Ассиметричные системы. Алгоритм RSA. Система Эль-Гамаля

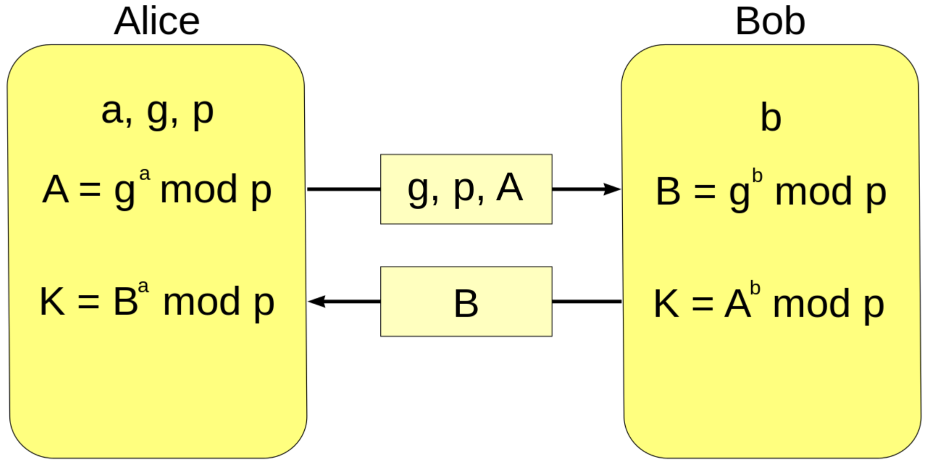
Лекция №6

Архетипы в криптографии

- Для упрощения описания участников криптографической системы часто используют условные обозначения - **архетипы**
- Чаще всего в криптографии используются следующие обозначения участников обмена информацией:
  - **Алиса (Alice)** – первый участник криптографической системы
  - **Боб (Bob)** – второй участник криптографической системы
  - **Кэрол, Карлос или Чарли (Carol, Carlos or Charlie)** — выступают в качестве третьего участника соединения
  - **Ева (Eve)** - пассивный злоумышленник, от англ. eavesdropper (подслушивающий). Она может прослушивать сообщения между Алисой и Бобом, но она не может влиять на них
  - **Мэллори (Mallory)** - активный злоумышленник, от англ. Malicious (злонамеренный). В отличие от Евы, Мэллори может изменять сообщения, воспроизводить старые сообщения, подменять сообщения и так далее
  - **Трент (Trent)**, доверенный арбитр — своего рода нейтральная третья сторона, чаще всего это посредник, который заслуживает доверия. Например сертификационный центр

Алгоритм Диффи-Хеллмана

- В 1976 году **Уитфилдом Диффи** (Whitfield Diffie) и **Мартином Хеллманом** (Martin Hellman) был предложен алгоритм, который позволяет двум сторонам получить общий секретный ключ, используя незащищенный от прослушивания, но защищённый от подмены канал связи
- Полученный ключ можно использовать для обмена сообщениями с помощью **симметричного шифрования**



$$K = A^b \bmod p = (g^a \bmod p)^b \bmod p = g^{ab} \bmod p = (g^b \bmod p)^a \bmod p = B^a \bmod p$$

Алгоритм Диффи-Хеллмана

- Пусть обоим абонентам - Алисе и Бобу известны некоторые два числа **g** и **p**, которые не являются секретными и могут быть известны всем остальным лицам
- Обычно значения **p** и **g** генерируются на одной стороне и передаются другой, при этом:
  - **p** является случайным простым числом
  - **g** является первообразным корнем по модулю **p**, то есть для него выполняется условие $\mathbf{g^{(p)} = 1 \ (mod\ p)}$
- Алиса генерирует случайное натуральное число **a** и вычисляет значение $\mathbf{A = g^a \ mod\ p}$
- Боб генерирует случайное натуральное число **b** и вычисляет значение $\mathbf{B = g^b \ mod\ p}$
- Алиса и Боб обмениваются значениями **A** и **B**
- Алиса вычисляет $\mathbf{B^a \ mod\ p}$, а Боб вычисляет $\mathbf{A^b \ mod\ p}$
- У обоих участников получается одно и то же число:

$$\mathbf{K = g^{ab} \ mod\ p}$$

- Это число **K** является паролем для симметричной системы

Алгоритм Диффи-Хеллмана

Пример вычисления **K**:

- **$p = 23$** – число является простым
- **$g = 5$**, так как $g^{(p)} = 1$ (mod p), где (p) – функция Эйлера
- **$a = 6$** – секретное число Алисы
- **$b = 18$** – секретное число Боба
- **$A = 5^6 \bmod 23 = 8$** – открытое число Алисы (отправляет Бобу)
- **$B = 5^{18} \bmod 23 = 6$** – открытое число Боба (отправляет Алисе)
- **$K_A = 6^6 \bmod 23 = 12$** – вычисленный ключ Алисы
- **$K_B = 8^{18} \bmod 23 = 12$** – вычисленный ключ Боба

