

Лекция №5

Протокол обмена ключами Диффи-Хэллмана

Елена Киршанова
Курс “Основы криптографии”

До сих пор...

- Псевдослучайные генераторы
- Поточковые шифры
- Блок-шифры
- Коды Аутентификации сообщений (MAC)
- Хэш-функции
- Шифрование с аутентификацией

Кроме псевдослучайных генераторов, **все** примитивы подразумевают наличие **общего секретного ключа**.

Как двум сторонам сгенерировать секретный ключ?

Обмен ключами Диффи-Хэллмана (Diffie-Hellman Key Exchange)



Whitfield Diffie ©Wikipedia



Martin Hellman ©Wikipedia

- протокол предложен Whitfield Diffie, Martin Hellman в 1976 *"New Directions in Cryptography"*
- используется в современных протоколах (TLS, Signal)
- в основе трудность задачи дискретного логарифма

Обмен ключами: определение

Сценарий: Алиса и Боб имеют общий канал связи. Их цель: сгенерировать общий секретный ключ

Злоумышленник – пассивный участник, просматривающий канал связи

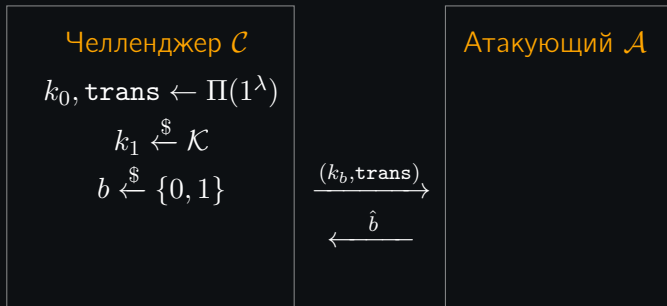
Протокол **обмена ключами** – ppt интерактивный алгоритм, состоящий из двух алгоритмов:

1. $\text{GenParam}(1^\lambda) \rightarrow \text{param}$
2. $\text{KeyGen}(\text{param}) \rightarrow k \in \mathcal{K}$ – интерактивный протокол.

Передаваемая информация по каналу – **trans**

Обмен ключами: безопасность

$\Pi = (\text{GenParam}, \text{KeyGen})$ – протокол обмена ключами



$W_{\Pi, \mathcal{A}}$ – событие $\hat{b} == b$.

$\text{Adv} = \left| \Pr[W_{\Pi, \mathcal{A}}] - \frac{1}{2} \right|$ – выигрыш \mathcal{A} .

Протокол обмена ключами Π безопасен относительно пассивного атакующего, если для любого ppt \mathcal{A} :

$$\text{Adv} = \text{negl}(\lambda).$$

Часть II

Крэш-курс по конечным полям.

Модульная арифметика

Возьмем p – большое простое число (~ 2000 бит)

- $\mathbb{Z}_p = \{0, 1, \dots, p-1\}$ – конечное поле
- умножение/сложение в \mathbb{Z}_p производится по модулю p , т.е. для $x, y \in \mathbb{Z}_p$

$$x + y \bmod p = \text{rem}(x + y, p)$$

$$x \cdot y \bmod p = \text{rem}(x \cdot y, p)$$

rem – остаток от целочисл. деления

Модульная арифметика

Возьмем p – большое простое число (~ 2000 бит)

- $\mathbb{Z}_p = \{0, 1, \dots, p-1\}$ – конечное поле
- умножение/сложение в \mathbb{Z}_p производится по модулю p , т.е. для $x, y \in \mathbb{Z}_p$

$$x + y \bmod p = \text{rem}(x + y, p)$$

$$x \cdot y \bmod p = \text{rem}(x \cdot y, p)$$

rem – остаток от целочисл. деления

Пример: $p = 7, \mathbb{Z}_p = \{0, 1, 2, 3, 4, 5, 6\}$

$$5 + 6 \bmod p = \text{rem}(11, 7) = 4$$

$$3 \cdot 3 \bmod p = \text{rem}(9, 7) = 2$$

Модульная арифметика

Возьмем p – большое простое число (~ 2000 бит)

