Blockchain and Cryptocurrencies

Week 5 — Chapter 4: Storing and Using Bitcoin

Prof. Dr. Peter Thiemann

Albert-Ludwigs-Universität Freiburg, Germany

SS 2020

Storing and Using Bitcoin

draws on material from

- Bitcoin and Cryptocurrency Technologies
- Bitcoin, Blockchain, and Cryptoassets
- Antonopoulos: Mastering Bitcoin

Contents

1. Transmitting Addresses

2. Storing Bitcoins
   - Wallets
   - Hot and Cold Storage

3. Splitting and Sharing Keys

4. Online Wallets and Exchanges

How to Transmit Addresses

- Bitcoin addresses correspond to public keys
- public key raw form (pk): 256 bits / 32 bytes (64 hex digits)
- public key hash: 160 bits / 20 bytes (40 hex digits)
  pkh = RIPEMD160 (SHA-256 (pk))
- textual transmission format: 34 bytes (printable ASCII characters)
  Base58Check (0x00 ∥ pkh)

Base58 Encoding

- textual encoding of binary data
- uses 58 ASCII characters for encoding in base 58
- all digits, uppercase, lowercase characters, except 0OIl
- more efficient than hexadecimal encoding (16 characters for encoding)
- compared to base64 encoding, base58 eliminates some characters that can create confusion

Reminder: Base64 encoding

- uses 64 printable ASCII characters to represent 6 bits in a byte
- used to transmit binary datat in emails

Base58Check Encoding

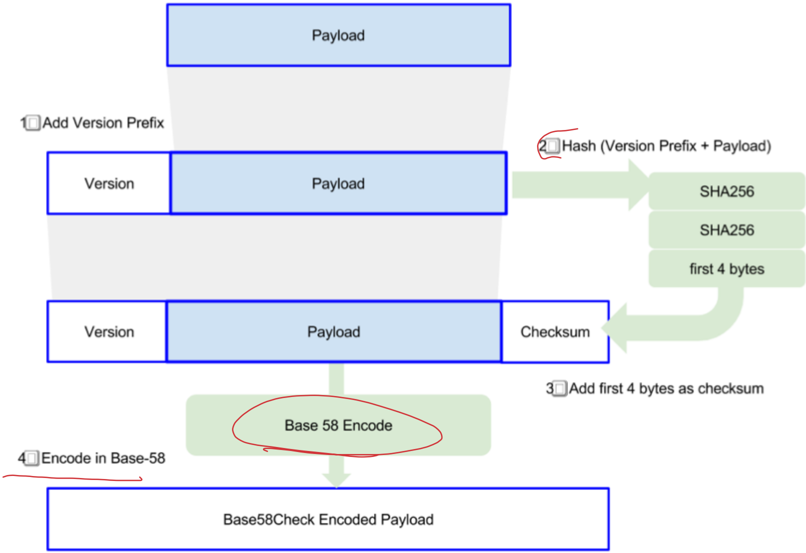


Figure 4-6 from Antonopoulos, Mastering Bitcoin

Version Prefix

In Bitcoin the version prefix byte denotes the type of encoded data:

Type	Prefix	Prefix in Base58
Bitcoin Address	0x00	1
P2SH Address	0x05	3
Private Key WIF	0x80	5, K, or L
	⋮	

Store private key

- WIF: wallet interchange format
- there are also compressed formats for private and public keys

Non-electronic Transmission

- Consider an address like `1thMirt546nngXqyPEz532S8fLwbozud8`

generated by `https://www.bitcoinqrcodemaker.com/api/?style=bitcoin&amp;address=1thMirt546nngXqyPEz532S8fLwbozud8`

Non-electronic Transmission

- Consider an address like `1thMirt546nngXqyPEz532S8fLwbozud8`
- instead of typing it in, you could supply a QR code

generated by https://www.bitcoinqrcodemaker.com/api/?style=bitcoin&amp;address=1thMirt546nngXqyPEz532S8fLwbozud8

