CBS COPENHAGEN BUSINESS SCHOOL
HANDELSHØJSKOLEN

Elective Course on Mastering Blockchain: Foundations to Consensus, session-04

Storing & Spending Bitcoins, Wallets

Raghava Mukkamala

Associate Professor & Director, Centre for Business Data Analytics

Copenhagen Business School, Denmark

Email: rrm.digi@cbs.dk, Centre: https://cbsbda.github.io/

Course Coordinator at SRMIST:

Prof. K. Shantha Kumari

Associate Professor

Data Science and Business Systems Department,

SRM Institute of Science and Technology, India

Shanthak@srmist.edu.in

EQUIS ACCREDITED AACSB ACCREDITED ASSOCIATION AMBA ACCREDITED CEMS PIM

Outline

- How to Store and Use Cryptocurrencies
    - Hot and Cold Storage
- Wallets, deterministic and non-deterministic keys
- Mnemonic Codes and Generation
- Hierarchical Key Generation
- **Optional/Extra Slides**
    - Extended/Hardened Hierarchical Key Generation
    - Splitting and Sharing Keys

Crypto Currencies - Simple Local Storage

Slides based on Bitcoin and Cryptocurrency Technologies: http://bitcoinbook.cs.princeton.edu/

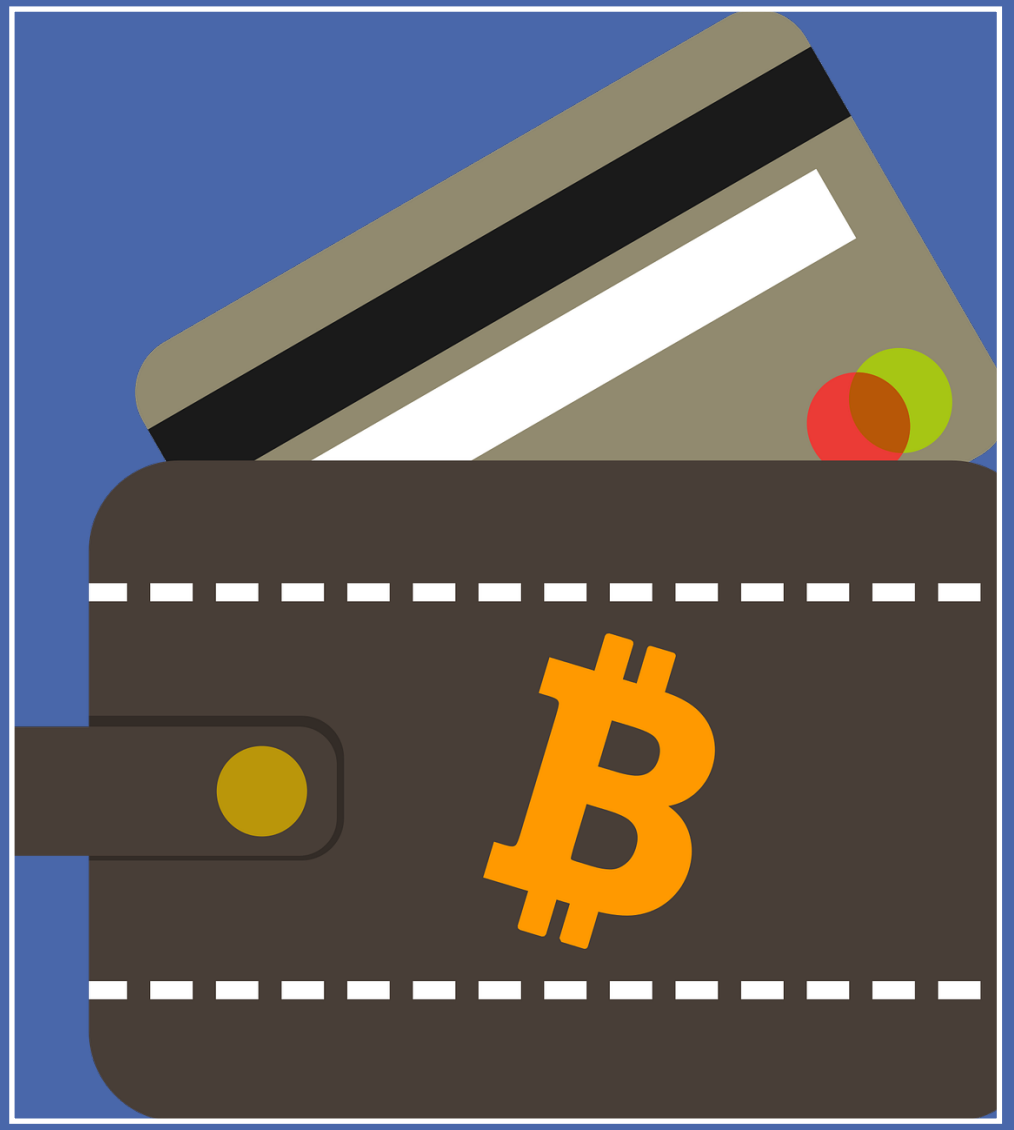


Image by PixLoger from Pixabay

CBS

Spending Bitcoins

To spend a Bitcoin, you need to know:

* some info from the public blockchain, and
* the owner's secret signing key

So it's all about key management.

Storing bitcoins is really all about storing and managing Bitcoin secret keys.

Arvind Narayanan et. al., (2016). Bitcoin and Cryptocurrency Technologies. p.102

How to Store and Use Bitcoins / Secret keys

Slides based on Bitcoin and Cryptocurrency Technologies: http://bitcoinbook.cs.princeton.edu/

Designed by studiogstock / Freepik

CBS

Goals

availability: You can spend your coins.

security: Nobody else can spend your coins.

convenience

> Different approaches to key management offer different trade-offs between availability, security, and convenience.