- $\mathbb{Z}_p = \{0, 1, \dots, p-1\}$ – конечное поле
- умножение/сложение в \mathbb{Z}_p производится по модулю p , т.е. для $x, y \in \mathbb{Z}_p$

$$x + y \bmod p = \text{rem}(x + y, p)$$

$$x \cdot y \bmod p = \text{rem}(x \cdot y, p)$$

rem – остаток от целочисл. деления

Пример: $p = 7, \mathbb{Z}_p = \{0, 1, 2, 3, 4, 5, 6\}$

$$5 + 6 \bmod p = \text{rem}(11, 7) = 4$$

$$3 \cdot 3 \bmod p = \text{rem}(9, 7) = 2$$

- Для ненулевого $x \in \mathbb{Z}_p$, $\exists x^{-1} \in \mathbb{Z}_p$: $x \cdot x^{-1} = 1 \bmod p$
Пример.: $1^{-1} = 1, 2^{-1} = 4, 3^{-1} = 5, 4^{-1} = 2, 5^{-1} = 3, 6^{-1} = 6$ в \mathbb{Z}_7
- Множество обратимых элементов $\mathbb{Z}_p^* = \mathbb{Z}_p \setminus \{0\}$

Структура \mathbb{Z}_p^*

- **Теорема Ферма:** $g^{p-1} = 1 \pmod p \quad \forall 0 \neq g \in \mathbb{Z}_p$

Пример.: $2^6 = 64 = 1 \pmod 7$

- \mathbb{Z}_p^* – **циклическая группа**, т.е.,
 $\exists g \in \mathbb{Z}_p$ s.t. $\mathbb{Z}_p^* = \{1 = g^0, g^1, g^2, \dots, g^{p-2}\}$

Пример: $\mathbb{Z}_7^* = \{1, 3, 3^2 = 2, 3^3 = 6, 3^4 = 4, 3^5 = 5\}$

- Не каждый элемент является образующим \mathbb{Z}_p , но мы знаем, как его эффективно отыскать
- **Порядок** $g \in \mathbb{Z}_p$, $\text{ord}(g)$ – **наименьшее** положительное a т.ч. $g^a = 1$
Для любого образующего g , $\text{ord}(g) = p - 1$.

Вычисления в \mathbb{Z}_p

Эффективность арифметики в \mathbb{Z}_p измеряется в $n = \lceil \log p \rceil$

- Сложение: $\mathcal{O}(n)$ битовых операций
- Умножение: $\mathcal{O}(n^2)$ (или $\mathcal{O}(n^{1.7})$) битовых операций
- Нахождение обратного: $\mathcal{O}(n^2)$ битовых операций

Вычисления в \mathbb{Z}_p

Эффективность арифметики в \mathbb{Z}_p измеряется в $n = \lceil \log p \rceil$

- Сложение: $\mathcal{O}(n)$ битовых операций
- Умножение: $\mathcal{O}(n^2)$ (или $\mathcal{O}(n^{1.7})$) битовых операций
- Нахождение обратного: $\mathcal{O}(n^2)$ битовых операций
- Возведение в степень, x^r : $\mathcal{O}(\log r)$ умножений в \mathbb{Z}_p (быстрое возведение в степень)
 - $y \leftarrow g, z \leftarrow 1$
 - **for** i **in** $[0, n]$:
 - **if** $r[i] == 1$: $z \leftarrow z \cdot y$
 - $y \leftarrow y^2$
 - **return** z

Вычисления в \mathbb{Z}_p

Эффективность арифметики в \mathbb{Z}_p измеряется в $n = \lceil \log p \rceil$

- Сложение: $\mathcal{O}(n)$ битовых операций
- Умножение: $\mathcal{O}(n^2)$ (или $\mathcal{O}(n^{1.7})$) битовых операций
- Нахождение обратного: $\mathcal{O}(n^2)$ битовых операций
- Возведение в степень, x^r : $\mathcal{O}(\log r)$ умножений в \mathbb{Z}_p (быстрое возведение в степень)
 - $y \leftarrow g, z \leftarrow 1$
 - **for** i **in** $[0, n]$:
 - **if** $r[i] == 1$: $z \leftarrow z \cdot y$
 - $y \leftarrow y^2$
 - **return** z

Пример: вычислим g^r для $r = 23 = (10111)_2$. I.e., $g^{23} = g^{16+4+2+1}$.

$$g^1 \rightarrow g^{1+2} \rightarrow g^{1+2+4} \rightarrow g^{1+2+4} \rightarrow g^{1+2+4+16}$$

Эти операции **эффективны** в \mathbb{Z}_p

Часть III

Задача Диффи-Хэллмана. Протокол Диффи-Хэллмана

Трудные (сегодня) задачи в \mathbb{Z}_p

1. Задача дискретного логарифма (dlog):

Для g — образующего \mathbb{Z}_p^* и $x \in \mathbb{Z}_p^*$, найти r т.ч. $g^r = x \bmod p$

Пример: Для $\langle 3 \rangle = \mathbb{Z}_7^*$ и 5 найти $r = 5$ ($3^5 = 5$).