Проверим вычисления

- **$K = K_A = K_B = 5^{18*6} \bmod 23 = 12$**

Алгоритм Диффи-Хеллмана

Пусть Ева перехватила A и B (канал открыт для чтения)

- Для расшифровки ей нужно решить уравнение:

$$\mathbf{5^x = 8 \pmod{23}}$$

- **Метод перебора** позволяет решить эту задачу, последовательно вычисляя: 5^1, 5^2, 5^3, 5^4,…, 5^{22} в конечном поле 23
- Поэтому в **практических реализациях** рекомендуется выполнять следующие требования:
  - для **a** и **b** используются числа порядка 10^{100}
  - для **p** используются числа порядка 10^{300}
  - число **g** не обязано быть большим и обычно имеет значение в пределах первого десятка
- Также для решения этой задачи в Криптоанализе используют: Алгоритм Шенкса, Алгоритм Полига-Хеллмана, Метод Полларда, Алгоритм Адлемана, Алгоритм COS подробнее: https://ru.wikipedia.org/wiki/Дискретное_логарифмирование

Алгоритм Диффи-Хеллмана

Данный алгоритм можно использовать для 3-х и более участников:

- Стороны договариваются о параметрах алгоритма **p** и **g**
- Стороны, Алиса, Боб и Кэрол генерируют свои ключи — **a**, **b** и **c** соответственно.
- Алиса вычисляет $\mathbf{g^a}$ и посылает его Бобу
- Боб вычисляет $\mathbf{(g^a)^b = g^{ab}}$ и посылает его Кэрол
- Кэрол вычисляет $\mathbf{(g^{ab})^c = g^{abc}}$ и получает тем самым общий секретный ключ
- Боб вычисляет $\mathbf{g^b}$ и посылает его Кэрол
- Кэрол вычисляет $\mathbf{(g^b)^c = g^{bc}}$ и посылает его Алисе
- Алиса вычисляет $\mathbf{(g^{bc})^a = g^{bca} = g^{abc}}$ — общий секретный ключ
- Кэрол вычисляет $\mathbf{g^c}$ и посылает его Алисе
- Алиса вычисляет $\mathbf{(g^c)^a = g^{ca}}$ и посылает его Бобу
- Боб вычисляет $\mathbf{(g^{ca})^b = g^{cab} = g^{abc}}$ и также получает общий секретный ключ

Распределение ключей

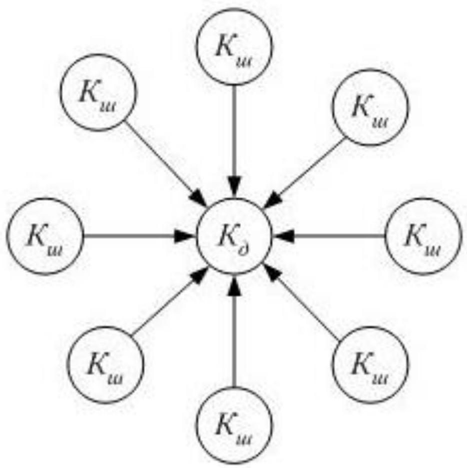
- В **симметричной криптографии** каждая из переписывающихся сторон должна иметь копию общего секретного ключа, что создает огромную проблему для такого варианта взаимодействия
- В криптосистемах с открытым ключом используются два ключа – **открытый ключ** и **закрытый ключ**
- **Открытый ключ** может быть известен для всех. В его помощью любой желающий может взять свое сообщение, зашифровать его открытым ключом получателя и отправить его получателю
- Расшифровать такое сообщение может только то, у кого есть **закрытый ключ**

сообщение + открытый ключ алисы = шифротекст
шифротекст + секретный ключ Алисы = сообщение.

- Такие системы в криптографии принято называть **ассиметричными системами** или **системами с открытым ключом (СОК)**

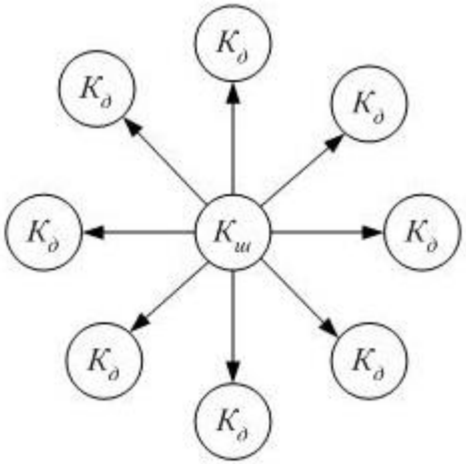
Распределение ключей (схема 1)

- Получатель сообщения производит вычисление пары ключей $К_ш$ и $К_д$. Ключ $К_д$ считается закрытым, $К_ш$ – открытым.
- Закрытый ключ $К_д$ получатель оставляет у себя, а открытый ключ $К_ш$ пересылает отправителю.
- Пользуясь открытым ключом $К_ш$ любой абонент может зашифровать текст и отослать его получателю. Расшифровать зашифрованное сообщение может только тот, у кого есть закрытый ключ, а так как он никуда не передается, то теоретически такой ключ может быть только у получателя.
- Пользуясь закрытым ключом $К_д$, получатель расшифровывает полученное сообщение.



