

# БДЗ по прикладной криптографии

Фирсов Георгий, М21-507

7 мая 2022 г.

## Содержание

Задание 1 . . . . .	2
Задание 2 . . . . .	2
Задание 3 . . . . .	3
Задание 4 . . . . .	3
Задание 5 . . . . .	3
Задание 6 . . . . .	3
Задание 7 . . . . .	3
Задание 8 . . . . .	3
Задание 9 . . . . .	3
Задание 10 . . . . .	3
Задание 11 . . . . .	4
Задание 12 . . . . .	4

## Задание 1

Анна генерирует два числа  $x \xleftarrow{R} \mathbb{Z}_1, y \xleftarrow{R} \mathbb{Z}_q$ , после чего отправляет Борису тройку  $(A_0, A_1, A_2) = (g^x, g^y, g^{xy+a})$ .

Борис генерирует свои два числа  $r \xleftarrow{R} \mathbb{Z}_q, s \xleftarrow{R} \mathbb{Z}_q$ , а затем отправляет Анне следующую пару:  $(B_1, B_2) = (A_1^r \cdot g^s, (A_2/g^b)^r \cdot A_0^s)$ . Заметим, что:

$$\begin{aligned} B_1 &= A_1^r \cdot g^s = g^y \cdot g^s = g^{y+s} \\ B_2 &= (A_2/g^b)^r \cdot A_0^s = g^{xy+a} \cdot g^{-b} \cdot g^{xs} = g^{x(y+s)+a-b} \end{aligned}$$

Если  $B_1$  возвести в степень  $x$  и затем умножить на обратный к полученному элемент число  $B_2$ , то получится  $g^{a-b}$ :

$$\begin{aligned} B_1^x &= (g^{y+s})^x = g^{x(y+s)} \\ B_2 \cdot (B_1^x)^{-1} &= g^{x(y+s)+a-b} \cdot g^{-x(y+s)} = g^{a-b} \end{aligned}$$

Если  $a = b$ , то  $g^{a-b} = g^0 = e_{\mathbb{G}}$ . Это свойство и можно использовать для проверки равенства чисел  $a$  и  $b$ .

**Ответ:** в) Анна проверяет равенство  $B_2/B_1^x = 1$ .

## Задание 2

Так как числа  $p, a, b$  общеизвестны, то считаю, что при разработке программы все *возможные* вычисления с данными параметрами выполняются заранее (то есть, собственно, на этапе разработки программы). Несложно увидеть, что:

$$\begin{aligned} H^{(n)}(x) &= \underbrace{H_{p,a,b}(H_{p,a,b}(\cdots H_{p,a,b}(x) \cdots))}_{n \text{ раз}} = \underbrace{a(a(\cdots ax + b \cdots) + b)}_{n \text{ раз}} = \\ &= a^n x + b \sum_{j=0}^{n-1} a^j \pmod p \end{aligned}$$

Обозначим:

$$\begin{aligned} a' &:= a^n \pmod p \\ b' &:= b \sum_{j=0}^{n-1} a^j \pmod p \end{aligned}$$

Тогда:

$$H^{(n)}(x) = a'x + b' \pmod p$$

Значения  $a'$  и  $b'$  возможно вычислить предварительно на этапе разработки программы, что позволит вычислять функцию  $H^{(n)}$  так же быстро, как и  $H_{p,a,b}$ .

Но может случиться так, что числа  $p, a, b$  *заранее* не известны (например, меняются с течением времени). Таким образом, возникает потребность поддержки вычисления  $a', b'$  на лету. В таком случае заметим, что:

$$\sum_{j=0}^{n-1} a^j = (a^n - 1) \cdot (a - 1)^{-1} \pmod p$$

При больших  $n$  возведение в степень  $n - 1$  потребует примерно столько же операций, сколько и возведение в степень  $n$ . Заметим, что возведение в степень можно производить, пользуясь следующей идеей:  $a^4 = a^2 \cdot a^2, a^8 = a^4 \cdot a^4$  и т.д. Данный алгоритм требует асимптотически  $\log_2(n)$  умножений.

**Ответ:** а)  $H^{(n)}$  может быть вычислена так же быстро, как  $H_{p,a,b}$  (в случае известных заранее значений  $p, a, b$ ); г) вычисление  $H^{(n)}$  требует времени  $O(\log n)$  (в случае неизвестных заранее значений  $p, a, b$ ).