2. Задача Diffie-Hellman (вычислительная версия) (CDH):

Для g — образующего \mathbb{Z}_p^* , $x = g^r \in \mathbb{Z}_p^*$, $y = g^t \in \mathbb{Z}_p^*$ найти $z = g^{r \cdot t} \bmod p$.

Пример: Для $\langle 3 \rangle = \mathbb{Z}_7^*$ и $x = 2, y = 6$ найти $z = 3^5 = 5 \bmod p$.

3. Задача принятия решения Diffie-Hellman (DDH):

Для g — образующего \mathbb{Z}_p^* , $a, b, c \xleftarrow{\$} \mathbb{Z}_p^*$, отличить тройки

$$(g^a, g^b, g^{ab}) \quad (g^a, g^b, g^c)$$

Трудные (сегодня) задачи в \mathbb{Z}_p

1. Задача дискретного логарифма (dlog):

Для g – образующего \mathbb{Z}_p^* и $x \in \mathbb{Z}_p^*$, найти r т.ч. $g^r = x \bmod p$

Пример: Для $\langle 3 \rangle = \mathbb{Z}_7^*$ и 5 найти $r = 5$ ($3^5 = 5$).

2. Задача Diffie-Hellman (вычислительная версия) (CDH):

Для g – образующего \mathbb{Z}_p^* , $x = g^r \in \mathbb{Z}_p^*$, $y = g^t \in \mathbb{Z}_p^*$ найти $z = g^{r \cdot t} \bmod p$.

Пример: Для $\langle 3 \rangle = \mathbb{Z}_7^*$ и $x = 2, y = 6$ найти $z = 3^5 = 5 \bmod p$.

3. Задача принятия решения Diffie-Hellman (DDH):

Для g – образующего \mathbb{Z}_p^* , $a, b, c \xleftarrow{\$} \mathbb{Z}_p^*$, отличить тройки

$$(g^a, g^b, g^{ab}) \quad (g^a, g^b, g^c)$$

$$\text{DDH} \leq \text{CDH} \leq \text{DLOG}$$

$A \leq B$ = “решение для B дает решение для A ”

В общем случае редукции в обратную сторону не известны.

Сложность задач DLOG/CHD/DDH?

Лучший из известных на сегодня алгоритмов для вычисления DLOG в \mathbb{Z}_p^* для $n = \lceil \log p \rceil$: **General Number Field Sieve** работает за суб-экспоненциальное время

$$e^{n^{1/3}}$$

Для уровня безопасности $\lambda = 128$ бит, необходимо взять

$$n \approx 3072 \text{ бит}$$

Вместо группы \mathbb{Z}_p^* на практике используется группа рациональных точек эллиптической кривой.

Преимущество: известны лишь экспоненциальные (от порядка группы) алгоритмы для задачи dlog.

Обмен ключами Диффи-Хэллмана

$\text{GenParam} \rightarrow (g, p)$, где p – большое простое число и $\langle g \rangle = \mathbb{Z}_p^*$

Алиса



Боб



Обмен ключами Диффи-Хэллмана

GenParam $\rightarrow (g, p)$, где p – большое простое число и $\langle g \rangle = \mathbb{Z}_p^*$

Алиса

$$a \leftarrow \{2, \dots, p-2\}$$



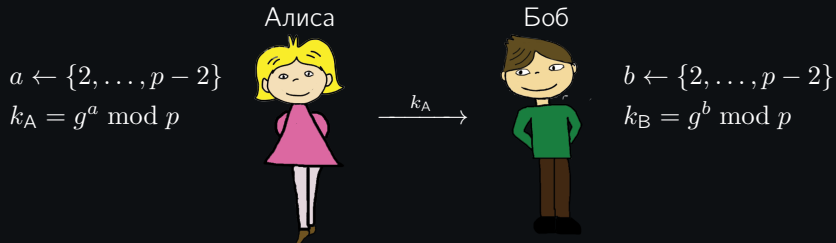
Боб

$$b \leftarrow \{2, \dots, p-2\}$$



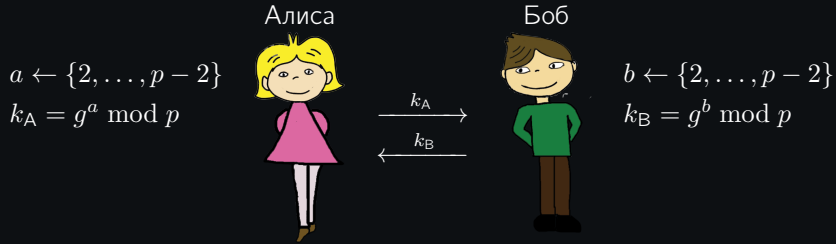
Обмен ключами Диффи-Хэллмана

GenParam $\rightarrow (g, p)$, где p – большое простое число и $\langle g \rangle = \mathbb{Z}_p^*$