Распределение ключей (схема 2)

- Отправитель сообщения производит вычисление пары ключей $К_ш$ и $К_д$. Ключ $К_ш$ считается закрытым, $К_д$ – открытым.
- Закрытый ключ $К_ш$ отправитель оставляет у себя, а открытый ключ $К_д$ пересылает получателю.
- Пользуясь закрытым ключом $К_ш$, отправитель производит шифрование сообщения и отправляет зашифрованный текст получателю.
- Пользуясь открытым ключом $К_д$, любой абонент может расшифровать зашифрованный текст. Причем ключ дешифрования $К_д$ может расшифровать только то сообщение, которое было зашифровано закрытым ключом $К_ш$.



Основные принципы СОК

1. Существует **односторонняя** математическая связь между открытым и закрытым ключом
2. Информация об открытом ключе никак **не помогает восстановить** закрытый ключ
3. Владение секретным ключом обеспечивает возможность расшифровать **только те сообщения, которые были зашифрованы открытым ключом** из данной пары ключей

Односторонние функции

- Для обеспечения принципов СОК необходимы **односторонние функции**
- **Односторонней** называется функция $F: X \to Y$ обладающая двумя свойствами:
  1. **Существует полиномиальный** алгоритм вычисления значений $F(x)$
  2. **Не существует полиномиального** алгоритма инвертирования функции F, то есть решения уравнения $F(x) = y$ относительно x
- Примеры односторонних функций:
  - Задача разложения на множители **произведения двух больших простых чисел**
  - Проблема вычисления **логарифма в конечном поле**
  - Вычисление **корней алгебраических уравнений**
  - **Эллиптические кривые**

Односторонние функции

- Замечание по односторонним функциям, которые используются в современной криптографии
- **Вышеперечисленные функции являются односторонними только в вычислительном отношении, то есть имея достаточно большие компьютерные мощности, их вполне обратить!!!**
- Поэтому на практике достаточно чтобы выполнялось условие, что прямое вычисление во много раз проще обратного вычисления, а для этого приходится работать с очень большими числами, и это требование с развитием современной техники ужесточается с каждым днем:
  - 2010 год – 1024 бита
  - 2020 год – 2048 бит
  - 2030 год – 3072 и 7680 бит

Функции с ловушкой

- **Односторонней функцией с секретом (с ловушкой)** называется функция $F_K: X \to Y$ обладающая тремя свойствами:

1. **Существует полиномиальный** алгоритм вычисления значений $F_K(x)$ для любых K и X
2. **Не существует полиномиального** алгоритма инвертирования функции F_K при **неизвестном K**
3. **Существует полиномиальный** алгоритм инвертирования функции F_K при **известном K**

Алгоритм RSA

Спустя почти год, после публикации алгоритма Диффи-Хеллмана **в 1977** году Ривест Рональд Линн (**Rivest**), Ади Шамир (**Shamir**) и Леонард Макс Адлеман (**Adleman**) опубликовали первый криптографический алгоритм **RSA** с открытым ключом, основывающийся на вычислительной сложности задачи факторизации больших целых чисел



Алгоритм RSA

- Алиса хочет получать зашифрованные сообщения, поэтому она выбирает функцию-ловушку $\mathbf{F_K}$ с секретом **K**
- Алиса сообщает всем заинтересованным участникам описание функции $\mathbf{F_K}$ (открытый ключ) в качестве своего алгоритма шифрования, но при этом значение секрета **K** (закрытый ключ) она никому не сообщает и **держит в секрете**
- Если Боб хочет послать Алисе сообщение **m** в зашифрованном виде, то он вычисляет $\mathbf{c = F_K(m)}$ и посылает **c** по открытому каналу Алисе
- Алиса для своего секрета **K** умеет инвертировать $\mathbf{F_K(m)}$, поэтому она может вычислить **m** по полученному **c**. Никто другой не знает **K** и не может вычислить **m** за приемлемый промежуток времени

