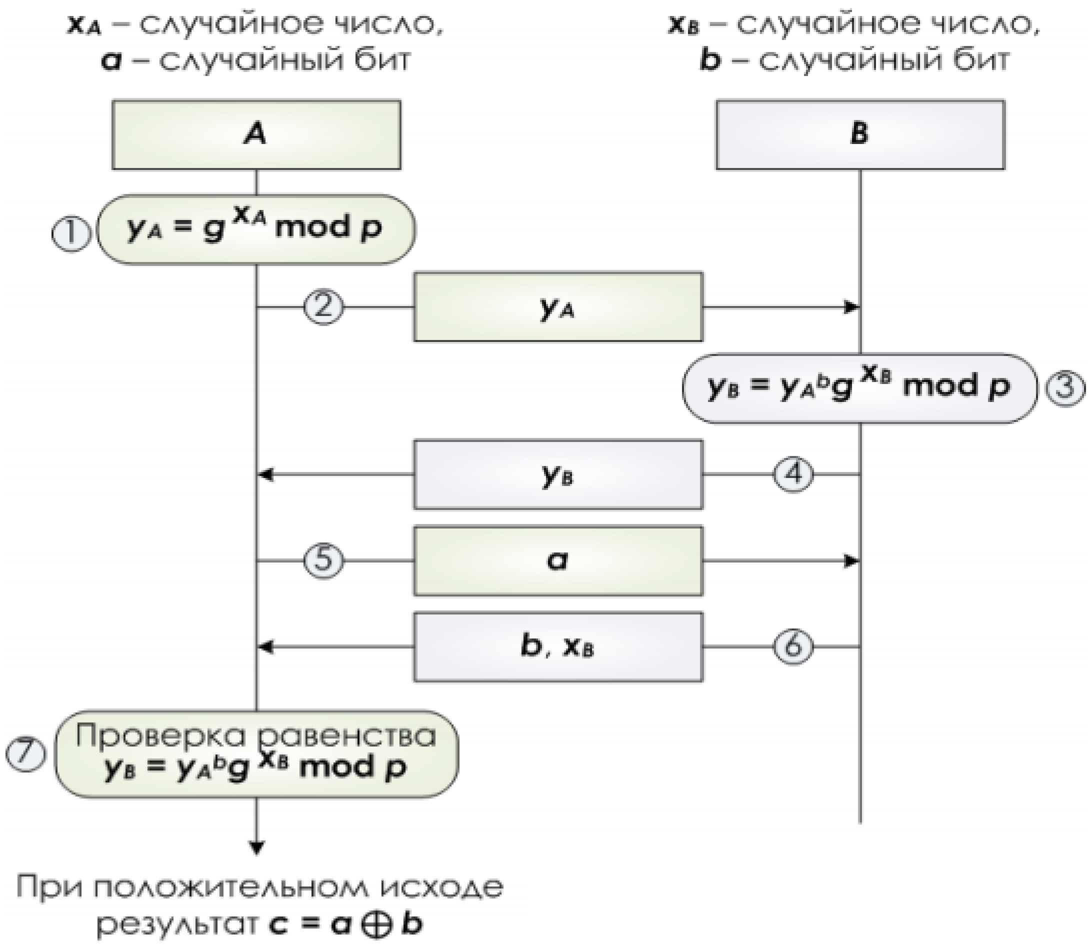
Протокол «Подбрасывание монеты»

- **Алиса** и **Боб** хотят провести жеребьевку
- К примеру, **подбросить монету**, но при этом они находятся друг от друга удалённо, в разных городах. В данной ситуации существует вероятность, что тот, кто подбрасывает монету, после броска монеты, может солгать другому, а другой может ему не поверить. Поэтому появилась нужда в алгоритме, выдающий независимый случайный результат.
- В 1981 году **Мануэль Блюм** опубликовал статью о протоколе «**подбрасывания монеты по телефону**» (CoinFlippingByTelephone), причём в заголовке своей работы он назвал это методом решения «нерешаемых задач». Для решения проблемы было использовано добавления в процесс третьего лица, на которое **Алиса** и **Боб** возлагали доверие. Протокол позволял сторонам генерировать случайное число, состоящее из m бит и состоял он из 7 этапов:

Протокол «Подбрасывание монеты»