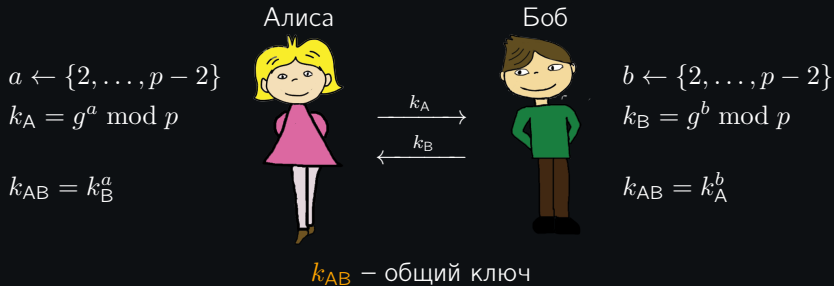
Обмен ключами Диффи-Хэллмана

GenParam $\rightarrow (g, p)$, где p – большое простое число и $\langle g \rangle = \mathbb{Z}_p^*$



Обмен ключами Диффи-Хэллмана

$\text{GenParam} \rightarrow (g, p)$, где p – большое простое число и $\langle g \rangle = \mathbb{Z}_p^*$



Корректность: $k_B^a = (g^b)^a = g^{ab} = (g^a)^b = k_A^b$.

Безопасность (неформально): атакующий видит траскрипт g^a, g^b .
Для того, чтобы различить g^{ab} от случайного элемента \mathbb{Z}_p^* , он должен решить задачу **ДНН**.

Безопасность протокола Диффи-Хэллмана

Теорема. Протокол Диффи-Хэллмана безопасен относительно пассивного атакующего под предположением сложности задачи **DHH**.

Челленджер \mathcal{C}

$$\text{trans} = (g^a, g^b), k_0 = g^{ab}$$

$$k_1 = g^c \rightarrow \mathbb{Z}_p^*$$

$$b \xleftarrow{\$} \{0, 1\}$$

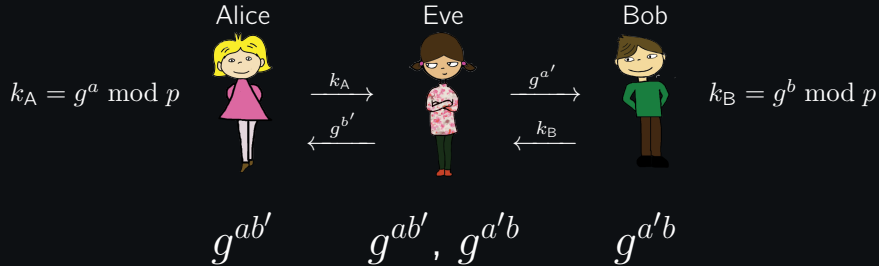
$$\xrightarrow{(k_b, \text{trans})}$$

$$\xleftarrow{\hat{b}}$$

Атакующий \mathcal{A}

Активная атака “человек по середине” (Man-in-the-middle attack)

Протокол Диффи-Хэллмана в “чистом виде” подвержен **активным** атакам.



Фикс: использовать (ассиметрическую) аутентификацию.
См. следующую лекцию

Протокол Диффи-Хэллмана на практике

- Атака “человек по середине” предотвращается с помощью цифровой подписи
- В реальных приложениях протокол ДХ вместо группы \mathbb{Z}_p^* использует группу рац, точек эллиптической кривой. Соответствующая задача: **EDDH**.
- Построить подходящую группу для задач DLOG, CDH, DDH **нетривиальна!** Не придумывайте свою, используйте стандарты.
- См. <https://safecurves.cr.yp.to/> для выбора хорошей кривой
- ГОСТа для протокола обмена ключами нет, есть RFC <https://www.ietf.org/rfc/rfc5246.txt> и рекомендации.