Arvind Narayanan et. al., (2016). Bitcoin and Cryptocurrency Technologies. p.102

- Simplest approach: store key in a file, on your computer or phone => Very convenient.
- As available as your device {device lost/w

  coins lost}
- As secure as your device {device compromi

  ⇒ coins stolen}

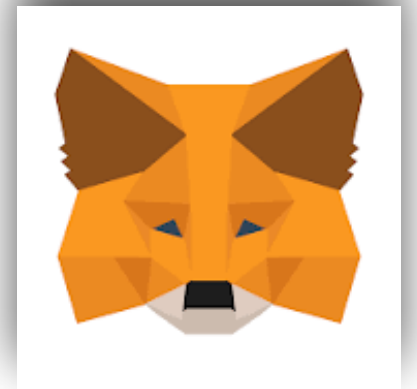
- storing your private keys on a local device, especially a mobile device, is a lot like carrying around money in your wallet/purse.
- It's useful to have some spending money, but you don't want to carry around your life savings because you might lose it, or somebody might steal it.

Arvind Narayanan et. al., (2016). Bitcoin and Cryptocurrency Technologies. p.102

Wallets

- If you're storing your bitcoins locally, you'd typically use wallet software, which is software that keeps track of all your coins, manages all the details of your keys, and makes things convenient with a nice user interface.
- If you want to send $4.25 worth of bitcoins to your local coffee shop, the wallet software will give you an easy way to do that.
- Wallet software is especially useful because you typically want to use a whole bunch of different addresses with different keys associated with them.
- As you may remember, creating a new public/private key pair is easy, and you can utilize this to improve your anonymity or privacy.
- Wallet software gives you a simple interface that tells you how much is in your wallet. When you want to spend bitcoins, it handles the details of which keys to use and how to generate new addresses and so on.

https://www.applause.com/blog/a-quick-guide-to-crypto-wallet-development



Metamask Crypto wallet

Arvind Narayanan et. al., (2016). Bitcoin and Cryptocurrency Technologies. p.102

Wallet software

Keeps track of your coins, provides nice user interface.