1. **Алиса** выбирает случайное большое целое число $\mathbf{X_A}$, и вычисляет значение
   $$\mathbf{Y_A} = \mathbf{g}^{\mathbf{X_A}} \bmod \mathbf{p},$$
   где **p** – простое число, а $\mathbf{g}^{\mathbf{p}} = 1 \bmod \mathbf{p}$
2. **Алиса** отправляет полученное значение $\mathbf{Y_A}$ **Бобу**
3. **Боб** генерирует случайный бит **b**, случайное большое целое число $\mathbf{X_B}$ и вычисляет значение:
   $$\mathbf{Y_B} = \mathbf{Y_A}^{\mathbf{B}} * \mathbf{g}^{\mathbf{X_B}} \bmod \mathbf{p}$$
4. **Боб** отправляет полученное значение $\mathbf{Y_B}$ **Алисе**
5. **Алиса** генерирует случайный бит **a** и отправляет его **Бобу**
6. **Боб** посылает Алисе **b** и $\mathbf{X_B}$
7. **Алиса** проверяет, выполняется ли сравнение:
   $$\mathbf{Y_B} = \mathbf{Y_A}^{\mathbf{B}} * \mathbf{g}^{\mathbf{X_B}} \bmod \mathbf{p}$$
8. Если условие выполняется то результатом жеребьевки является значение $\mathbf{c} = \mathbf{a} \oplus \mathbf{b}$

Протокол «Подбрасывание монеты»



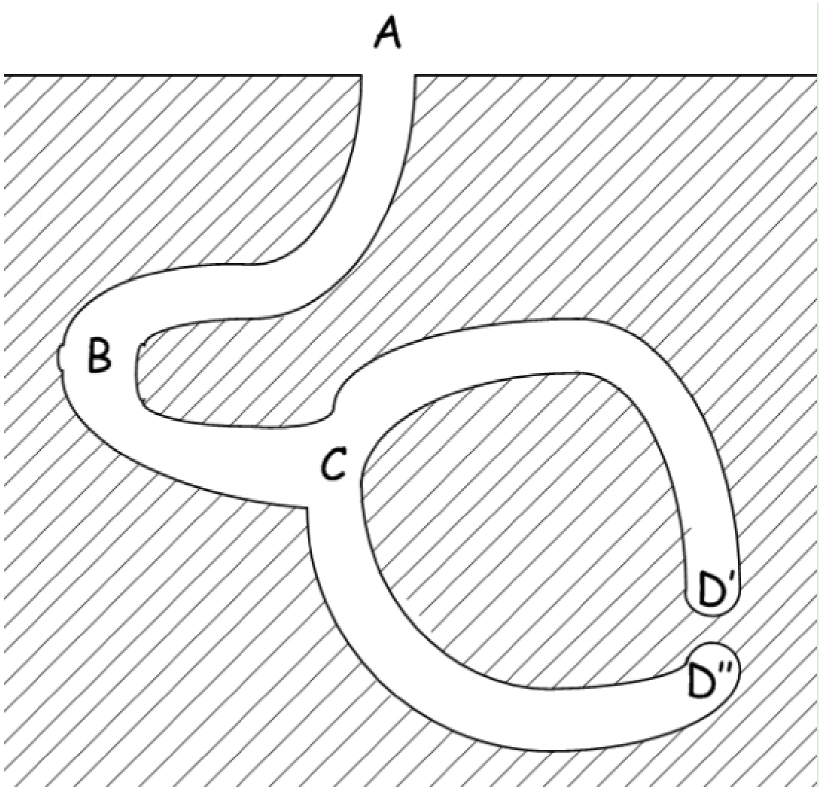
Доказательство с нулевым разглашением

- **Доказа́тельство с нулевы́м разглаше́нием** (информа́ции) в криптографии (англ. Zero-knowledge proof) — интерактивный криптографический протокол, позволяющий одной из взаимодействующих сторон («**The verifier**» — проверяющей) убедиться в достоверности какого-либо утверждения (обычно математического), не имея при этом никакой другой информации от второй стороны («**The prover**» — доказывающей)
- Причём последнее условие является **необходимым**, так как обычно доказать, что сторона обладает определёнными сведениями в большинстве случаев **тривиально**, если она имеет право просто раскрыть информацию
- Вся **сложность** состоит в том, чтобы доказать, что у одной из сторон есть информация, **не раскрывая** её содержание
- Протокол должен учитывать, что доказывающий сможет убедить проверяющего только в случае, если утверждение действительно доказано. В противном случае сделать это будет невозможно, или крайне маловероятно из-за вычислительной сложности.

