О подходах к синтезу схем подписи, основанных на итеративном использовании функций хэширования

Антон Гуселев,
Иван Лавриков

— ТК 026 —
РусКрипто 2018

С чем будем иметь дело

Что такое постквантовая криптография?

Криптографические протоколы, механизмы и алгоритмы, для которых **не существует эффективных методов анализа**, реализуемых на **квантовом компьютере** будем называть постквантовыми.

Угроза для криптографических механизмов, в основе которых лежат

- блочные критпографические примитивы

и тех, стойкость которых основана на трудной разрешимости задач

- дискретного логарифмирования;
- факторизации.

Как создать постквантовую криптографию?

1. Увеличить длину ключа блочных криптографических примитивов;
2. Использовать подходы на основе:
   - теории решеток;
   - кодов, исправляющих ошибки;
   - систем нелинейных уравнений многих переменных;
   - изогений на суперсингулярных эллиптических кривых;
   - функций хэширования, *пока известны подходы к их использованию при создании только схем подписи*;
   - ...

«Квантовость» функций хэширования

Пока не известно квантовых алгоритмов которые могут быть эффективно применены для построения **коллизии**, **первого** или **второго** прообраза.

Наилучшие результаты по анализу функций хэширования:

- алгоритм Гровера снижает трудоемкость процедуры поиска прообраза с $O(2^n)$ до $O(2^{\frac{n}{2}})$;
- алгоритм Брассарда и др. снижает трудоемкость процедуры поиска коллизии с $O(2^{\frac{n}{2}})$ до $O(2^{\frac{n}{3}})$.

При этом, для современных функций хэширования длина внутреннего состояния является «большой».

Также известно, что $2^{\frac{n}{3}}$ является нижней границей для оценки трудоемкости применения квантовых алгоритмов для поиска коллизий.

Как построить (классическую) схему подписи только с использованием функции хэширования?

Наилучшим, на настоящий момент, считается следующий подход:

1. Взять схему *одноразовой подписи*
   (лучше всего, построенную на основе многократного применения функций хэширования, поскольку именно они считаются наиболее эффективно реализуемыми и стойкими).
2. Взять дерево Меркля
   (или любое другое дерево, в основе построения которого лежат принципы хэширования).
3. Объединить «много» ключей схемы одноразовой подписи в дерево Меркля.

Схема одноразовой подписи

Для подписания каждого сообщения $(m_1, \ldots, m_l)$ необходим уникальный секретный ключ $\mathtt{sk}_{\mathsf{OTS}} = (\mathtt{sk}_1, \ldots, \mathtt{sk}_l)$, на основании которого вычисляется открытый ключ

$$\mathtt{pk}_{\mathsf{OTS}} = (\mathtt{pk}_1, \ldots, \mathtt{pk}_l) = (\mathsf{H}^{w-1}_{\mathtt{sk}_1}(x), \ldots, \mathsf{H}^{w-1}_{\mathtt{sk}_l}(x)).$$

Подписью под сообщением будет

$$\sigma = (\sigma_1, \ldots, \sigma_l) = (\mathsf{H}^{m_1}_{\mathtt{sk}_1}(x), \ldots, \mathsf{H}^{m_l}_{\mathtt{sk}_l}(x)).$$

Проверка подписи – проверка выполнения равенства

$$(\mathsf{H}^{w-1-m_1}_{\sigma_1}(x), \ldots, \mathsf{H}^{w-1-m_l}_{\sigma_l}(x)) = (\mathtt{pk}_1, \ldots, \mathtt{pk}_l).$$

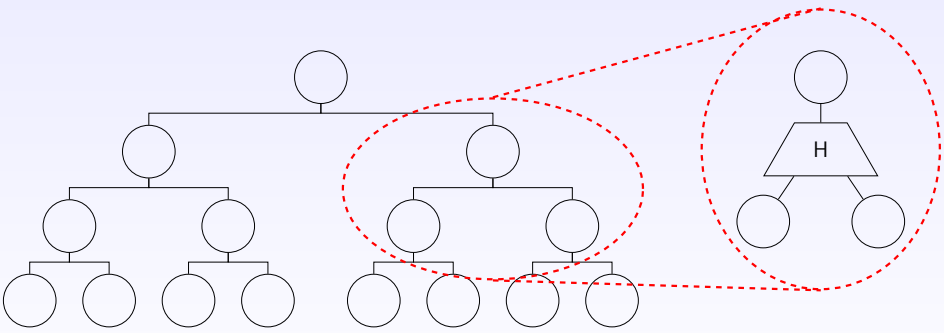
* x – *начальный вектор*, w – *основание системы счисления.*

Как построить (классическую) схему подписи только с использованием функции хэширования?

Наилучшим, на настоящий момент, считается следующий подход:

1. Взять схему *одноразовой подписи*
   (лучше всего, построенную на основе многократного применения функций хэширования, поскольку именно они считаются наиболее эффективно реализуемыми и стойкими).
2. Взять дерево Меркля
   (или любое другое дерево, в основе построения которого лежат принципы хэширования).
3. Объединить «много» ключей схемы одноразовой подписи в дерево Меркля.