Non-electronic Transmission

- Consider an address like 1thMirt546nngXqyPEz532S8fLwbozud8
- instead of typing it in, you could supply a QR code

generated by `https://www.bitcoinqrcodemaker.com/api/?style=bitcoin&amp;address=1thMirt546nngXqyPEz532S8fLwbozud8`

Contents

1. Transmitting Addresses

2. Storing Bitcoins
   - Wallets
   - Hot and Cold Storage

3. Splitting and Sharing Keys

4. Online Wallets and Exchanges

Storing Bitcoins

What do we need to transact with bitcoins?

Storing Bitcoins

What do we need to transact with bitcoins?

- public information

Storing Bitcoins

What do we need to transact with bitcoins?

- public information
  - to send: addresses of payment receivers

Storing Bitcoins

What do we need to transact with bitcoins?

- public information
  - to send: addresses of payment receivers
  - to receive: own address(es)

Storing Bitcoins

What do we need to transact with bitcoins?

- public information
  - to send: addresses of payment receivers
  - to receive: own address(es)
- secret information

Storing Bitcoins

What do we need to transact with bitcoins?

- public information
  - to send: addresses of payment receivers
  - to receive: own address(es)
- secret information
  - to send: secret key(s) to sign off on inputs

Storing Bitcoins

What do we need to transact with bitcoins?

- public information
  - to send: addresses of payment receivers
  - to receive: own address(es)
- secret information
  - to send: secret key(s) to sign off on inputs
  - to receive: script(s) matching the hashes of P2SH addresses

Storing Bitcoins

What do we need to transact with bitcoins?

- public information
  - to send: addresses of payment receivers
  - to receive: own address(es)
- secret information
  - to send: secret key(s) to sign off on inputs
  - to receive: script(s) matching the hashes of P2SH addresses

Key Issues

Storing, managing, and safeguarding bitcoin secret keys and scripts

Aspects of Storing Bitcoins

Availability

Aspects of Storing Bitcoins

Availability

Security

Aspects of Storing Bitcoins

Availability

Security

Convenience

Aspects of Storing Bitcoins

Availability

Security

Convenience

- Each kind of storage strikes some balance
- Hard to achieve all three

Local Storage

- the simplest way
- files on your local computer / device

Availability:	o
Security	–
Convenience	++

Contents

1 Transmitting Addresses

2 Storing Bitcoins
- Wallets
- Hot and Cold Storage

3 Splitting and Sharing Keys

4 Online Wallets and Exchanges

Wallet Software (local)

1. key management (creation, import, storing)
2. management of UTXOs
3. management of change addresses
4. management of receiver addresses (import, storing)
5. composing transactions w correct locking / unlocking scripts

Random Wallets

Type-0 Non-Deterministic Wallets

- also: JBOK (Just a bunch of keys)
- pre-generates 100 random private keys
- generates more as needed

Random Wallets

Type-0 Non-Deterministic Wallets

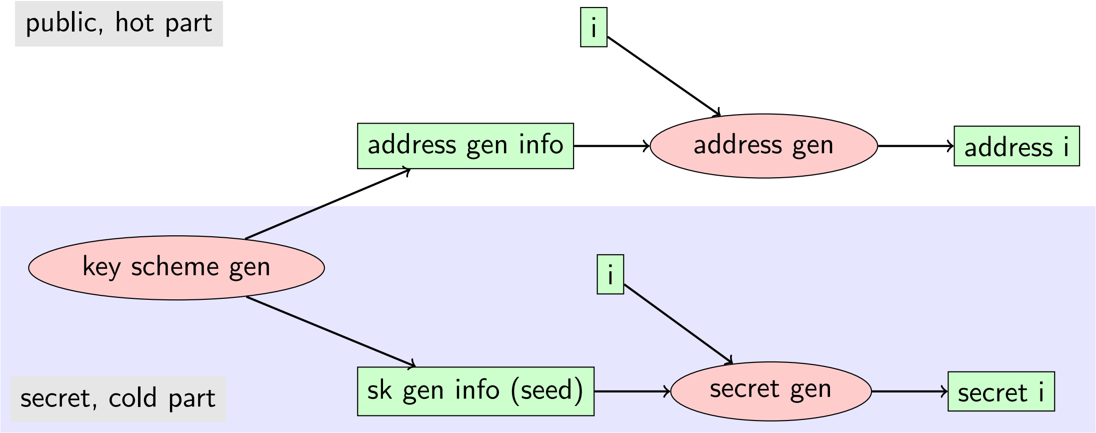
- also: JBOK (Just a bunch of keys)
- pre-generates 100 random private keys
- generates more as needed