Пещера нулевого знания Б. Шнайера

- **Пегги** знает магическое слово («ключ»), ввод которого позволяет открыть ей дверь между **C** и **D**
- **Виктор** хочет узнать, действительно ли **Пегги** знает пароль, при этом **Пегги** не хочет выдавать сам пароль
- **Виктор** идёт к разветвлению, то есть в точку **B**, и кричит оттуда: «**Пегги** нужно выйти справа» или «**Пегги** нужно выйти слева». Получаем каждый раз вероятность того, что **Пегги** не знает пароль, равна 50 %
- Если же повторить процесс **k** раз, то вероятность будет равна $1/2^k$
- При 20 повторениях эта вероятность будет порядка 10^{-6}, что является достаточным

Пегги – доказывающий
Виктор - проверяющий



Протокол Фиата-Шамира

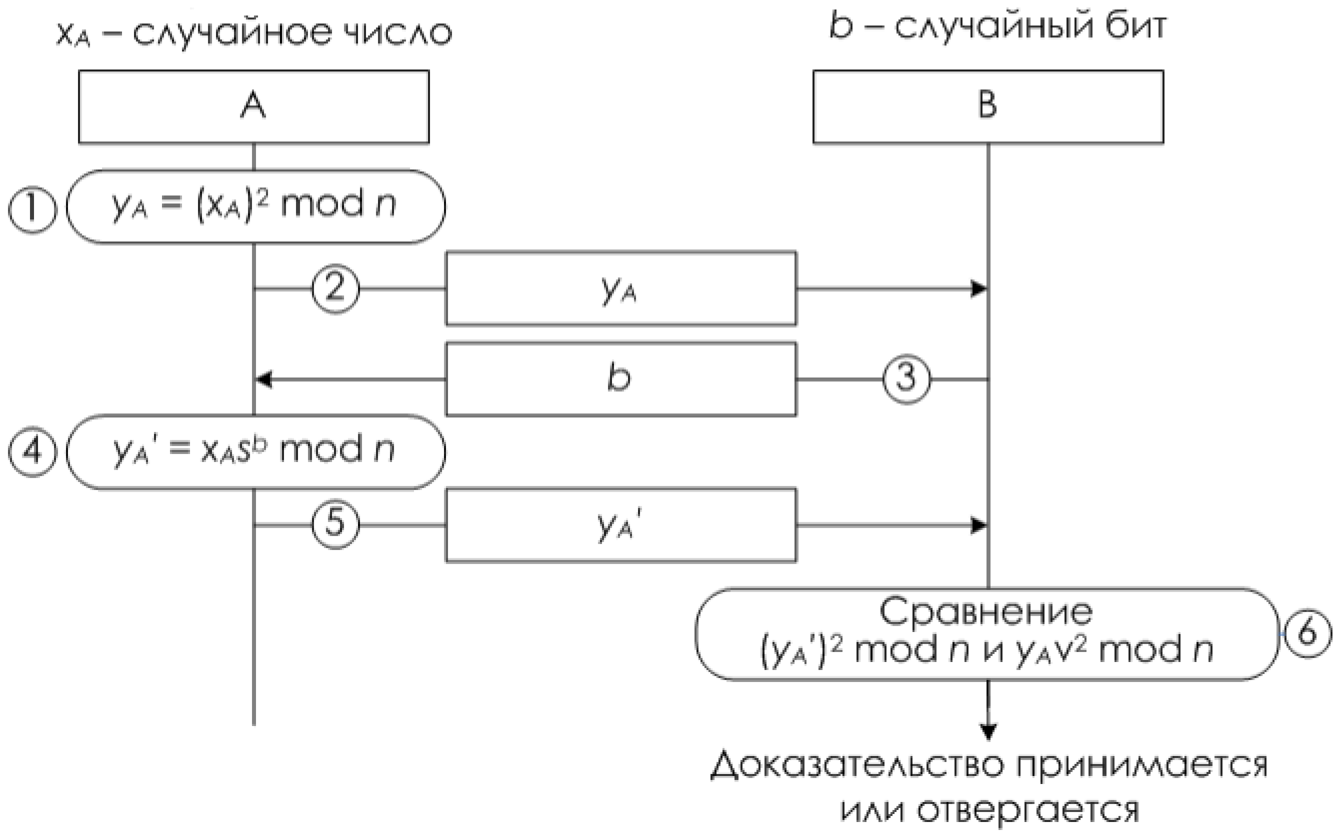
- Протокол **Фиата — Шамира** — это один из наиболее известных протоколов идентификации с нулевым разглашением (Zero-knowledge protocol). Протокол был предложен **Амосом Фиатом** (англ. Amos Fiat) и **Ади Шамиром** (англ. Adi Shamir) в 1983 году

