CSE 5095-007: Blockchain Technology

Lecture 11
Wallets and Key Management

Ghada Almashaqbeh
UConn - Fall 2022

Outline

- Key management in cryptocurrencies.
  - Hot and cold storage.
  - Hierarchical wallets.
  - Cold info storage.
  - Splitting and sharing keys.
  - Online wallets.

Spending Coins

- Recall that coins in any cryptocurrency are virtual.
  - Strings of bits.
- Spending any amount of coins requires:
  - Some public information from the blockchain.
  - The secret key associated with the address (or public key) that owns the coins.
    - Needed to provide digital signatures.
- Losing the secret keys means losing these coins; no one will be able to spend them.

Key Management

- Storage and retrieval of secret keys.
  - Involves also when and how to generate new keys for future transactions.
- Goals:
  - Security; only the legitimate owner gets to spend a given coin.
  - Availability; coins owners can spend them whenever they wish.
  - Usability; it is relatively easy for the average user to store/retrieve/use secret keys.
- We will focus on Bitcoin in the slides.
  - The concepts we will study can be applied to any other cryptocurrency.

Wallet Software

- User (or client) software that keeps track of coins that a client owns.
- Provides a convenient user interface to simplify operations.
  - Issuing transactions.
  - Tracking total balance.
  - Generate keys when needed.
  - Bookkeeping of these keys.
- Encode addresses as text strings (Base58) or in the form of QR codes.
  - Simplifies sharing addresses with others.
- A large number of wallets is available.
  - Different flavors; desktop, mobile, web applications, etc.
  - Different vendors; metamask, jaxx, coinbase, etc.
  - Security is a driving factor of which one to choose.

Naive Solution

- Store the keys in a file on your laptop or smartphone.
  - Security is tied to your device; breaking into the device allows an attacker to steal your keys (and so your coins).
- This is called hot storage.
  - Easy to use but risky.

Hot vs. Cold Storage I

- Hot storage.
  - Storing secret keys on a device that is connected to the Internet and used frequently for variety of applications.
  - Usable/coventient.
- Cold storage.
  - Offline storage, like on a machine or memory device that is stored in some safe location.
  - Less convenient.
- Good practice: The majority of the coins are in the cold storage with a few in the hot storage.

Hot vs. Cold Storage II

- Seperate keys are needed for each.
  - Otherwise, compromising hot storage will compromise cold storage.
- But both should be aware of their addresses to allow transferring currency.
  - Cold storage stores its cold secret keys, cold addresses, and hot public addresses.
  - Hot storage stores hot secret keys and public addresses, as well as cold public addresses.

New Addresses for Cold Storage?

- A good practice in Bitcoin in order to break transaction linkability is to create a new address for each new transactions.
- How can a hot wallet learn the addresses of a cold wallet?
  - Remember cold wallet is offline, no internet connectivity.
- Even generating a large chunk of addresses at the beginning will not work.
  - Cold storage needs to connect whenever a new batch of addresses is generated.

