

1.Задание: реализация одноразовой подписи на дереве Меркле для 16 сообщений.

2.Аннотация:

Подпись Меркла — алгоритм постквантовой и многоразовой цифровой подписи, основанный на использовании дерева Меркла и какой-либо одноразовой цифровой подписи.

Впервые подпись была опубликована Ральфом Мерклом в 1979 в его статье «Secrecy, authentication, and public key systems», как многоразовая цифровая подпись, использующая одноразовую подпись Лэмпорта.

Описание алгоритма

Подготовка

Подпись Меркла базируется на уже существующей одноразовой цифровой подписи, криптостойкость которой должна быть основана на криптостойкой хеш-функции. Примерами таких подписей могут быть Winternitz One-time Signature Scheme или подпись Лэмпорта. Одноразовая цифровая подпись должна состоять из трех основных операций:

1. Генерация секретного ключа X и публичного ключа Y .
2. Генерация подписи b из сообщения d с помощью секретного ключа X .
3. Проверка подписи b с помощью открытого ключа Y .

Также необходимо определиться с криптостойкой хеш-функцией $H: \{0,1\}^* \rightarrow \{0,1\}^s$, которая позже будет использоваться для вычисления публичного ключа.

Генерация ключей

Генерируются массивы ключей X и Y длины $N = 2^n$. Длина выбирается равной степени двойки, так как это необходимо для построения дерева Меркла. Каждая пара (X_i, Y_i) используется как пара закрытый-открытый ключ для базовой одноразовой подписи. Массив X -является закрытым ключом подписи Меркла, для генерации открытого ключа строится дерево Меркла:

Для каждого Y_i вычисляем хеш-значение $H(Y_i)$ — это будет нулевой слой дерева, то есть его листья. Обозначим этот слой a_0 . Всего a_0 содержит $l_0 = 2^n$ элементов. После чего вычисляем следующий a_1 слой, имеющий размер $l_1 =$

2^{n-1} : каждый j -ый элемент слоя a_1 вычисляется через два элемента с предыдущего слоя $a_{1,j} = H(a_{0,j*2} || a_{0,j*2+1})$, где $||$ -операция конкатенации. Таким образом, каждый следующий i -ый слой, имеет длину $l_i = 2^{n-i}$ и вычисляется через $i-1$ слой. В итоге, мы придем к последнему слою дерева a_n , имеющему длину $l_n = 1$ и являющемуся корнем дерева.

За открытый ключ берется корень построенного дерева Меркла.

Генерация подписи.

Для генерации подписи из массивов X и Y выбирается i -ая пара ключей (X_i, Y_i) . Для сообщения d вычисляется одноразовая цифровая подпись b_i с помощью закрытого ключа X_i . Далее рассмотрим путь от корня $a_{0,n}$ дерева Меркла до листа $a_{0,i}$, соответствующему ключу Y_i . Обозначим вершины, через которые проходит этот путь, как массив A длины $n+1$, где A_n -это корень дерева, а A_0 -это $a_{0,i}$. Значение каждой вершины A_j кроме $A_0 = H(Y_i)$, вычисляется как $H(A_{j-1} || auth_{j-1})$, где $auth_{j-1}$ и A_{j-1} являются непосредственными потомками A_j . Таким образом, для того, чтобы вычислить путь от корня дерева достаточно знать Y_i и массив вершин $auth_1, auth_2, auth_3 \dots auth_n$, который будем называть аутентификационным путем. Таким образом, в подпись сообщения d входит: верификационный ключ Y_i , одноразовая подпись b_i , и аутентификационный путь для вычисления пути до корня дерева.

Проверка подписи.

Во-первых, получатель проверяет одноразовую подпись b_i на соответствие сообщению d . Если проверка прошла успешно, то начинает построение пути от $H(Y_i)$ до корня дерева. Сначала вычисляется $A_0 = H(Y_i)$ потом $A_1 = H(A_0 || auth_0)$ и так далее. Если A_n равно открытому ключу.

Преимущества:

Постквантовость

В часто используемых алгоритмах цифровой подписи, таких как RSA и DSA, используется сложность факторизации чисел и сложность вычисления дискретного логарифма. Однако, используя алгоритм Шора и квантовый

компьютер, эти подписи могут быть взломаны за полиномиальное время. В отличие от них подпись Меркла является постквантовой, так как её криптографическая стойкость основана на стойкости криптографической хеш-функции и стойкости одноразовой цифровой подписи.