Disadvantages

- each single key must be remembered
- frequent backup required

Deterministic Wallets

- starting from a single random seed, addresses are generated algorithmically
- only the seed needs to be remembered (and kept secret)



A Simple Deterministic Wallet

Illustration of principle using RSA

Foundation

Given an RSA key pair (e, d) and $n = pq$

- $\gcd(e, (p-1)(q-1)) = 1$ implies $\gcd(e^i, (p-1)(q-1)) = 1$ for all i
- moreover, $ed \equiv 1 \mod (p-1)(q-1)$ implies $e^i d^i \equiv 1 \mod (p-1)(q-1)$
- hence, (e^i, d^i) is also a key pair (with power computed $\mod (p-1)(q-1)$)

A Simple Deterministic Wallet

Illustration of principle using RSA

Foundation

Given an RSA key pair (e, d) and $n = pq$

- $\gcd(e, (p-1)(q-1)) = 1$ implies $\gcd(e^i, (p-1)(q-1)) = 1$ for all i
- moreover, $ed \equiv 1 \mod (p-1)(q-1)$ implies $e^i d^i \equiv 1 \mod (p-1)(q-1)$
- hence, (e^i, d^i) is also a key pair (with power computed $\mod (p-1)(q-1)$)

Key generation scheme

- Generate an RSA key pair (e, d) for some $n = pq$
- Pick a random number k and a hash function H
- The ith secret key is $e^{H(k\|i)} \mod (p-1)(q-1)$
- Public address generation info: d and k
- The ith public key is $d^{H(k\|i)}$

A Simple Deterministic Wallet

Illustration of principle using RSA

Foundation

Given an RSA key pair (e, d) and $n = pq$

- $\gcd(e, (p-1)(q-1)) = 1$ implies $\gcd(e^i, (p-1)(q-1)) = 1$ for all i
- moreover, $ed \equiv 1 \mod (p-1)(q-1)$ implies $e^i d^i \equiv 1 \mod (p-1)(q-1)$
- hence, (e^i, d^i) is also a key pair (with power computed $\mod (p-1)(q-1)$)

Key generation scheme

- Generate an RSA key pair (e, d) for some $n = pq$
- Pick a random number k and a hash function H
- The ith secret key is $e^{H(k\|i)} \mod (p-1)(q-1)$
- Public address generation info: d and k
- The ith public key is $d^{H(k\|i)}$

NOT practical — do not use

The *i*th public key generated by this scheme is much too big.

Hierarchical Deterministic Wallets

The real thing

- tree structure to express additional organizational meaning, e.g.
    - one branch for incoming payments
    - another branch for change from outgoing payments
- insecure servers can generate public keys without having access to private keys

Hierarchical Deterministic Wallets

The real thing

- tree structure to express additional organizational meaning, e.g.
  - one branch for incoming payments
  - another branch for change from outgoing payments
- insecure servers can generate public keys without having access to private keys

Child Key Derivation

Based on one-way hash functions that combine

- A parent private or public key
- A seed called a chain code (256 bits)
- An index number (32 bits)

