Дипломная работа

Криптография на основе функций хэширования: Подписи без состояния

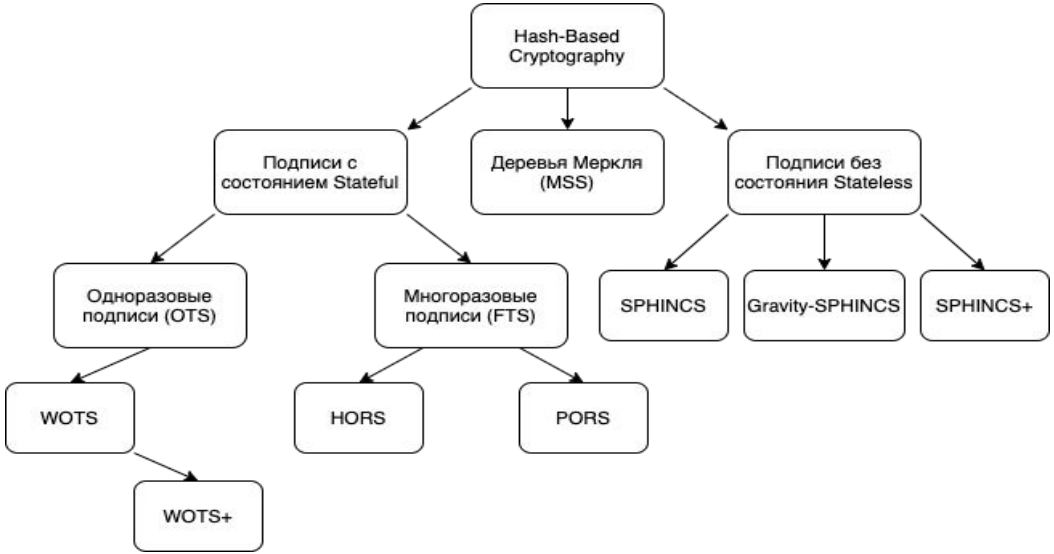
Введение

Криптографический алгоритм	Воздействие квантового компьютера
AES	Нужен больший размер ключа
SHA-2, SHA-3	Нужен больший выход
RSA	Взломано
ECDSA, ECDH (Elliptic Curve Cryptography)	Взломано
DSA (Finite Field Cryptography)	Взломано

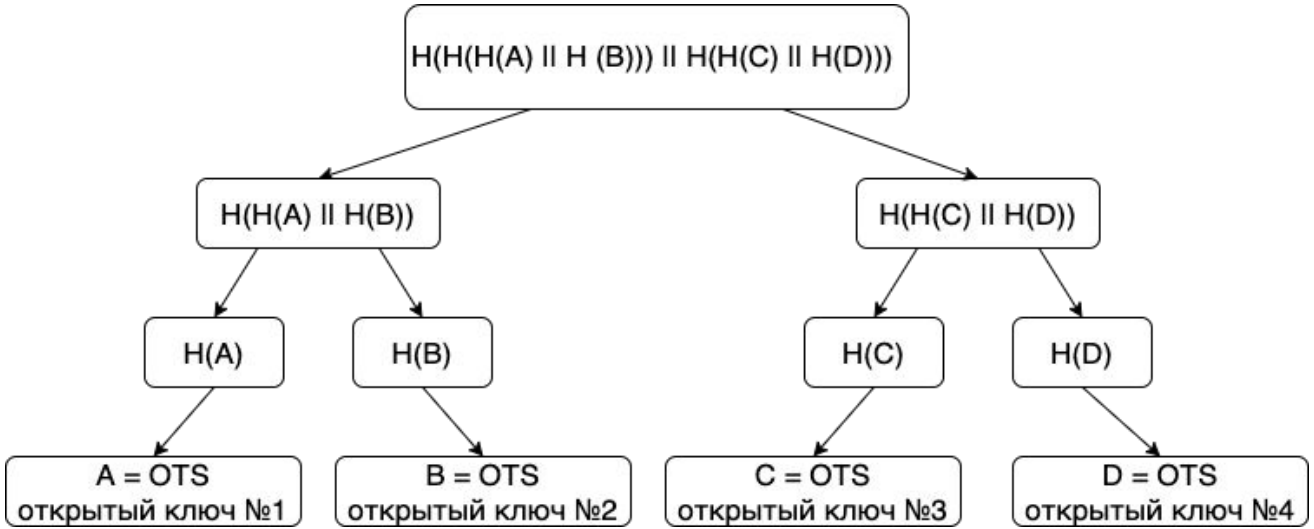
Схема ЭЦП в виде трёх алгоритмов:

1. Алгоритм генерации ключей Gen:
   a. Вход: 1^n, где n — параметр безопасности.
   b. Выход: Открытый ключ pk и личный ключ sk.
2. Алгоритм подписи Sign:
   a. Вход: Сообщение m и личный ключ sk.
   b. Выход: Подпись σ .
3. Алгоритм проверки подписи Verify:
   a. Вход: Открытый ключ pk, сообщение m и подпись σ .
   b. Выход: Проверка успешна (true) или отклонена (false).

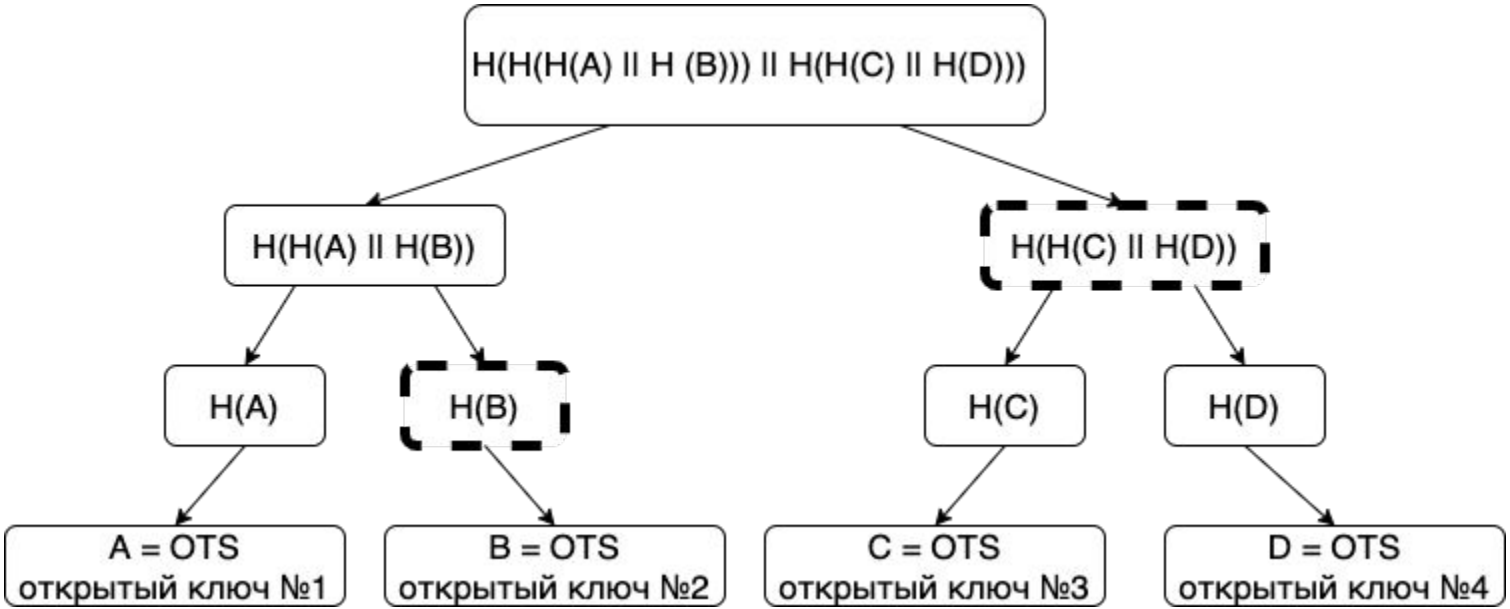
Классификация подписей на основе функций хэширования:



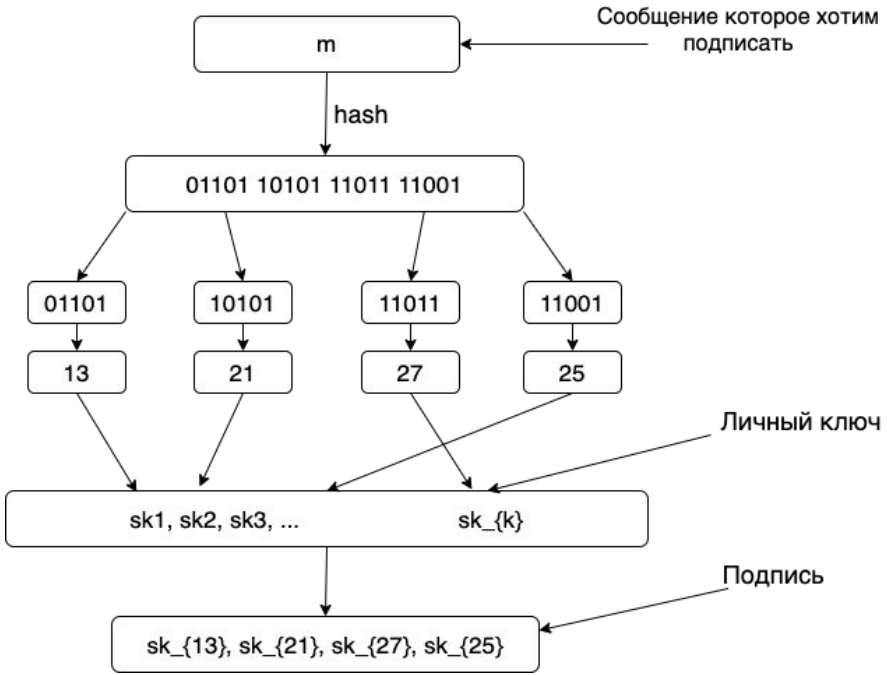
Деревья Меркля



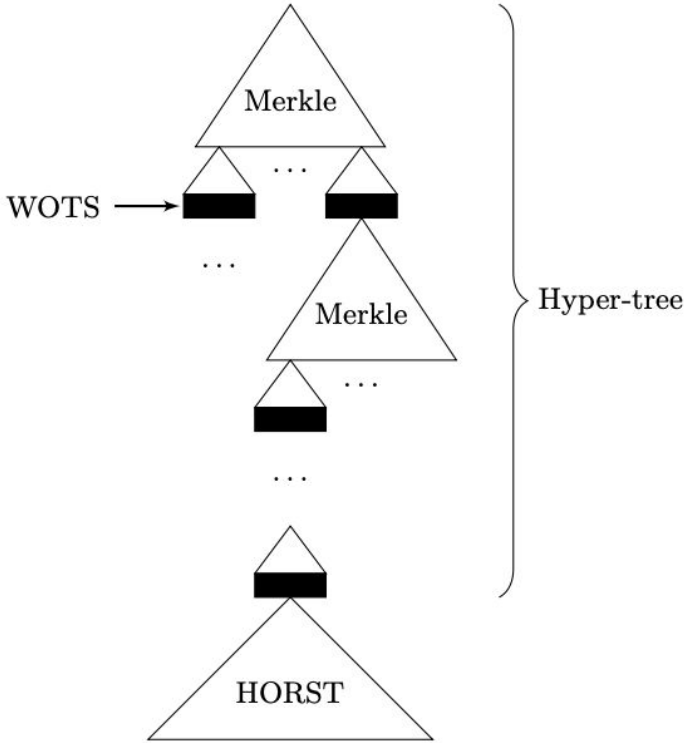
Путь аутентификации дерева Меркля



HORS


m
Сообщение которое хотим подписать
hash
01101 10101 11011 11001
01101
10101
11011
11001
13
21
27
25
Личный ключ
sk1, sk2, sk3, ...
sk_{k}
Подпись
sk_{13}, sk_{21}, sk_{27}, sk_{25}