Nice trick: use a separate address/key for each coin.

benefits privacy (looks like separate owners)

wallet can do the bookkeeping, user needn’t know

Arvind Narayanan et. al., (2016). Bitcoin and Cryptocurrency Technologies. p.102

Encoding addresses

Encode as text string: base58 notation

123456789ABCDEFGHJKLMNPQRSTUVWXYZabcdefghijkmnopqrstuvwxyz

or use QR code

Arvind Narayanan et. al., (2016). Bitcoin and Cryptocurrency Technologies. p.102

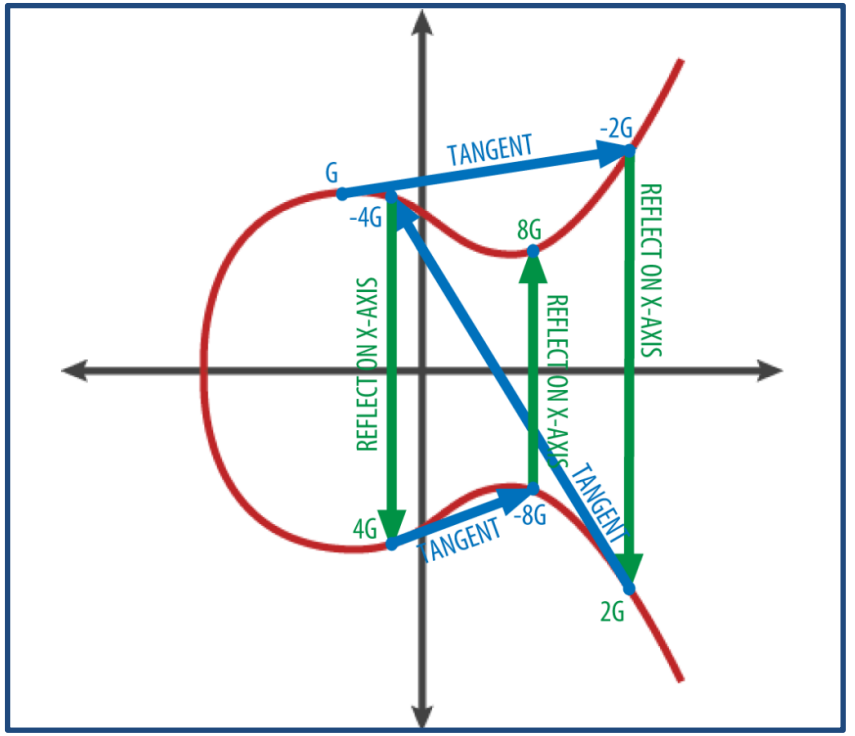
Public Key in Bitcoin

Generating a Public Key

Starting with a private key in the form of a randomly generated number k, we multiply it by a predetermined point on the curve called the *generator point* G to produce another point somewhere else on the curve, which is the corresponding public key K. The generator point is specified as part of the `secp256k1` standard and is always the same for all keys in bitcoin:

$$K = k * G$$

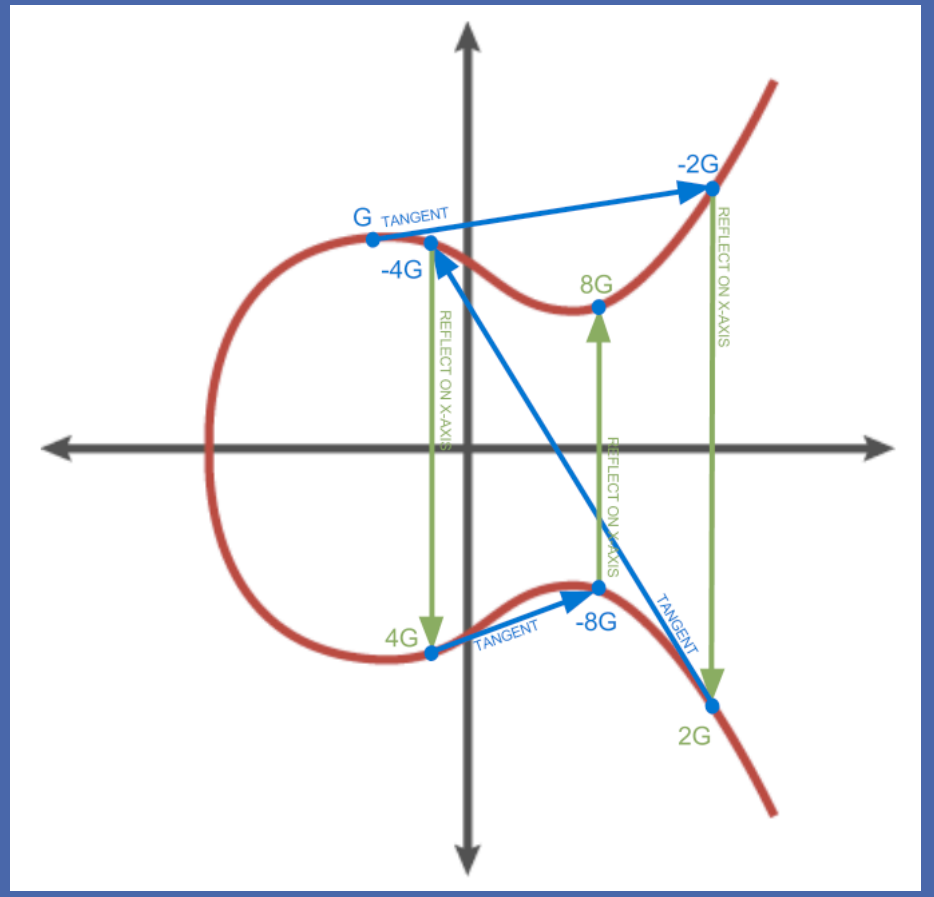
where k is the private key, G is the generator point, and K is the resulting public key, a point on the curve. Because the generator point is always the same for all bitcoin users, a private key k multiplied with G will always result in the same public key K. The relationship between k and K is fixed, but can only be calculated in one direction, from k to K. That's why a bitcoin address (derived from K) can be shared with anyone and does not reveal the user's private key (k).