- Пусть **A** знает некоторый секрет **s**. Необходимо доказать знание этого секрета некоторой стороне **B** без разглашения какой-либо секретной информации
- **A** доказывает **B** знание **s** в течение **t** раундов. Раунд называют также **аккредитацией**. Каждая аккредитация состоит из 3-х этапов

Предварительные действия:

- Доверенный центр **T** выбирает и публикует модуль **n**=**p*****q**, где **p**, **q** - простые числа и держатся в секрете
- Претендент **A** выбирает **s** взаимно-простое с **n**, где **s** принадлежит [1,**n**-1]. Затем вычисляется $\mathbf{V} = \mathbf{S}^2 \bmod \mathbf{n}$
- **V** регистрируется **T** в качестве открытого ключа **A**

Протокол Фиата-Шамира



Протокол Фейга-Фиата-Шамира

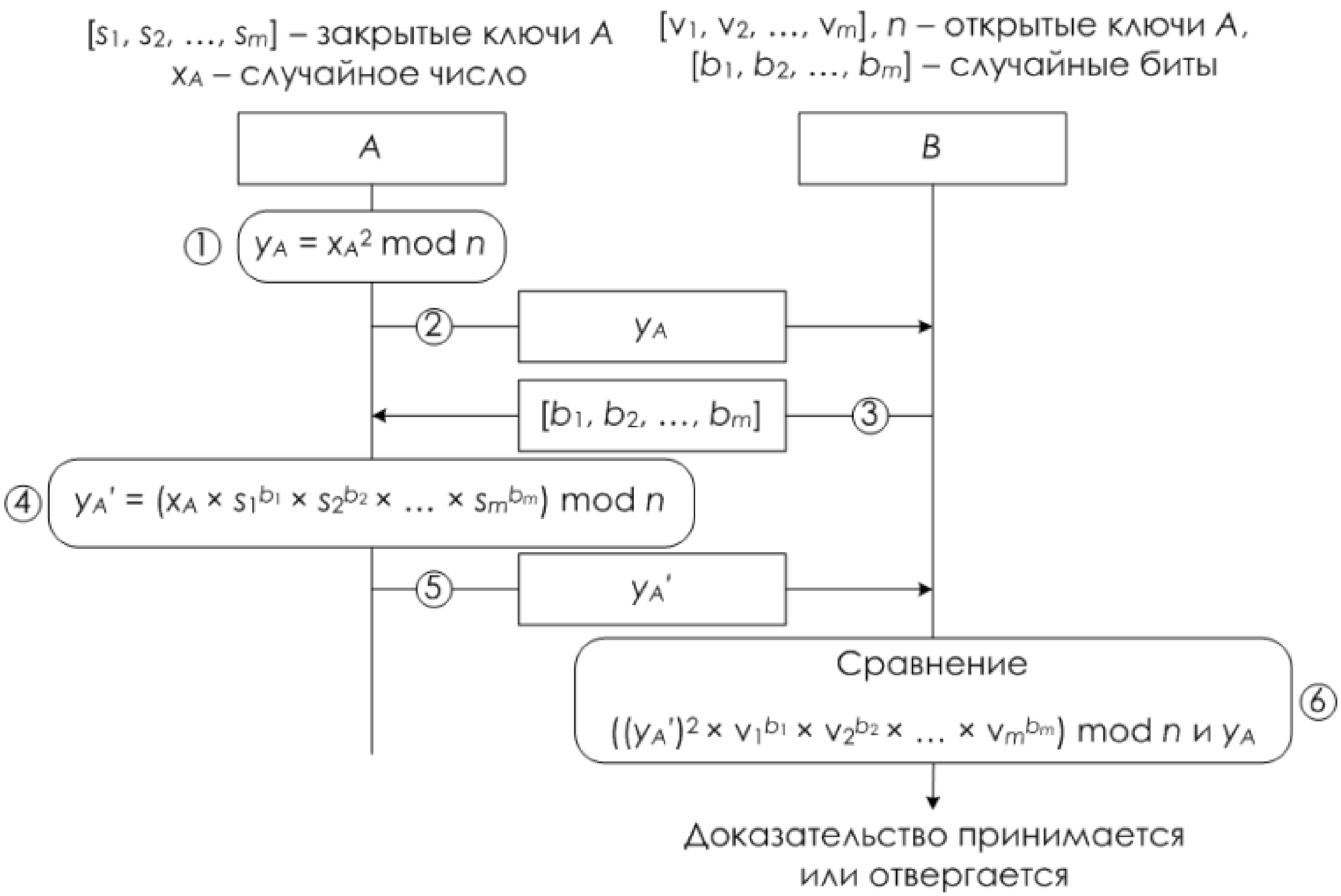
- Протокол **Фиата — Шамира** — это один из наиболее известных протоколов идентификации с нулевым разглашением (Zero-knowledge protocol). Протокол был предложен **Амосом Фиатом** (англ. Amos Fiat) и **Ади Шамиром** (англ. Adi Shamir) в 1983 году