Часть IV

РАКЕ. Парольный обмен ключами с аутентификацией

Password Authenticated Key Exchange (PAKE): мотивация

- Классический Диффи-Хэллмана (ДХ) протокол уязвим к активной атаке
- Безопасная реализация ДХ подразумевает аутентифицированный канал
- Аутентификация: либо **доверенная сторона**, либо **парольные решения**

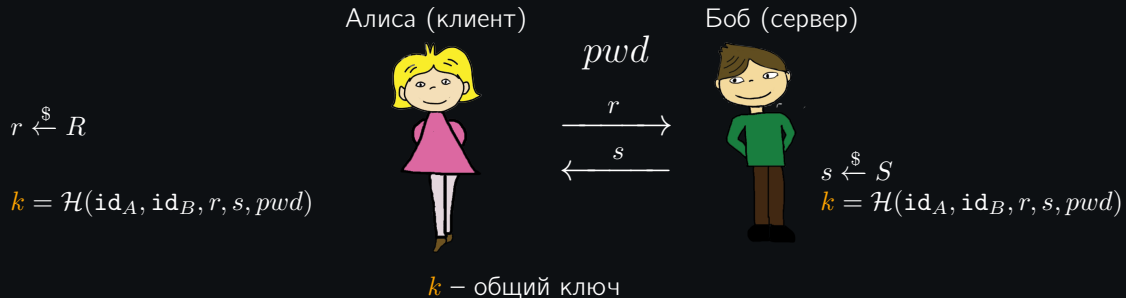
Сценарий: две стороны, сервер и клиент, знают пароль *pwd*

Задача: обезопасить клиента от атакующего, представляющегося сервером.

Упрощенная версия PAKE₀

\mathcal{H} – криптографическая хэш-функция. R, S – множество нонсов

id_A – идентификатор Алисы, id_B – идентификатор Боба



Проблема: Если pwd легко угадывается, пассивный злоумышленник может легко провести атаку перебором на k

Упрощенная версия PAKE₁

\mathcal{H} – криптографическая хэш-функция

id_A – идентификатор Алисы, id_B – идентификатор Боба

$\text{GenParam} \rightarrow (g, p)$, где p – большое простое число и $\langle g \rangle = \mathbb{Z}_p^*$

Алиса (клиент)

Боб (сервер)

$$a \leftarrow \{2, \dots, p-2\}$$

$$w = v^a$$



pwd

u

v



$$b \leftarrow \{2, \dots, p-2\}$$

$$v = u^b$$

$$k = \mathcal{H}(\text{id}_A, \text{id}_B, u, v, w, pwd)$$

PAKE₁ безопасен относительно **пассивных** злоумышленников при условии сложности задачи CDH.

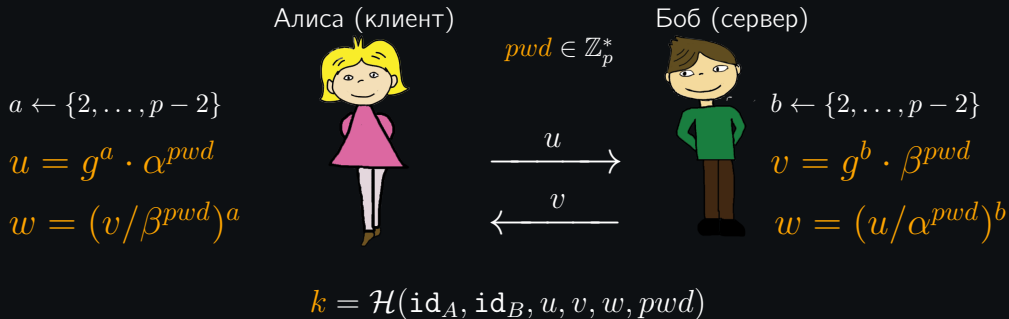
PAKE₁ небезопасен относительно **активного** атакующего, выдающего себя за сервер

Версия РАКЕ₂

\mathcal{H} – криптографическая хэш-функция

id_A – идентификатор Алисы, id_B – идентификатор Боба

Зафиксируем большое простое p , и $\langle g \rangle = \mathbb{Z}_p^*$, $\alpha, \beta \in \mathbb{Z}_p^*$



РАКЕ₂ безопасен относительно **активных** злоумышленников
при условии сложности задачи CDH

PAKE на практике

- Пароли на сервере хранятся в виде $g^{\mathcal{H}(\text{salt}, \text{pwd})}$ (атакующий, получив это значение, не может эффективно вычислить pwd)
- Популярный пример PAKE: Secure Remote Password (SRP) (iCloud Key Vault)
- Другой пример: OPAQUE
<https://tools.ietf.org/html/draft-krawczyk-cfrg-opaque-00> – криптографический стойкий и эффективный PAKE