Дерево Меркля

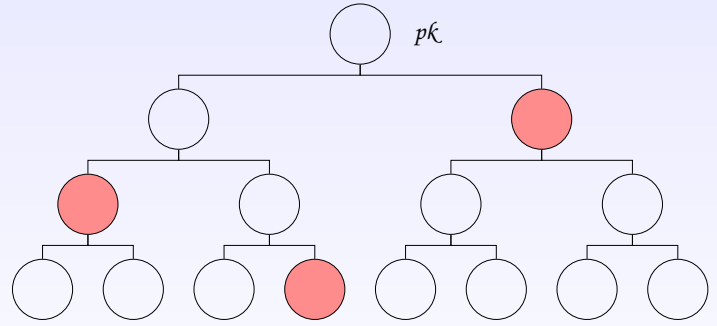


Как построить (классическую) схему подписи только с использованием функции хэширования?

Наилучшим, на настоящий момент, считается следующий подход:

1. Взять схему *одноразовой подписи*
   (лучше всего, построенную на основе многократного применения функций хэширования, поскольку именно они считаются наиболее эффективно реализуемыми и стойкими).
2. Взять дерево Меркля
   (или любое другое дерево, в основе построения которого лежат принципы хэширования).
3. Объединить «много» ключей схемы одноразовой подписи в дерево Меркля.

Схема подписи в классическом смысле



- **Секретный ключ** – набор секретных ключей схемы одноразовой подписи;
- **Открытый ключ** – корень дерева (pk);
- **Подпись** – метки вершин, необходимые для вычисления корня.

*Номер листа (3), от которого начинается вычисление корня, определяется номером секретного ключа схемы одноразовой подписи.

Известные схемы подписи, основанные на использовании функций хэширования

В отраслевых рекомендациях Инженерного совета Интернета (Internet Engineering Task Force, IETF), определяющего политику регулирования сети Интернет, описаны две схемы подписи, в основе которых лежат представленные выше принципы:

- XMSS
- LMS

Схемы отличаются лишь порядком хранения и способом вычисления значений.

Известны и другие схемы подписи (основанные на использовании представленных выше принципов), в частности SPHINCS, однако их характеристики менее исследованы.

Можно ли создать схему подписи без использования деревьев Меркля?

Целесообразно рассмотреть и другие, более прозрачные, подходы к построению схем подписи, основанных на применении функций хэширования.
В основе предлагаемых подходов лежит следующая идея:

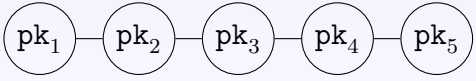
✓ упрощение проведения исследований
✓ снижение трудоемкости реализации

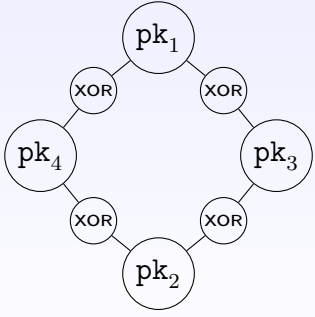
$\Rightarrow$ снижение числа вычисляемых функций хэширования

(На данном этапе схемы одноразовой подписи не затрагиваются.)

– Можно ли создать схему подписи без использования деревьев Меркля?
– А почему нет?

Способы формирования открытого ключа из открытых ключей схемы одноразовой подписи могут быть следующими





Подпись – некоторая информация, зависящая от всех предполагаемых к использованию открытых ключей схемы одноразовой подписи.

Способ № 1

Вычисление открытого ключа:

$$\mathsf{H}\left((pk_1 \| pk_2 \| \ldots \| pk_l) \oplus (vek_1 \| vek_2 \| \ldots \| vek_l)\right) = p\mathcal{K},$$

где vek_i, $i = 1, 2, \ldots, l$ – некоторые секретные векторы.

Вычисление подписи:

1. Определение номера еще не использованного секретно ключа (k) схемы одноразовой подписи.
2. Вычисление дополнительной информации

$$add = pk_1 \oplus vek_1 \| pk_2 \oplus vek_2 \| \ldots \| pk_{k-1} \oplus vek_{k-1} \| vek_k \| \\ \| pk_{k+1} \oplus vek_{k+1} \| \ldots \| pk_l \oplus vek_l.$$

Проверка подписи: проверка выполнения равенства

$$\mathsf{H}\left((0 \| 0 \| \ldots \| 0 \| pk_k \| 0 \| \ldots \| 0) \oplus add\right) = p\mathcal{K},$$

где pk_k – открытый ключ схемы одноразовой подписи.

Способ № 2

Вычисление открытого ключа

$$\mathsf{H}\left(\bigoplus_{i=1}^{l} pk_i\right) = p\mathcal{k}.$$

Вычисление подписи:

1. Определение номера еще не использованного секретно ключа (k) схемы одноразовой подписи.
2. Вычисление дополнительной информации

$$add = pk_1 \oplus pk_2 \oplus \cdots \oplus pk_{k-1} \oplus pk_{k+1} \oplus \cdots \oplus pk_l.$$

Проверка подписи: проверка выполнения равенства

$$\mathsf{H}(pk_k \oplus add) = p\mathcal{k},$$

где pk_k – открытый ключ схемы одноразовой подписи.

Особенности «новых» подходов

Основа стойкости:

- определение ключа = построение прообраза для функции хэширования;
- подделка подписи = построение коллизии для функции хэширования.

Предложенный подход **может не дать** «выигрыша» в части криптографических качеств, однако значительно **упростит** подходы к обоснованию свойств с точки зрения т.н. «доказуемой стойкости».

СПАСИБО ЗА ВНИМАНИЕ!

Вопросы?