### Задание 3

Рассмотрим по очереди все варианты, отобрав подходящие:

- $p_1 = (k_1, k_2), p_2 = (k'_1), p_3 = (k'_2)$ : владельцы долей  $p_2, p_3$  не смогут вдвоем восстановить ключ, так как  $k'_1 \oplus k'_2 = ???$ .
- $p_1 = (k_1, k_2), p_2 = (k_2, k'_2), p_3 = (k'_2)$ : владелец  $p_2$  может один восстановить ключ, так как  $k = k_2 \oplus k'_2$ .
- $p_1 = (k_1, k_2), p_2 = (k_1, k_2), p_3 = (k'_2)$ : владельцы  $p_1, p_2$  не смогут восстановить вдвоем ключ, так как никакая комбинация  $k_1, k_2$  в сумме не даст  $k$ .
- $p_1 = (k_1, k_2), p_2 = (k'_1, k'_2), p_3 = (k'_2)$ : владельцы  $p_2, p_3$  не смогут восстановить вдвоем ключ, так как никакая комбинация  $k'_1, k'_2$  в сумме не даст  $k$ .
- $p_1 = (k_1, k_2), p_2 = (k'_2), p_3 = (k'_1, k_2)$ : данный вариант подходит, так как:
  - $p_1, p_2$ :  $k_2 \oplus k'_2 = k$
  - $p_1, p_3$ :  $k_1 \oplus k'_1 = k$
  - $p_2, p_3$ :  $k'_2 \oplus k_2 = k$

При этом восстановление ключа ни одним участником единолично невозможно, так как ни один из них не обладает двумя частями с одинаковыми индексами.

**Ответ:** д)  $p_1 = (k_1, k_2), p_2 = (k'_2), p_3 = (k'_1, k_2)$

### Задание 4

### Задание 5

### Задание 6

### Задание 7

### Задание 8

### Задание 9

### Задание 10

1. *Некорректное хэширование.* Так как требуется подделка подписи для любого сообщения, то рассматривается модель стойкости UUF-CMA (Universal Unforgeability under Chosen Message Attack). Рассмотрим противника  $\mathcal{A}$ , действующего по следующему сценарию:

- $\mathcal{A}$  генерирует некоторое сообщение  $m \in \mathcal{M}$  (любое).
- $\mathcal{A}$  вычисляет хэш сообщения:  $c = H(m)$ .
- $\mathcal{A}$  вычисляется значение  $R = h^{-c}$  ( $h = pk$ ), значение  $z$  полагается равным 0.
- Подпись  $\sigma = (R, z) = (pk^{-H(m)}, 0)$  отправляется оракулу проверки подписи.

Заметим, что проверка этой подписи выполнится всегда:

$$\left. \begin{array}{l} c \leftarrow H(m) \\ g^z = e_{\mathbb{G}} \\ R \cdot h^c = h^{-H(m)} \cdot h^{H(m)} = h^0 = e_{\mathbb{G}} \end{array} \right\} \implies V(pk, m, \sigma) = \text{accept}$$

Таким образом:

$$\mathbf{Adv}_{\text{SS}}^{\text{UUF-CMA}}(\mathcal{A}) = \Pr[V(pk, m, S(sk, m)) = \text{accept}] = 1$$

2. *Некорректная генерация случайных чисел.* Обозначим  $c_i = H(m_i, R_i), i \in \{0, 1\}$  и, зная, что  $\rho_1 = a\rho_0 + b$ , запишем:

$$\begin{cases} z_0 = \rho_0 + c_0\alpha \\ z_1 = a\rho_0 + b + c_1\alpha \end{cases}$$

$$\begin{cases} az_0 = a\rho_0 + ac_0\alpha \\ z_1 = a\rho_0 + b + c_1\alpha \end{cases}$$

Тогда:

$$\begin{aligned} z_1 - az_0 &= b + \alpha(c_1 - ac_0) \\ \alpha &= (z_1 - az_0 - b)(c_1 - ac_0)^{-1} \end{aligned}$$

В общем-то  $(c_1 - ac_0)$  – некоторый случайный элемент из  $\mathbb{Z}_q$ , а так как  $q$  – простое число, то для обратимости элемента требуется только то, чтобы он был ненулевым. Вероятность того, что элемент  $(c_1 - ac_0)$  обратим, равна:

$$\Pr \left[ \begin{array}{c} \text{Элемент } (c_1 - ac_0) \\ \text{обратим в кольцо } \mathbb{Z}_q \end{array} \right] = 1 - \frac{1}{|\mathbb{Z}_q|} = \frac{q-1}{q}$$

**Задание 11**

**Задание 12**