"The chain code is used to introduce seemingly random data to the process, so that the index is not sufficient to derive other child keys. Thus, having a child key does not make it possible to find its siblings, unless you also have the chain code. The initial chain code seed (at the root of the tree) is made from random data, while subsequent chain codes are derived from each parent chain code." [Antonopoulos, Mastering Bitcoin, p90]

Contents

1 Transmitting Addresses

2 Storing Bitcoins
- Wallets
- Hot and Cold Storage

3 Splitting and Sharing Keys

4 Online Wallets and Exchanges

Hot and Cold Storage

Hot Storage

- private keys are stored in wallet software that is connected
- full availability (sending and receiving payments) and convenience
- but subject to the usual security concerns

Hot and Cold Storage

Hot Storage

- private keys are stored in wallet software that is connected
- full availability (sending and receiving payments) and convenience
- but subject to the usual security concerns

Cold Storage

- private keys stored in a safe place offline
- receiving payments is still possible
- access UTXOs we first need to import private key

Hot and Cold Storage

Hot Storage

- private keys are stored in wallet software that is connected
- full availability (sending and receiving payments) and convenience
- but subject to the usual security concerns

Cold Storage

- private keys stored in a safe place offline
- receiving payments is still possible
- access UTXOs we first need to import private key

Problem

- Transfer from Hot to Cold should not compromise anonymity
- Use a fresh address for each transfer, but where does it come from?

Managing Cold Storage

- deterministic seeded wallet
- seed could be stored (even remembered) offline

⇒ mnemonic code words

- represent a number (seed) using a sequence of words from a fixed list
- serves as wallet backup

Mnemonic Code Words

Seed generation (BIP0039)

1. Create random number S of 128 bits
2. Checksum $=$ first four bits of SHA256(S)
3. Consider the bit string $S \| \mathrm{Checksum}$
4. Divide into 12 sections of 11 bits serving as index into 2048 word dictionary
5. Return the resulting 12 words

Example

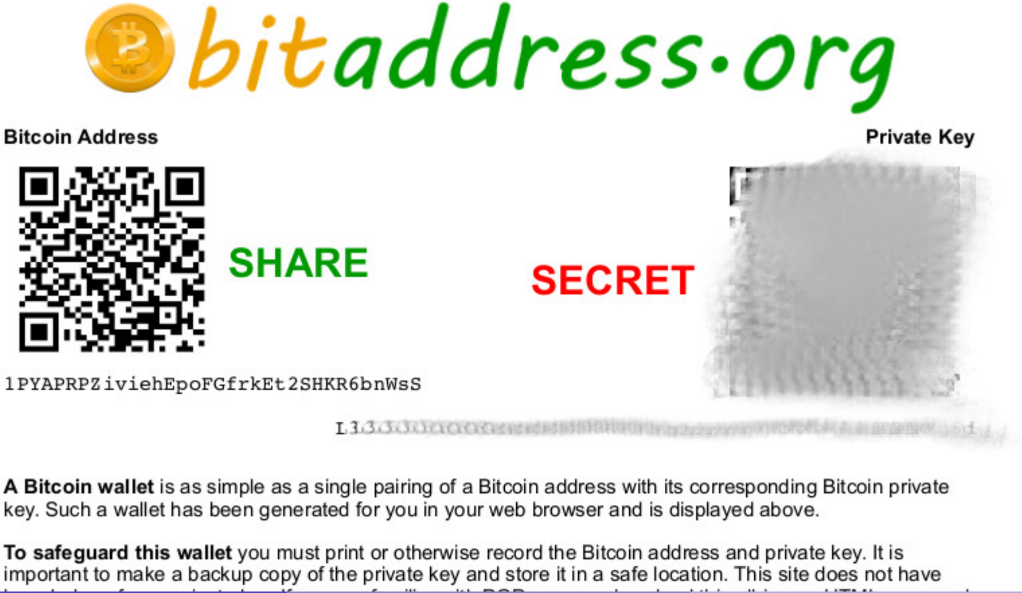
Random number S `0c1e24e5917779d297e14d45f14e1a1a`