Многоразовость

Одной из основных проблем цифровых подписей, основанных на криптостойких хеш-функциях, является то, что они, как правило, употребляются в контексте одноразовых цифровых подписей, то есть подписей, в которых одна пара ключей используется для подписи лишь одного сообщения. Однако, существуют способы расширения таких подписей до многоразовых. Одним из таких способов как раз является построение дерева Меркла, использующееся для аутентификации различных одноразовых подписей.

Недостатки:

Проблема обхода дерева

Эта проблема связана с нахождением эффективного алгоритма вычисления аутентификационных данных. Тривиальное решение — сохранить все данные в памяти — требует слишком много памяти. С другой стороны, подход вычисления аутентификационного пути в области, где он нужен, будет слишком затратен для некоторых вершин дерева.[10] Если длина массивов X и Y будет больше, чем 2^{25} , использование подписи Меркла становится непрактичным.

3.Ход работы

Работа программы заключается в создании на основе выбранной хеш-функции и 16 введенных сообщений дерева Меркла и последующей проверки подписи выбранного сообщения.

После ввода 16 сообщений программа использует на каждом из них выбранную хеш-функцию и затем применяет логическое сложение попарно. Затем то же самое делает с получившемся парами. Это продолжается пока не будет достигнут корень дерева.

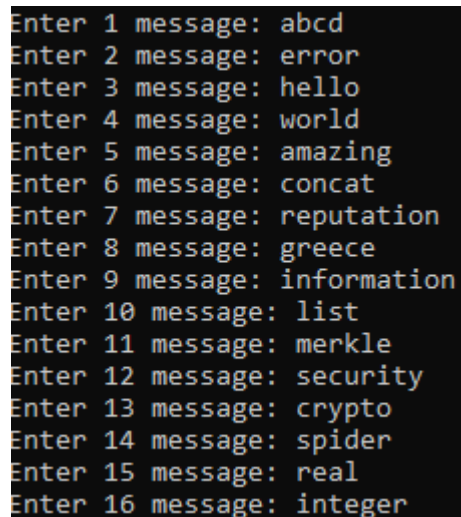
Далее происходит вывод промежуточных результатов и корня дерева.

Затем пользователь выбирает номер сообщения для проверки и запускается алгоритм проверки.

Выбранное сообщение подается на вход выбранной хеш-функции. Затем одноразовая подпись, последовательно логически складывается с каждым значением аутентификационного пути. После этого полученный результат сравнивается с открытым ключом(корнем дерева) . Если результат равен открытому ключу значит результат проверки положительный, если не равен отрицательный.

4.Контрольный пример

При запуске программы пользователь должен ввести 16 сообщений. Введем 16 случайных сообщений в качестве примера:



```
Enter 1 message: abcd
Enter 2 message: error
Enter 3 message: hello
Enter 4 message: world
Enter 5 message: amazing
Enter 6 message: concat
Enter 7 message: reputation
Enter 8 message: greece
Enter 9 message: information
Enter 10 message: list
Enter 11 message: merkle
Enter 12 message: security
Enter 13 message: crypto
Enter 14 message: spider
Enter 15 message: real
Enter 16 message: integer
```

Рис 1.Контрольный пример.

Далее программа выводит hash-коды каждого сообщения(зависит от выбранной hash-функции).

```
596
1131
1085
1063
2257
1596
4934
1520
5981
683
1587
3174
1711
1596
619
2246
```

Рис 2.Контрольный пример.

Далее выводятся промежуточные и окончательный результат работы программы(подпись, основанная на дереве Меркле и выбранной hash-функции):

```
1663 1087 3837 6134 6143 3703 1727 2799
8191
```

Рис 3.Контрольный пример.

Далее выводится приглашение пользователю ввести номер сообщения для проверки его подписи, затем выводится результат проверки:

```
Enter number of message to verificate
3
Verification completed succesfully
```

Рис 4.Контрольный пример.

5.Вывод:

В ходе выполнения лабораторной работы был изучен принцип работы дерева Меркле. И реализован алгоритм создания подписи Меркле, для произвольной hash-функции.