Алгоритм RSA

Генерация ключей

- Возьмем два больших простых числа **p** и **q**
- Определим **n**, как результат умножения **p** и **q** (n= p*q).
- Определим **m**, как результат умножения **p-1** и **q-1** (m= (p-1)*(q-1))
- Выберем случайное число, которое назовем **d**. Это число должно быть взаимно простым (не иметь ни одного общего делителя, кроме 1) с **m**
- Определим такое число **e**, для которого является истинным следующее соотношение

$$(e*d) \bmod ((p-1)*(q-1))=1$$

- Открытым ключом назовем пару чисел {**e**, **n**}, а секретным ключом пару чисел {**d**, **n**}

Алгоритм RSA

Шифрование

- Исходный текст разбивается на блоки фиксированной длины и каждый блок переводится в числовой эквивалент $\mathbf{S_i}$
- Каждый блок шифруется с помощью открытого ключа по формуле:

$$\mathbf{C_i = S_i^e (mod\ n)}$$

- $\mathbf{C_i}$ записывается в выходной файл

Дешифрование

- Читается текущий блок $\mathbf{C_i}$
- Каждый блок дешифруется с помощью закрытого ключа по формуле:

$$\mathbf{S_i = C_i^d (mod\ n)}$$

- $\mathbf{S_i}$ записывается в выходной файл

Алгоритм RSA

Пример

- Выберем простые числа **p**=3 и **q**=11
- Вычислим **n** = **p***q** = 3*11 = 33
- Вычислим **m** = (**p**-1)*(**q**-1) = 2*10 = 20
- Подберем число **d** = 7 взаимно простое с **m** и найдем число **e** = 3 такое, что **d***e** = 1 (mod 20)
- {3,33} – открытый ключ, {7,33} – закрытый ключ
- Зашифруем сообщение BED = {2,5,4} открытым ключом
  - $C_1 = 2^3 \bmod 33 = 8$
  - $C_2 = 5^3 \bmod 33 = 26$
  - $C_3 = 4^3 \bmod 33 = 31$
- Расшифруем сообщение {8, 26, 31} закрытым ключом
  - $S_1 = 8^7 \bmod 33 = 2$
  - $S_2 = 26^7 \bmod 33 = 5$
  - $S_3 = 31^7 \bmod 33 = 4$

Требования к RSA

- Число битов для n должно быть, по крайней мере, 1024. Это означает, что **n** должно быть приблизительно 2^{1024}, или 309 десятичных цифр.
- Два простых числа **p** и **q** должны каждый быть по крайней мере 512 битов. Это означает, что **p** и **q** должны быть приблизительно 2^{512} или 154 десятичными цифрами.
- Значения **p** и **q** не должны быть очень близки друг к другу.
- **p – 1** и **q – 1** должны иметь по крайней мере один большой простой сомножитель.
- Отношение **p/q** не должно быть близко к *рациональному числу* с маленьким *числителем* или знаменателем