Sidebar: **Speeding up vanity address generation.** In Bitcoin, if we call the private key x, the public key is g^x. The exponentiation represents what's called scalar multiplication in an elliptic curve group. The address is $H(g^x)$, the hash of the public key. We won't get into the details here, but exponentiation is the slow step in address generation.

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.63
Arvind Narayanan et. al., (2016). Bitcoin and Cryptocurrency Technologies. p.102

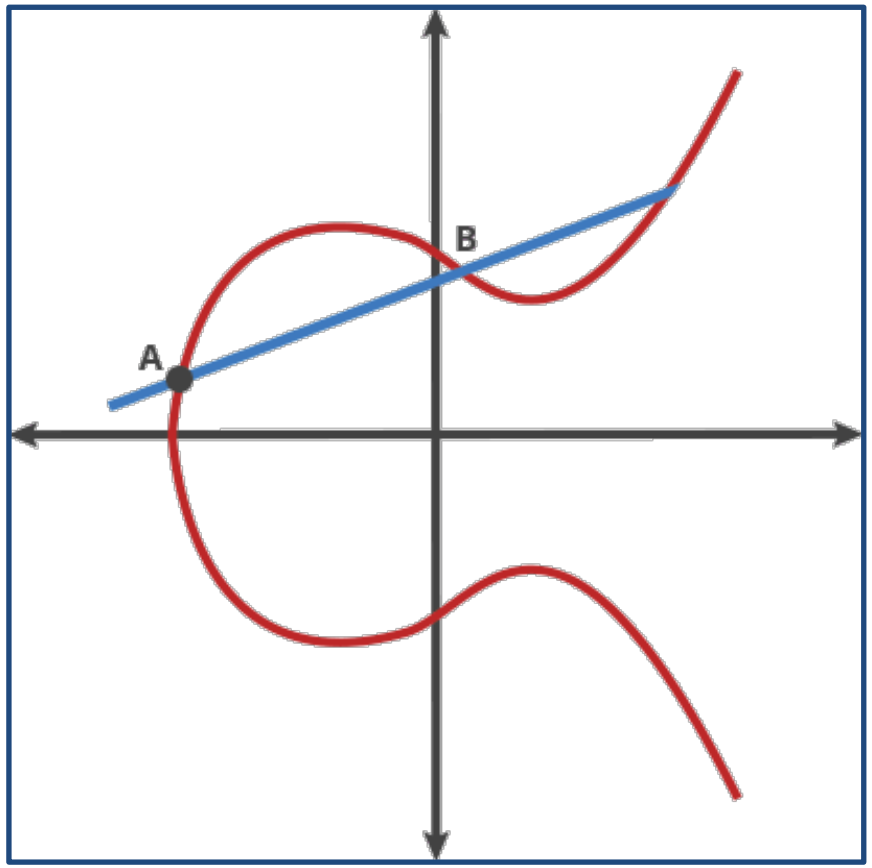
Elliptic Curve Digital Signature



CBS

Elliptic Curve Cryptography

A dot B = C
A dot C = D



- Elliptic curve cryptography is asymmetric cryptography based on the discrete logarithm problem as expressed by
  - addition and
  - multiplication