- Пусть **A** знает некоторый секрет **s**. Необходимо доказать знание этого секрета некоторой стороне **B** без разглашения какой-либо секретной информации
- **A** доказывает **B** знание **s** в течение **t** раундов. Раунд называют также **аккредитацией**. Каждая аккредитация состоит из 3-х этапов

Предварительные действия:

- Доверенный центр **T** выбирает и публикует модуль **n**=**p*****q**, где **p**, **q** - простые числа и держатся в секрете
- Претендент **A** выбирает **s** взаимно-простое с **n**, где **s** принадлежит [1,**n**-1]. Затем вычисляется $\mathbf{V = S^2} \bmod \mathbf{n}$
- **V** регистрируется **T** в качестве открытого ключа **A**

Протокол Фейга-Фиата-Шамира



Прикладной протокол «Защита БД»

Задача:
Существует открытая база данных адресов сотрудников, любой сотрудник может получить адрес коллеги, по его фамилии, однако получить список адресов всех сотрудников, например с целью рассылки спама, должно быть невозможно

Прикладной протокол «Защита БД»

Решение:

1. Выбирается однонаправленная **хэш-функция** и **симметричный алгоритм** шифрования
2. У каждой записи в базе данных два поля. Индексным полем является фамилия сотрудника, обработанная хэш-функцией (md5)
3. Поле данных адреса шифруется с помощью используемой в качестве ключа фамилии. Не зная фамилии, невозможно расшифровать поле данных
4. Для поиска по фамилии, она сначала хэшируется, и выполняется поиск значения хэш-функции в базе данных, затем расшифровывается поле адреса

Прикладной протокол «Защита БД»

Пример таблицы

Hash (Фамилия)	E_k(Адрес, Фамилия)
cd1a2693791e24bb09b002275442d9e2	U2FsdGVkX1+2H/JsG0LG0eA3oNWzcz0/re9NhJgdm0uZI37WUPnH3RrisVzjZKdq
…	…

Md5(“**Иванов**”) = cd1a2693791e24bb09b002275442d9e2
AES (“**Ленина, 15**”, “**Иванов**”) =
U2FsdGVkX1+2H/JsG0LG0eA3oNWzcz0/re9NhJgdm0uZI37WUPnH3RrisVzjZKdq

При поступлении запроса на поиск по фамилии «Иванов», сначала эту фамилию необходимо хешировать, чтобы определить строку, а затем взять поле адреса и расшифровать его с помощью функции:
D_k(“U2FsdGVkX1+2H/JsG0LG0eA3oNWzcz0/re9NhJgdm0uZI37WUPnH3RrisVzjZKdq”,”Иванов”)

Известные прикладные протоколы

- **TLS** (англ. Transport Layer Security) — криптографический протокол, обеспечивающие защищённую передачу данных между узлами в сети Интернет. TLS использует асимметричную криптографию для обмена ключами, симметричное шифрование для конфиденциальности и коды аутентичности сообщений для сохранения целостности сообщений. Данный протокол широко используется в приложениях, работающих с сетью Интернет, таких как Веб-браузеры, работа с электронной почтой, обмен мгновенными сообщениями и IP-телефония (VoIP)
- **IPsec** (сокращение от IP Security) — набор протоколов для обеспечения защиты данных, передаваемых по межсетевому протоколу IP. Позволяет осуществлять подтверждение подлинности (аутентификацию), проверку целостности и/или шифрование IP-пакетов. IPsec также включает в себя протоколы для защищённого обмена ключами в сети Интернет. В основном, применяется для организации VPN-соединений
- **SSH** (англ. Secure Shell — «безопасная оболочка») — сетевой протокол прикладного уровня, позволяющий производить удалённое управление операционной системой и туннелирование TCP-соединений (например, для передачи файлов).