SPHINCS



Программная реализация

```
15:49:52.202207 1.832s th_a       functor.cpp:161             operator()          ] Push block: head 10, round 11.0, producer 1.2.19, transactions 0, id 0000000b53db568515c798
04abaaf771ed8d428e
15:49:53.300929 1.099s th_a       application.cpp:494         handle_transaction  ] Got 1 transactions from network
15:49:54.046508 745579 th_a       functor.cpp:161             operator()          ] Push block: head 11, round 12.0, producer 1.2.9, transactions 0, id 0000000c81d95320845b30f
81e6f87afa7edfa5f
15:49:54.305080 258572 th_a       application.cpp:494         handle_transaction  ] Got 554 transactions from network
15:49:55.311926 1.007s th_a       application.cpp:494         handle_transaction  ] Got 790 transactions from network
15:49:56.091859 779933 th_a       functor.cpp:161             operator()          ] Push block: head 12, round 13.0, producer 1.2.9, transactions 1376, id 0000000d0249267ef5c5
78ba5efa5a54be87e969
15:49:56.398248 306389 th_a       application.cpp:494         handle_transaction  ] Got 526 transactions from network
15:49:57.401084 1.003s th_a       application.cpp:494         handle_transaction  ] Got 436 transactions from network
15:49:58.358252 957168 th_a       functor.cpp:161             operator()          ] Push block: head 13, round 14.0, producer 1.2.21, transactions 1813, id 0000000e3bb0ceb8ad4
1a88be1063bb3e62971c4
15:49:58.620480 262228 th_a       application.cpp:494         handle_transaction  ] Got 282 transactions from network
15:50:00.457556 1.837s th_a       functor.cpp:161             operator()          ] Push block: head 14, round 15.0, producer 1.2.8, transactions 591, id 0000000f1d3422a88a061
9b9bec52b4b1b6ad7b0
15:50:02.376232 1.919s th_a       functor.cpp:161             operator()          ] Push block: head 15, round 16.0, producer 1.2.15, transactions 220, id 000000108a5ce6144dac
aa9ca03e4be981ca632c
```

Пример работы блокчейна Bitshares

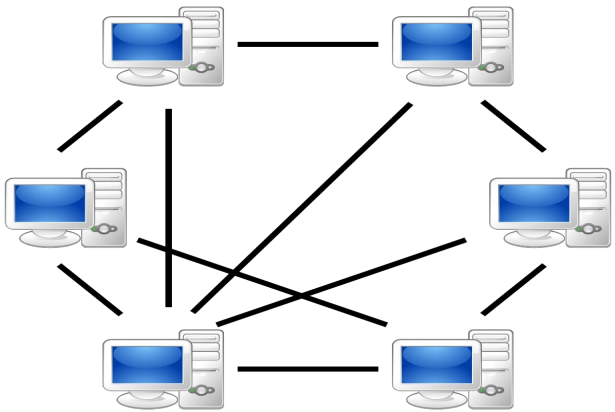
Репозиторий с кодом

Модель транзакций Bitshares

Взаимодействие с Bitshares

1. Blockchain API - используется для запроса блокчейн-данных(счета, активы, торговая история и так далее). Кроме того, данные хранятся в самом блокчейне (блоки, транзакции и так далее), объекты более высокого (например, счета, балансы и так далее) можно получить через полную базу данных узла.
2. Wallet API - отдельный модуль взаимодействия с блокчейном, для удобство разработчиков и тестирование новых операций.

Одноранговый сетевой протокол Bitshares



Peer to peer

Результаты программной реализации

Таблица 1: Сравнение подписей

Algorithm	**Key generation**	**Sign**	**Verify**
SPHINCS-256	12.6 ms	236 ms	2.73 ms
SPHINCS$^+$	11.7 ms	196 ms	2.3 ms
Gravity-SPHINCS	10.3 ms	204 ms	2.4ms
ECDSA(P-256)	0.924 ms	0.553 ms	0.478 ms

Заключение

В дипломной работе получены следующие результаты:

1. Подготовлен анализ публикаций по основам НВС (Hash-Based Cryptography).
2. Изучены современные алгоритмы электронно цифровых подписей на основе функций хэширования.
3. Реализованы ЭЦП на основе функций хэширования на языке Python.
4. Интегрированы в блокчейн архитектуру Bitshares.
5. Составлена таблица сравнения скорости алгоритмов подписей без состояния с подписью на эллиптических кривых ECDSA для подписания транзакции.