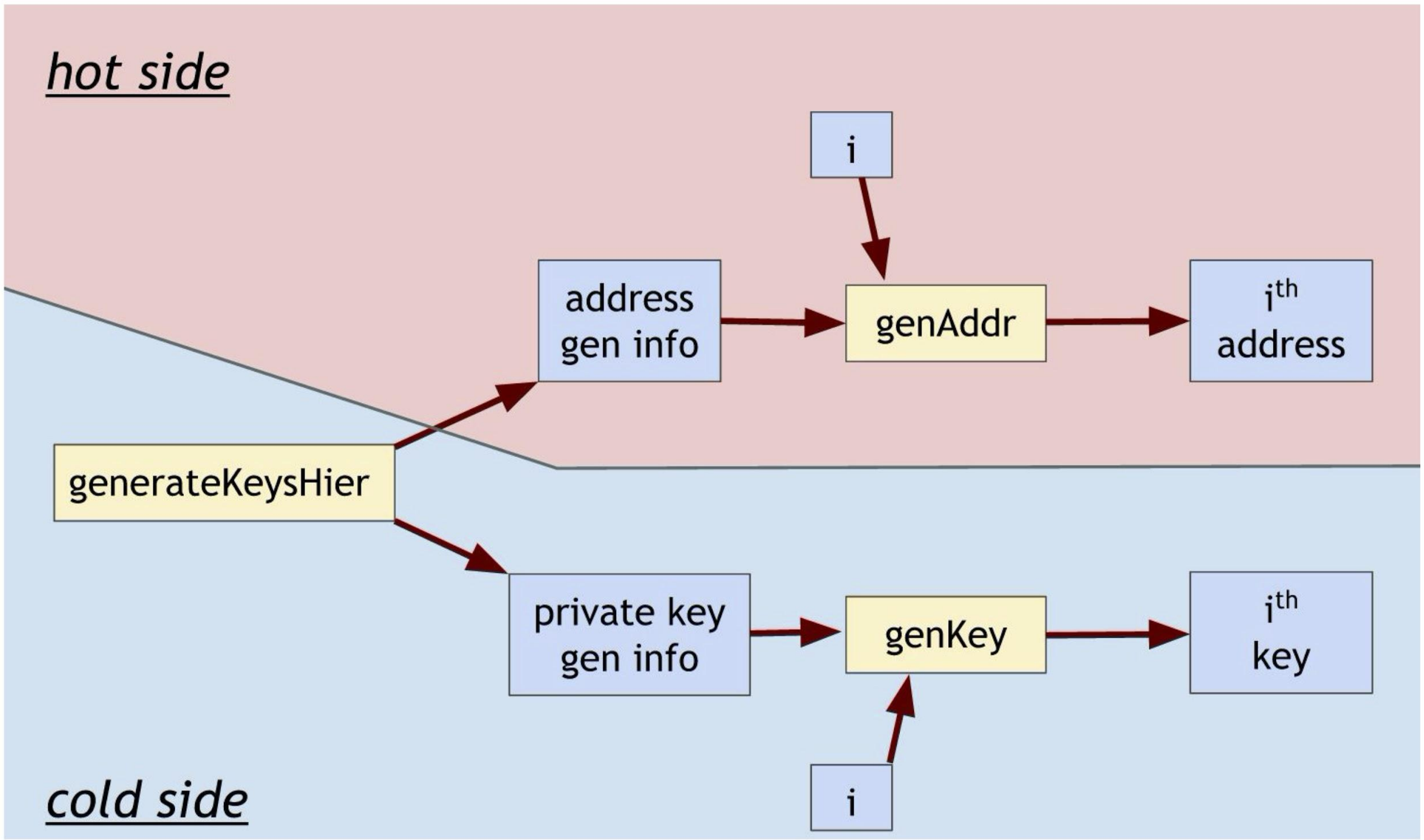
Hierarchical Key Generation I

- Allows cold storage to generate an infinite number of key pairs (or addresses).
- Usual keyGen algorithms generate a single key pair; public and private.
- Hierarchical keyGen instead generates some info that allows generating all future keys in a deterministic way.
  - Usually called master public key and master secret key.
- The master public key does not reveal any info about the master secret key or all future generated secret keys.

Hierarchical Key Generation II

- By having the operation indexed with some integer *i*, the master key allows generating the i^{th} future key.
  - Works for both public and secret keys.
- Hot storage has a copy of the master public key, while the master secret key is known only to the cold storage.
- Not all digital signature schemes support this hierarchical approach.
  - ECDSA supports that, details come later.

Hierarchical KeyGen - Pictorially



*From Chapter 5, Bitcoin and Cryptocurrency Technologies book.

Hierarchical KeyGen in ECDSA (BIP32)

- A group G, in which DDH is believed to be hard, of order p (where p is some large prime number) and a group generator g.
- KeyGen is extended into 3 algorithms:
  - keyGenHer(1^n): msk = x, mpk = g^x, Hash function H.
    - n is the security parameter.
    - x is some integer selected at random from Z_p.
  - addrGen(mpk, i): r = H(i || mpk), pk_i = mpk*g^r = g^{x+r}, $addr_i$= H(pk_i)
  - keyGen(msk, i): r = H(i || mpk), sk_i = msk + r = x+r
- A hot wallet will be able to generate the ith address (or public key) and a cold storage will be able to generate the ith private key.

Security Issue I

- No forward or backward security.
  - Given i, ski, mpk, it is easy to determine msk.
    - Assume an attacker compromised the hot wallet (got hold of mpk), and one secret key of the cold storage has been leaked.
    - This allows the attacker to compute msk, and hence, all previous as well as future secret keys.
    - ***Exercise:*** track the algorithms and see how is msk computed.
- Source of vulnerability; the way the randomness r is computed.

Security Issue II

- BIP32 gives an alternative construction that preserves forward and backward security if a private key is leaked.
  - However, no hierarchical addresses anymore, single address but hierarchical secret keys.
  - Transaction linkability!!
- Guteso et al. [Guteso et al., 2015] developed a bitcoin hierarchical wallet that tolerates leakage of m keys.
  - Drawbacks:
    - m must be fixed in advance.
    - Size of the mpk grows with m.