RSA: Условие 1

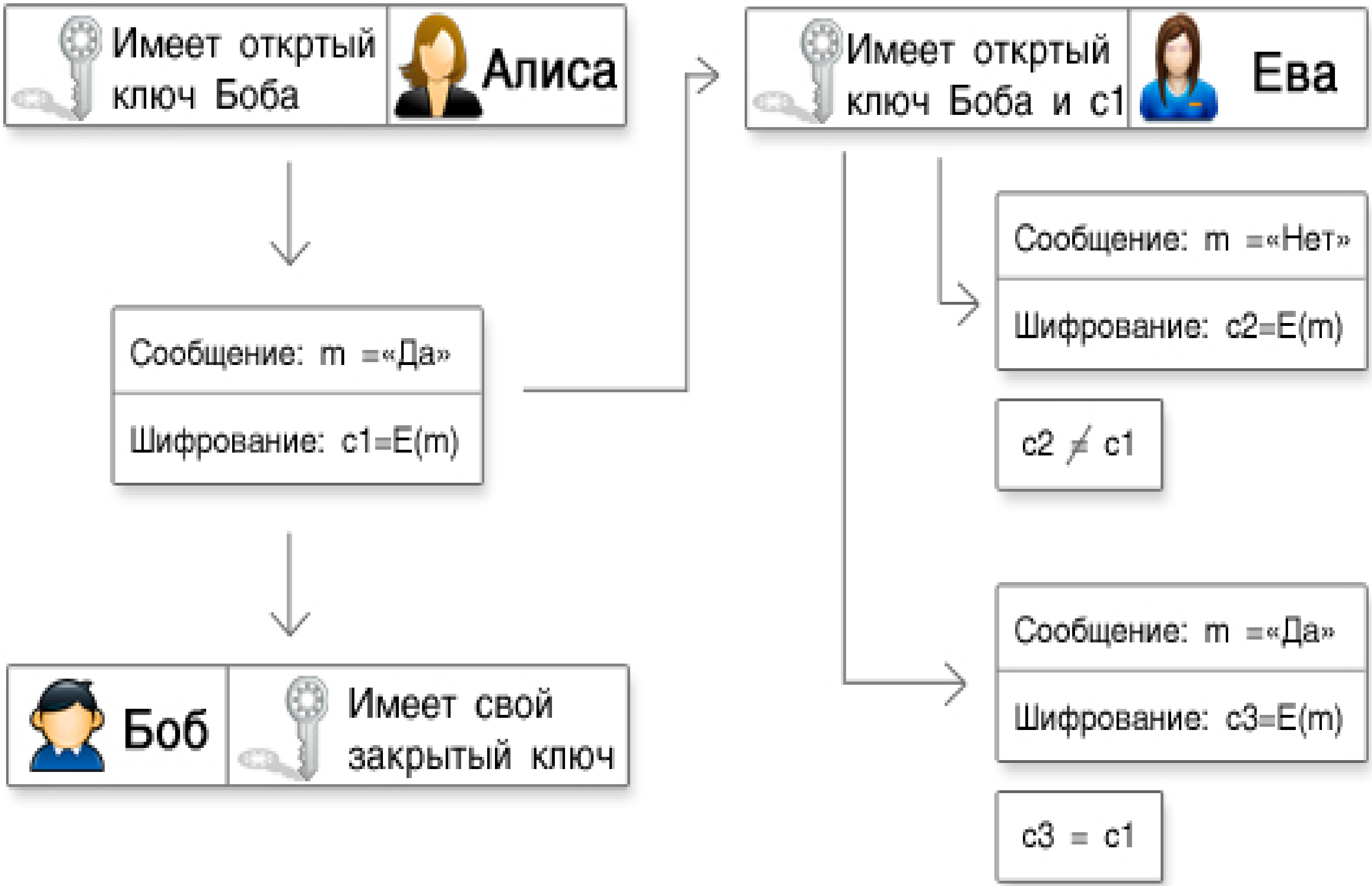
Ассиметричный алгоритм шифрования является **стойким**, если атакующий имеет два открытых текста $\mathbf{M_1}$ и $\mathbf{M_2}$, а так же зашифрованный текст $\mathbf{C_i}$, но не может с вероятностью большей, чем 1/2 определить к какому из сообщений $\mathbf{M_1}$ или $\mathbf{M_2}$ относится $\mathbf{C_i}$

Проверка условия 1

Боб спрашивает у Алисы: «Алиса, мы идем сегодня в кино», причем сообщение не шифруется. Алиса отвечает Бобу, но не хочет, чтобы кто-то знал, поэтому шифрует свой ответ на открытом ключе Боба и отправляет шифротекст Бобу.

Ева перехватывает зашифрованное сообщение и знает, что Алиса ответила либо «Да», либо «Нет». Ева располагает открытым ключом Боба, поэтому последовательно шифрует сообщение «Да» и «Нет», соответственно одно из них совпадет с зашифрованным сообщением Алисы и Ева узнает, пойдет ли Алиса сегодня в кино или нет