List of words army van defense carry jealous true garbage claim echo media make crunch

Paper Wallets

- Bitcoin private keys printed on paper
- Secured against hackers, if generated offline and never stored on a computer (before the first and only use)
- Can be generated by tools like https://www.bitaddress.org/

Example Paper Wallet



1PYAPRPZiviehEpoFGfrkEt2SHKR6bnWsS

[redacted]

A Bitcoin wallet is as simple as a single pairing of a Bitcoin address with its corresponding Bitcoin private key. Such a wallet has been generated for you in your web browser and is displayed above.

To safeguard this wallet you must print or otherwise record the Bitcoin address and private key. It is important to make a backup copy of the private key and store it in a safe location. This site does not have

Tamper-Resistant Devices

- External device that stores private key
- It may even generate the private key and perform the signature externally

⇒ the key never leaves the device

- The only way to steal the key is stealing the device

Example

Contents

1 Transmitting Addresses

2 Storing Bitcoins
- Wallets
- Hot and Cold Storage

3 Splitting and Sharing Keys

4 Online Wallets and Exchanges

Splitting and Sharing Keys

- Another approach to secure storing of a secret key
- Divide the key into several parts and store them separately
- No single part is sufficient to reconstruct the key
- But from any $n > 1$ parts, it is possible to reconstruct the key

Example for Key Reconstruction

Reconstruct key from any $n = 2$ fragments

Generate Fragments

- Suppose s is a secret key with $0 < s < p$ where p is prime.
- Choose a random number m with $0 < m < p$.
- Generate key fragments $(i, s + mi \mod p)$ for any $0 < i < p$

Example for Key Reconstruction

Reconstruct key from any $n = 2$ fragments

Generate Fragments

- Suppose s is a secret key with $0 < s < p$ where p is prime.
- Choose a random number m with $0 < m < p$.
- Generate key fragments $(i, s + mi \mod p)$ for any $0 < i < p$

Reconstruct Key (all calculations are $\mod p$!)

Example for Key Reconstruction

Reconstruct key from any $n = 2$ fragments

Generate Fragments

- Suppose s is a secret key with $0 < s < p$ where p is prime.
- Choose a random number m with $0 < m < p$.
- Generate key fragments $(i, s + mi \mod p)$ for any $0 < i < p$

Reconstruct Key (all calculations are $\mod p$!)

- Given any two different fragments (i, a) and (j, b) calculate $m' = \frac{b-a}{j-i}$

Example for Key Reconstruction

Reconstruct key from any $n = 2$ fragments

Generate Fragments

- Suppose s is a secret key with $0 < s < p$ where p is prime.
- Choose a random number m with $0 < m < p$.
- Generate key fragments $(i, s + mi \mod p)$ for any $0 < i < p$

Reconstruct Key (all calculations are $\mod p$!)

- Given any two different fragments (i, a) and (j, b) calculate $m' = \frac{b-a}{j-i}$
- Recall that $a = s + mi \mod p$ and $b = s + mj \mod p$

Example for Key Reconstruction

Reconstruct key from any $n = 2$ fragments

Generate Fragments

- Suppose s is a secret key with $0 < s < p$ where p is prime.
- Choose a random number m with $0 < m < p$.
- Generate key fragments $(i, s + mi \mod p)$ for any $0 < i < p$

Reconstruct Key (all calculations are $\mod p$!)

- Given any two different fragments (i, a) and (j, b) calculate $m' = \frac{b-a}{j-i}$
- Recall that $a = s + mi \mod p$ and $b = s + mj \mod p$
- Hence $b - a = s + mj - (s + mi) = m(j - i) \mod p$

Example for Key Reconstruction

Reconstruct key from any $n = 2$ fragments

Generate Fragments

- Suppose s is a secret key with $0 < s < p$ where p is prime.
- Choose a random number m with $0 < m < p$.
- Generate key fragments $(i, s + mi \mod p)$ for any $0 < i < p$

Reconstruct Key (all calculations are $\mod p$!)