Cold Info Storage

- On a device, and lock that device in a safe.
- Brain wallets:
  - Encrypt under a password that the user memorizes.
  - Security issue: offline password cracking.
- Paper wallet:
  - Print the key material and store the paper in a safe.
- Tamper resistant hardware device.
  - Either the device generates the secret key and stores it, or just use it to store the private key.
  - The device signs the transactions; it does not allow retrieving the secret key at all.

Splitting and Sharing Keys

- Distribute the trust of storing a key among multiple devices/locations instead of one.
- Works by creating several shares of the key and store each share at a different place.
  - Usually a threshold is used, meaning that having t out of n shares allows constructing the secret key, and hence, use it to sign transactions.
- Good for both availability and security.
  - Any t shares can be used.
  - As long as less than t shares are revealed, no information will be leaked about the secret key.
    - Much better than having a single key stored at a single location.

Shamir Secret Sharing

- A widely used scheme to share secrets.
- A (t, n)-secret sharing scheme consists of two algorithms:
  - Share(s): Outputs shares s1, s2, …, sn.
  - Reconstruct(x1, …, xt): Outputs s.
    - x1, …, xt: any t subset of the shares s1, …, sn
- t-privacy:
  - For any two secrets s, s', and any subset X of size < t, the shares of s are indistinguishable from the shares of s'.
- The basic idea of Shamir Secret Sharing:
  - Generate a polynomial of degree (t-1) with the free factor set as the secret key.
  - Each share is simply the evaluation of the polynomial at the share index.
  - Reconstruct is computing the free factor using lagrange interpolation.

Example - (2, 2) Secret Sharing

- $n = 2$ and $t = 2$.
- A prime $p = 11$, so we work in Zp (the integers 0, …, 10).
- To share a secret s, first choose some r at random from Zp
- Share(s):
  - $s_i = (s + i*r) \bmod p$ for i in {1, 2}
- Reconstruct(s1, s2):
  - $s = (2x_1 - x_2) \bmod p$
- 2-privacy: each si is uniformly distributed over Zp.

Mulitsig and Threshold Signatures

- Multisig:
  - No need to reconstruct the key, keep the random shares apart.
    - Or simply generate multiple keys instead of one.
  - To authorize a transaction, all devices (or t of them) need to sign.
  - If collective signing is used, one signature will be produced.
- Threshold signatures:
  - A signing key is divided among several parties such that any t subset of them can jointly produce a signature.
  - Produce a single signature instead of many.
    - More efficient, signature verification algorithm will be executed once.
  - There is an increasing interest of threshold signatures these days across a large number of cryptocurrency systems.
    - Among them, the BLS scheme is a popular one (check https://crypto.stanford.edu/~dabo/pubs/papers/BLSmultisig.html).

Online Wallets

- A wallet in the form of a web app.
  - The site stores keys.
  - The site issues transactions when the users asks for that.
  - The user logs in using some credentials.
- Great usability; log in using any device connected to the Internet.
- Site compromised, wallet is compromised.
- Usually used when trading on exchanges.

Cryptocurrency Exchanges

- Pretty much a centrally managed bank system.
    - Accept deposits in cryptocurrency or fiat currency.
        - With a promise to pay back when a client ask for any withdrawal.
    - Allow customers to pay in cryptocurrency, receive payments, and trade cryptocurrencies.
        - Mainly match buyers with sellers at the exchange rate along with charging fees.
    - Nothing goes to the network, it is just that the exchange makes a different promise to the customers.
        - Everything is local.

Cryptocurrency Exchanges - Issues

- Highly convenient.
  - Several services at the same place.
  - Very close to the traditional banking systems already in use.
  - Gives a direct value for cryptocurrency in terms of fiat currency.
- High risk.
  - Requires pre-identification - know your customer policy.
    - Any anonymity aspect promised by a cryptocurrency is taken away!
  - Security risks - a target for attackers.
    - Exchanges accumulate currency of all customers.
- Regulation issues.
  - Traditional banks has to prove holding a specific fraction of money in reserve.

References

- [Guteso et al., 2015] Gutoski, Gus, and Douglas Stebila. "Hierarchical deterministic bitcoin wallets that tolerate key leakage." In International Conference on Financial Cryptography and Data Security, pp. 497-504. Springer, Berlin, Heidelberg, 2015.