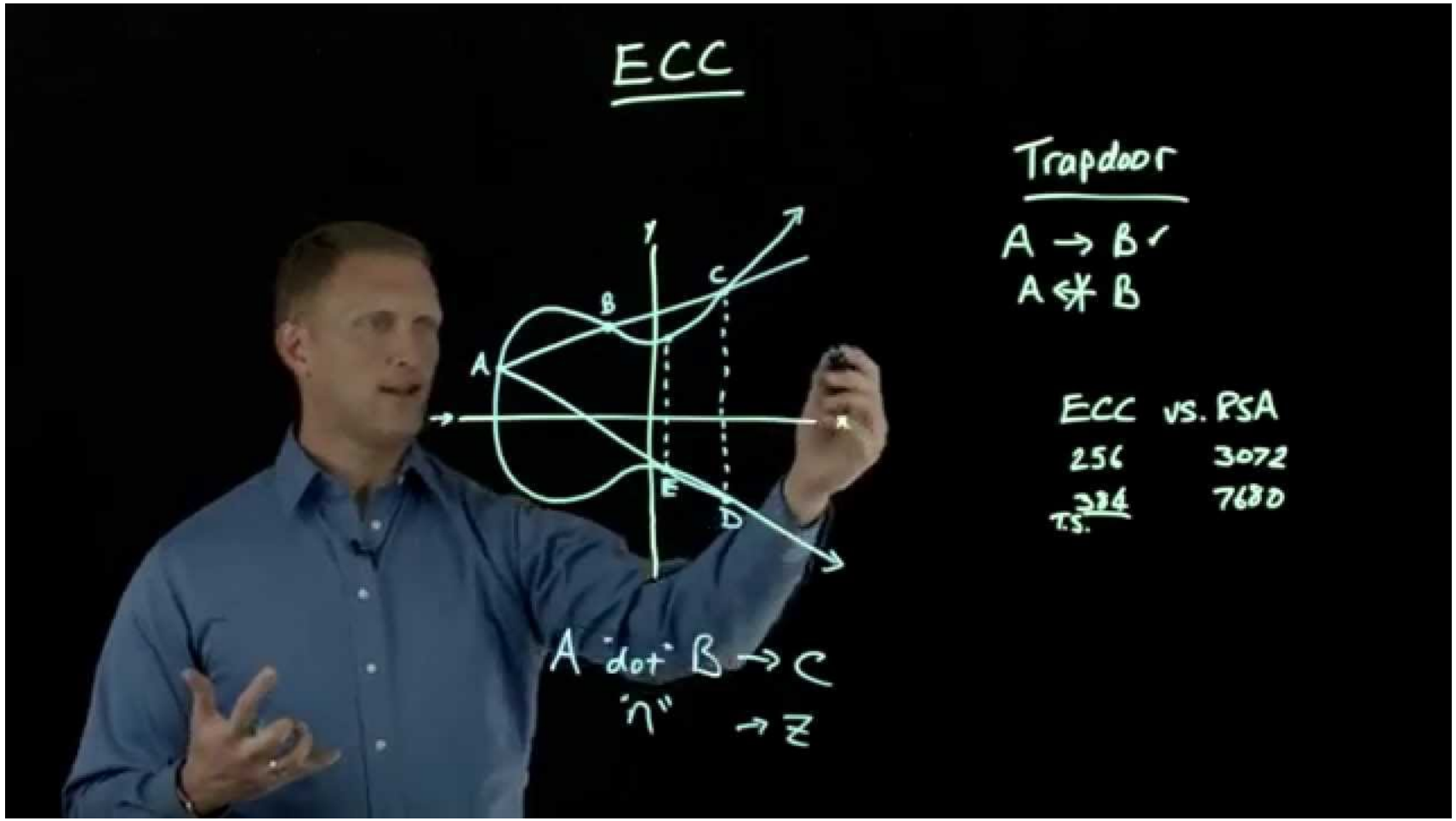
  on the points of an elliptic curve.
- Bitcoin uses an elliptic curve and set of mathematical constants, as defined in a standard called **secp256k1**