- Given any two different fragments (i, a) and (j, b) calculate $m' = \frac{b-a}{j-i}$
- Recall that $a = s + mi \mod p$ and $b = s + mj \mod p$
- Hence $b - a = s + mj - (s + mi) = m(j - i) \mod p$
- Hence $m' = \frac{b-a}{j-i} = \frac{m(j-i)}{j-i} = m \mod p$

Example for Key Reconstruction

Reconstruct key from any $n = 2$ fragments

Generate Fragments

- Suppose s is a secret key with $0 < s < p$ where p is prime.
- Choose a random number m with $0 < m < p$.
- Generate key fragments $(i, s + mi \mod p)$ for any $0 < i < p$

Reconstruct Key (all calculations are $\mod p$!)

- Given any two different fragments (i, a) and (j, b) calculate $m' = \frac{b-a}{j-i}$
- Recall that $a = s + mi \mod p$ and $b = s + mj \mod p$
- Hence $b - a = s + mj - (s + mi) = m(j - i) \mod p$
- Hence $m' = \frac{b-a}{j-i} = \frac{m(j-i)}{j-i} = m \mod p$
- Taken together $a - m'i = (s + mi) - mi = s \mod p$, the secret key

Example for Key Reconstruction

Reconstruct key from any $n = 2$ fragments

Generate Fragments

- Suppose s is a secret key with $0 < s < p$ where p is prime.
- Choose a random number m with $0 < m < p$.
- Generate key fragments $(i, s + mi \mod p)$ for any $0 < i < p$

Reconstruct Key (all calculations are $\mod p$!)

- Given any two different fragments (i, a) and (j, b) calculate $m' = \frac{b-a}{j-i}$
- Recall that $a = s + mi \mod p$ and $b = s + mj \mod p$
- Hence $b - a = s + mj - (s + mi) = m(j - i) \mod p$
- Hence $m' = \frac{b-a}{j-i} = \frac{m(j-i)}{j-i} = m \mod p$
- Taken together $a - m'i = (s + mi) - mi = s \mod p$, the secret key

Remark

Works because Z_p (the set $0, 1, \ldots, p - 1$ with addition and multiplication mod p) is a field!

Illustration for Key Reconstruction From $n = 2$ Fragments

Generalization to $n = 3, 4, \ldots$

Generate Fragments

- Given n choose random numbers $m_1, \ldots, m_{n-1}$ with $0 < m_i < p$
- Generate key fragments $(i, s + m_1 i + m_2 i^2 + \cdots + m_{n-1} i^{n-1})$

Generalization to $n = 3, 4, \ldots$

Generate Fragments

- Given n choose random numbers $m_1, \ldots, m_{n-1}$ with $0 < m_i < p$
- Generate key fragments $(i, s + m_1 i + m_2 i^2 + \cdots + m_{n-1} i^{n-1})$

Reconstruct Key

- All key fragments are points on a polynomial function of degree $n - 1$ over field Z_p
- Theorem: Such a polynomial is uniquely determined by n points
- Given n points (key fragments), we can recover the coefficients m_i and s by Lagrange interpolation

Avoiding Reconstruction

Remaining Issue

- To sign, the key must be present

Avoiding Reconstruction

Remaining Issue

- To sign, the key must be present

Threshold Cryptography

- can perform partial signature with a key fragment
- signature complete once a sufficient number of partial signatures have been applied

Avoiding Reconstruction

Remaining Issue

- To sign, the key must be present

Threshold Cryptography

- can perform partial signature with a key fragment
- signature complete once a sufficient number of partial signatures have been applied

Another Alternative

- multi signatures
- less convenient to use than P2PKH

Contents

1. Transmitting Addresses

2. Storing Bitcoins
   - Wallets
   - Hot and Cold Storage

3. Splitting and Sharing Keys

4. Online Wallets and Exchanges

Online Wallets

Wallet in the Browser

- convenient (simple installation, multiple devices)
- but your secret keys are stored with the service provider