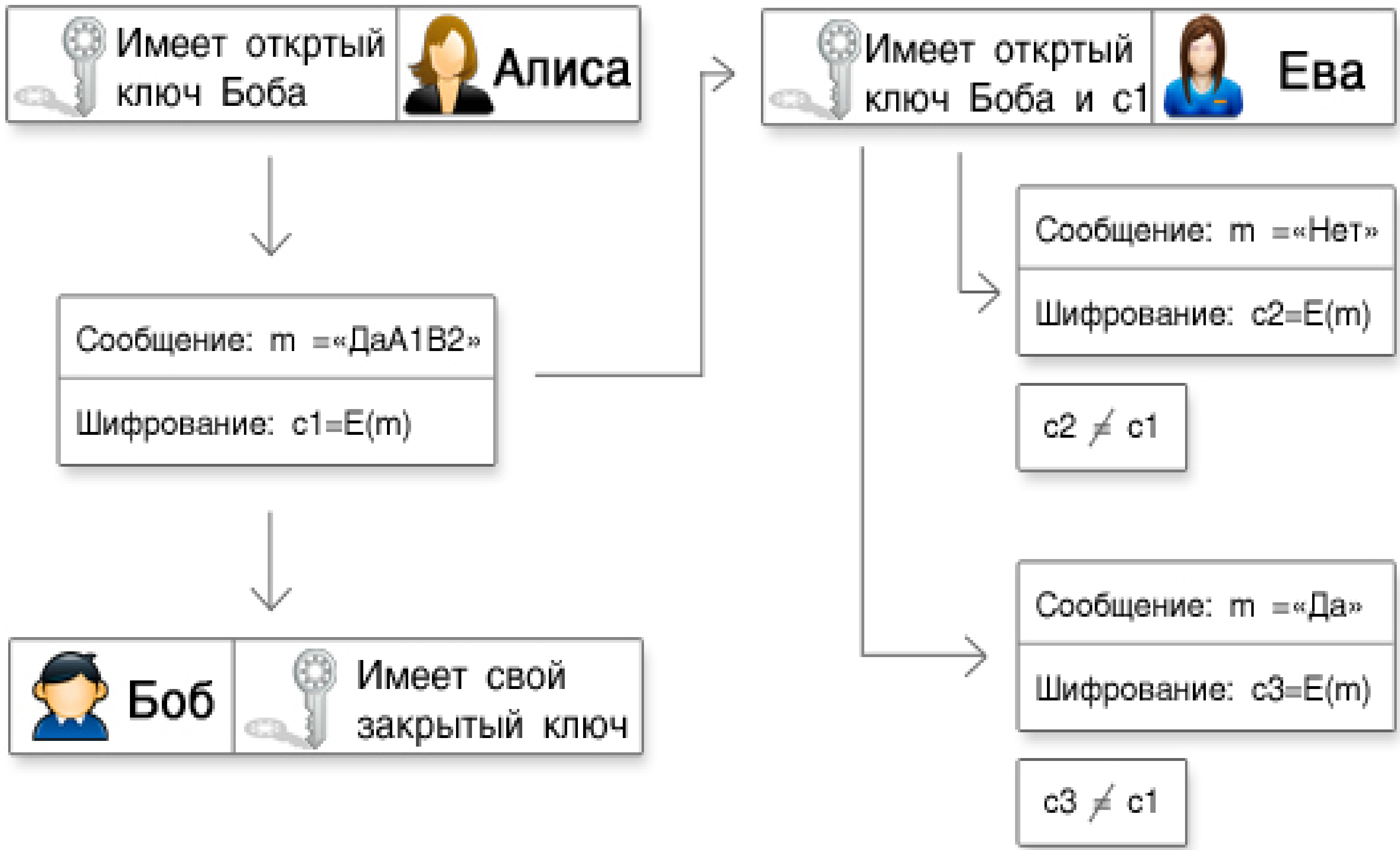
Не выполнение условия1. Ева прослушивает канал связи



Решение проблемы

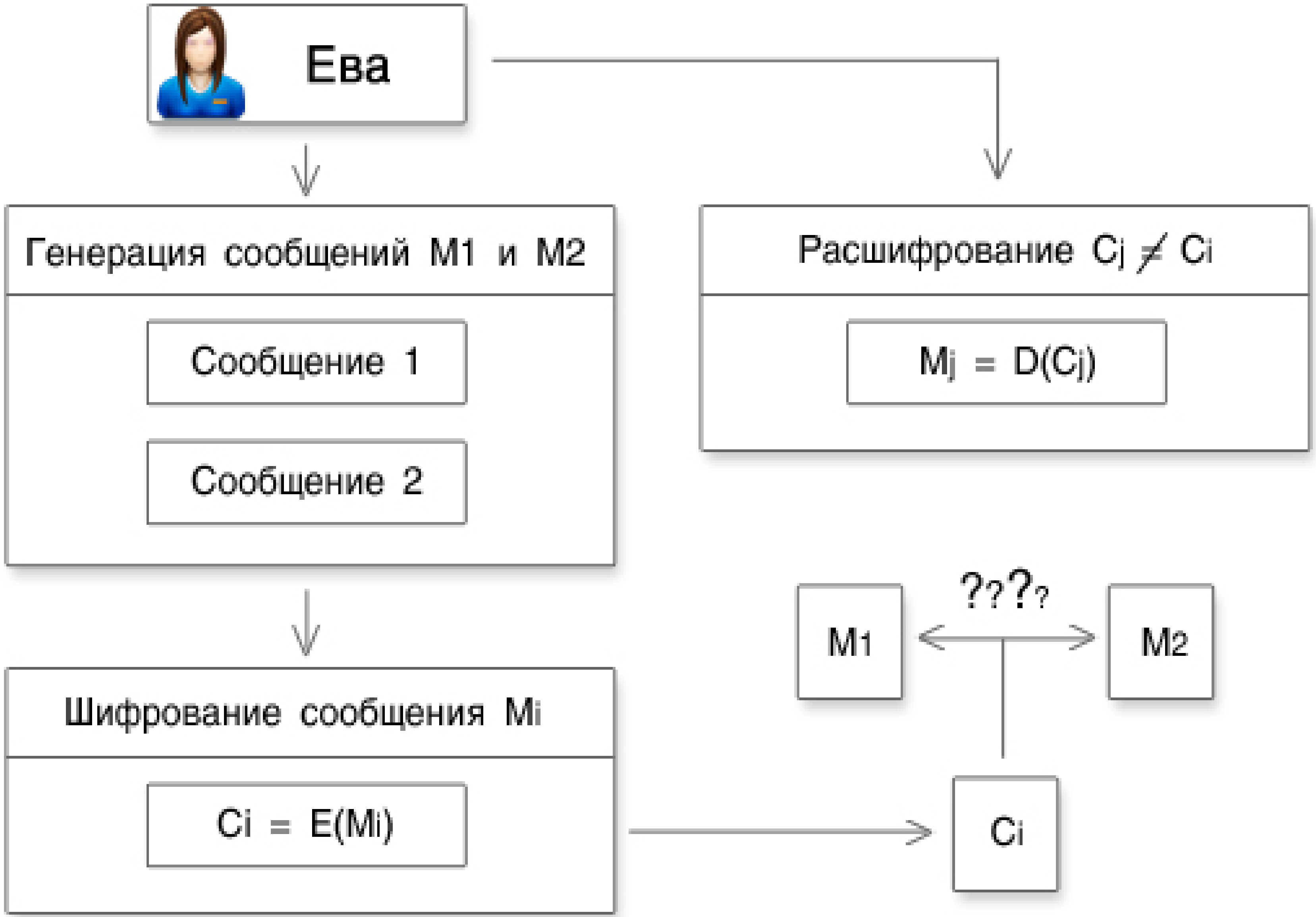
1. Использование системы Эль-Гамаля со случайным сеансовым ключом **K**
2. К сообщению добавляется некоторая случайная величина, а затем полученный текст шифруется. Таким образом, если Ева перехватывает сообщение C_1=E("ДаA1B2"), то зашифровав «Нет» и «Да» C_2= E("Нет") , C_3=E("Да"), будет видно, что C_1, C_2 и C_3 не совпадут.

