1. Цель работы

Реализовать одноразовую подпись на основе хэш-функций согласно выбранному варианту задания: подпись Лэмпорта. Реализация должна использовать стандартную реализацию хэш-функции SHA-256.

2. Описание алгоритма

Пусть у Алисы есть 256-битная криптографическая хеш-функция и криптографически стойкий генератор псевдослучайных чисел. Она хочет создать и использовать пару ключей Лэмпорта – секретный ключ и соответствующий ему открытый ключ.

2.1. Создание пары ключей

Чтобы создать секретный ключ, Алиса использует генератор случайных чисел для получения пар случайных чисел. Эти числа будут секретным ключом Алисы, и она будет хранить их в секретном месте, чтобы использовать в дальнейшем.

Чтобы создать открытый ключ, Алиса хеширует каждое из чисел секретного ключа. Эти хеши составляют открытый ключ Алисы, который она публикует.

2.2. Подпись сообщения

Теперь Алиса хочет подписать сообщение. Для начала она хеширует сообщение. Затем, для каждого бита в этом хеше, она берёт соответствующее число из секретного ключа. Если, например, первый бит в хеше сообщения равен нулю, она берёт *первое* число из первой пары секретного ключа. Если же первый бит равен единице, она использует *второе* число из первой пары. И так далее. В итоге получается набор случайных чисел. Эти числа и составляют подпись, которую Алиса отправляет вместе с сообщением.

Стоит обратить внимание, что после того как Алиса использовала свой секретный ключ, он никогда не должен быть использован снова. Остальные числа, которые она не использовала в подписи, Алиса никогда не должна публиковать или использовать. Предпочтительно, чтобы она их удалила, так

как иначе кто-то может получить к ним доступ и сгенерировать с их помощью поддельную подпись.

2.3. Проверка подписи

Боб хочет проверить подпись, которым Алиса подписала сообщение. Он также хеширует сообщение. Затем для каждого бита в этом хеше он выбирает число из открытого ключа Алисы. Эти числа выбираются по такому же принципу, по какому Алиса выбирала числа для составления подписи. То есть, если первый бит хеша сообщения равен нулю, Боб выбирает *первое* число из первой пары в открытом ключе. И так далее.

Затем Боб хеширует каждое число из подписи Алисы. Если эти хеши в точности соответствуют хешам, которые он только что получил из открытого ключа Алисы, Боб считает подпись подлинной. Если не соответствуют – фальшивой.

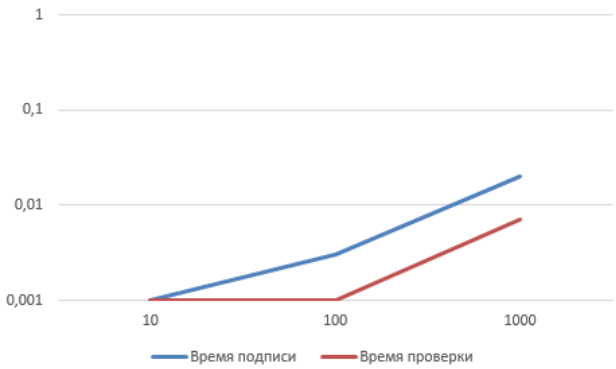
После того, как Алиса опубликует подпись, никто всё ещё не знает остальные числа из каждой пар, и, таким, образом, не может создать подпись для сообщений, имеющих иной хеш.

3. Результаты моделирования

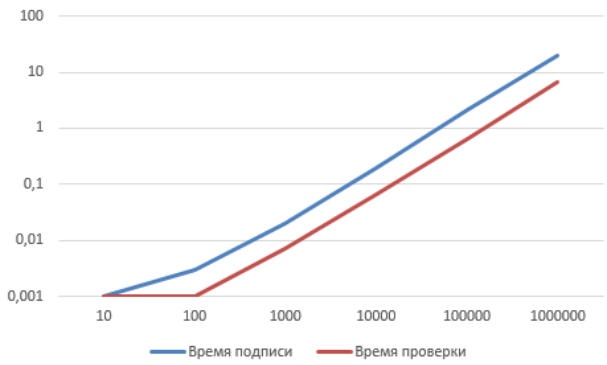
Имеется программа, содержащая независимые функции для генерации подписи, ее проверки и генерации ключей. По умолчанию, программа подписывает случайное сообщение размером 30 бит, но возможно ввести свое сообщение. Сгенерированные ключи записываются в отдельный файл. Сообщение и соответствующая ей подпись также помещаются в другой файл.

На рисунке 1 приведены графики зависимостей времени подписи и проверки подписи от размера сообщения.

На рисунке 2 приведены графики зависимостей размеров секретного, публичного ключей и размера подписи от размера сообщения.

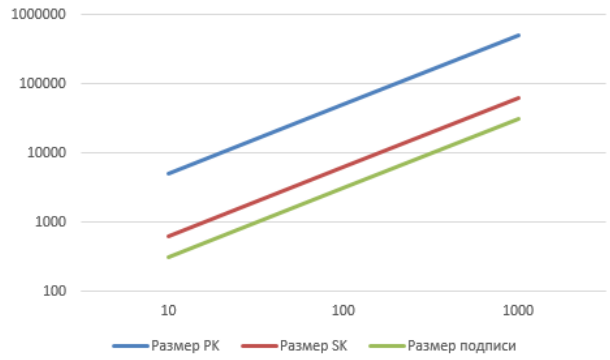


А) Масштаб 1

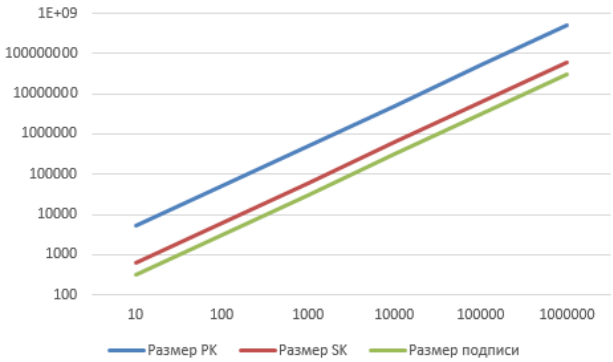


Б) Масштаб 2

Рисунок 1 – Графики зависимостей времени подписи и проверки подписи от размера сообщения



А) Масштаб 1



Б) Масштаб 2

Рисунок 2 – Графики зависимостей размеров секретного, публичного ключей и размера подписи от размера сообщения

4. Выводы

Время подписи алгоритма имеет зависимость от размера входного сообщения близкую к линейной. На графиках оси имеют логарифмические шкалы. Время проверки имеет ту же зависимость, но несколько меньше.

В результате получается подпись, размер которой зависит от размера хеша, в который преобразуется сообщение. На каждый бит хеша приходится 256 бит подписи. Т.к. для подписи используется одно число из пары, то размер подписи меньше размера секретного ключа в два раза. Данные показатели также имеют линейные зависимости.