Exchanges

Exchanges

- An exchange is similar to a bank
- For each customer, it keeps accounts in several different (crypto-) currencies
- It offers a trading service like this
  - Alice offers 3 BTC for 35000 USD
  - Bob accepts this offer
  - The exchange atomically (usually for a fee)
    - checks that Alice has 3 BTC and that Bob has 35000 USD
    - adjusts the account balances of Alice and Bob accordingly
- This trade does **not** materialize as a Bitcoin transaction!
- Neither Alice nor Bob (nor the exchange which might only accept deposits in USD!) need to own Bitcoin addresses
- Only if they want to withdraw from the exchange, they need to create their own wallet and have the exchange pay their account balance

Exchanges II

Convenience

connection between fiat currencies and crypto currencies

Exchanges II

Convenience

connection between fiat currencies and crypto currencies

Risks

Bank run too many customers demands their money at the same time

- exchange may no longer be able comply because they have only fractional reserves (like a bank)
- more customers become worried and demand their money

Ponzi schemers may run the exchange

- payouts are performed using the deposits of new customers
- with a large fraction of funds disappearing

Hacker attack if the exchange is honest, it controls bitcoin addresses with large amounts

Exchanges II

Convenience

connection between fiat currencies and crypto currencies

Risks

Bank run too many customers demands their money at the same time

- exchange may no longer be able comply because they have only fractional reserves (like a bank)
- more customers become worried and demand their money

Ponzi schemers may run the exchange

- payouts are performed using the deposits of new customers
- with a large fraction of funds disappearing

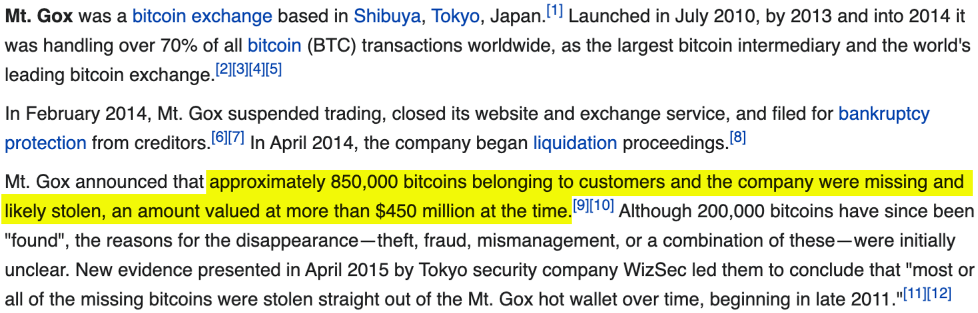
Hacker attack if the exchange is honest, it controls bitcoin addresses with large amounts

All the above have happened!

most famous example: Mt. Gox

Mt. Gox

from Wikipedia



Mt. Gox was a bitcoin exchange based in Shibuya, Tokyo, Japan.[1] Launched in July 2010, by 2013 and into 2014 it was handling over 70% of all bitcoin (BTC) transactions worldwide, as the largest bitcoin intermediary and the world's leading bitcoin exchange.[2][3][4][5]

In February 2014, Mt. Gox suspended trading, closed its website and exchange service, and filed for bankruptcy protection from creditors.[6][7] In April 2014, the company began liquidation proceedings.[8]

Mt. Gox announced that approximately 850,000 bitcoins belonging to customers and the company were missing and likely stolen, an amount valued at more than $450 million at the time.[9][10] Although 200,000 bitcoins have since been "found", the reasons for the disappearance—theft, fraud, mismanagement, or a combination of these—were initially unclear. New evidence presented in April 2015 by Tokyo security company WizSec led them to conclude that "most or all of the missing bitcoins were stolen straight out of the Mt. Gox hot wallet over time, beginning in late 2011."[11][12]

Thanks!