Выполнение условия1. Ева прослушивает канал связи



RSA: Условие 2

Допустим, у Евы есть две функции, одна F_1 шифрует сообщения, вторая F_2 расшифровывает шифротекст. Затем Ева генерирует два сообщения M_1 и M_2. Затем наугад одно из сообщений шифруется функцией F_1, на выходе функции - шифротекст C_i. C_i возвращается Еве, её задача угадать с вероятностью большей, чем 1/2 к какому из сообщений M_1 или M_2 принадлежит C_i. При этом Ева может расшифровать любое сообщение, кроме C_i (иначе задача лишена смысла). Считается, что криптосистема с открытым ключом стойкая, если злоумышленник не может с вероятностью большей, чем 1/2 сказать какому из сообщений соответствует шифротекст.


Ева
Генерация сообщений M1 и M2
Сообщение 1
Сообщение 2
Шифрование сообщения Mi
Ci = E(Mi)
Расшифрование Cj ≠ Ci
Mj = D(Cj)
M1
M2
Ci
????

Проверка условия 2

Пусть у Евы есть два открытых сообщения M_1 и M_2 и один шифротекст $C_i = M_1^e mod(N)$

Ева создает сообщение, используя открытый ключ (*e*,*N*): $C^* = 2^e\, C_i\; mod(N)$

затем используя функцию F_2 расшифровывает это сообщение, таким образом:

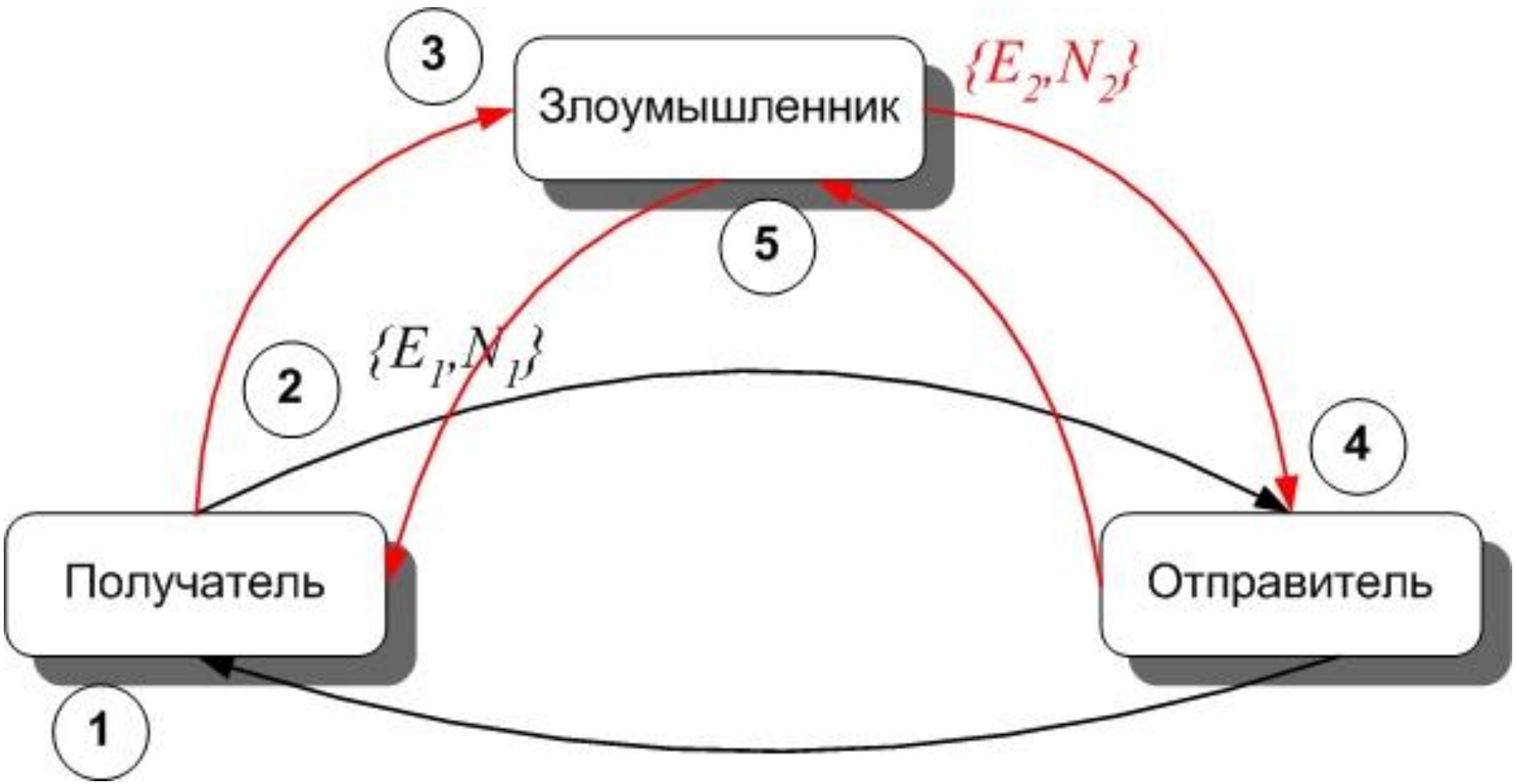
$$M^* = C^{*d} mod(N) = 2^{ed} * M_1^{ed} mod(N) = 2\; M_1\; mod(N)$$

Вычисляя $M^*/2$ Ева получит сообщение M_1

Решение проблемы

1. Использование системы Эль-Гамаля со случайным сеансовым ключом **K**
2. К сообщению добавляется некоторая случайная величина, а затем полученный текст шифруется. Таким образом, если Ева перехватывает сообщение C_1=E("ДаA1B2"), то зашифровав «Нет» и «Да» C_2= E("Нет") , C_3=E("Да"), будет видно, что C_1, C_2 и C_3 не совпадут.

Уязвимость в алгоритме RSA



Система Эль-Гамаля

Схема была предложена Тахером Эль-Гамалем в 1985 году. Эль-Гамаль разработал один из вариантов алгоритма Диффи-Хеллмана. Он усовершенствовал систему Диффи-Хеллмана и получил два алгоритма, которые использовались для шифрования и для обеспечения аутентификации. В отличие от RSA алгоритм Эль-Гамаля не был запатентован и, поэтому, стал более дешевой альтернативой, так как не требовалась оплата взносов за лицензию.

Это криптосистема с открытым ключом, основанная на трудности вычисления дискретных логарифмов в конечном поле. Криптосистема включает в себя алгоритм шифрования и алгоритм цифровой подписи.

Генерация ключей Эль-Гамаля

1. Генерируется случайное простое число **p**.
2. Выбирается целое число **g** взаимно простое с **p**
3. Выбирается случайное целое число **x** такое, что **1 < x < p**
4. Вычисляется **y = g^x (mod p)**
5. Результат генерации:
   - **{p, g, y}** – открытый ключ
   - **{x}** - закрытый ключ

Шифрование Эль-Гамаля

1. Выбирается **сессионный ключ** — случайное целое число **K** такое, что **1 < k < p - 1**
2. Вычисляются числа:
   - $\mathbf{a = g^K \ (mod\ p)}$
   - $\mathbf{b} = \mathbf{y^K\ M\ (mod\ p)}$
3. Пара чисел **{a,b}** является шифротекстом.

Нетрудно увидеть, что длина шифротекста в схеме Эль-Гамаля длиннее исходного сообщения **M** вдвое

Дешифрование Эль-Гамаля

1. **$M = B / (A^x) \bmod p$**

Альтернативный вариант без операции деления:

1. **$y = g^x \bmod p$**
2. **$a = g^k \bmod p$**
3. **$b = M \oplus (y^k \bmod p)$**
4. **$M = (a^x \bmod p) \oplus b$**