If $log_a b = x$, given x finding both a and b is very hard

https://cdn.arstechnica.net/wp-content/uploads/2013/10/elliptic-curve-crypt-image02.gif

https://arstechnica.com/information-technology/2013/10/a-relatively-easy-to-understand-primer-on-elliptic-curve-cryptography/

Elliptic Curve Cryptography Overview



Elliptic Curve Cryptography Overview: https://www.youtube.com/watch?v=dCvB-mhkT0w

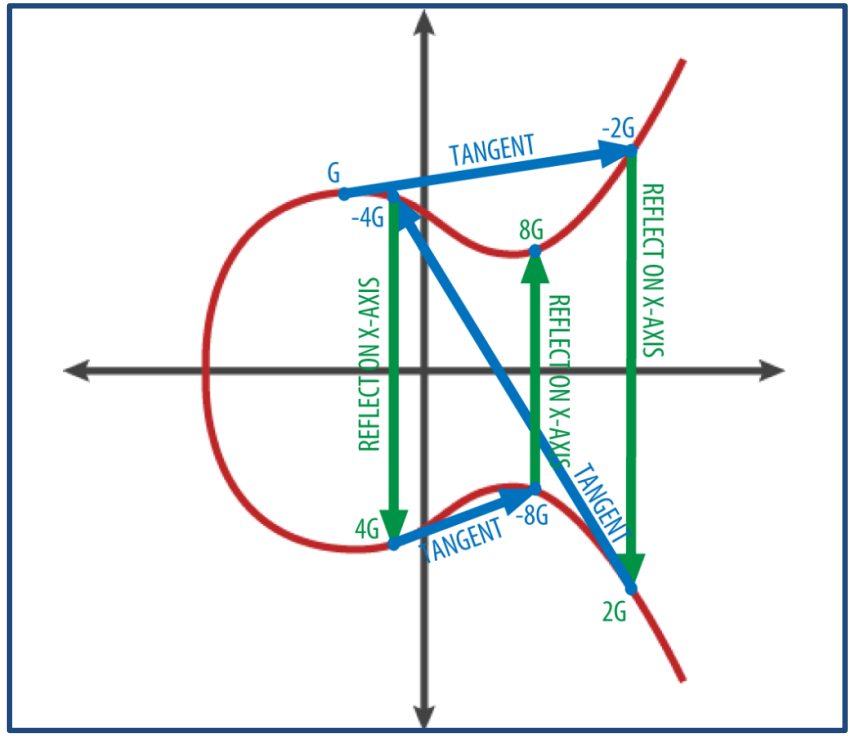
Elliptic Curve Cryptography

$$y^2 = \left(x^3 + 7\right) \text{ over } \left(\mathbb{F}_p\right)$$

or

$$y^2 \mod p = \left(x^3 + 7\right) \mod p$$

The *mod p* (modulo prime number p) indicates that this curve is over a finite field of prime order *p*, also written as $\mathbb{F}_p$, where p = $2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$, a very large prime number.



$$\begin{aligned}
4^0 &= 1 &&= 0 \times 7 + 1 &&\equiv 1 \pmod 7 \\
4^1 &= 4 &&= 0 \times 7 + 4 &&\equiv 4 \pmod 7 \\
4^2 &= 16 &&= 2 \times 7 + 2 &&\equiv 2 \pmod 7 \\
4^3 &= 64 &&= 9 \times 7 + 1 &&\equiv 1 \pmod 7 \\
4^4 &= 256 &&= 36 \times 7 + 4 &&\equiv 4 \pmod 7 \\
4^5 &= 1024 &&= 146 \times 7 + 2 &&\equiv 2 \pmod 7 \\
&\vdots
\end{aligned}$$

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.64
https://en.wikipedia.org/wiki/Modular_arithmetic

RSA versus ECDA

Table 10.3 Comparable Key Sizes in Terms of Computational Effort for Cryptanalysis

Symmetric Scheme (key size in bits)	ECC-Based Scheme (size of *n* in bits)	RSA/DSA (modulus size in bits)
56	112	512
80	160	1024
112	224	2048
128	256	3072
192	384	7680
256	512	15360

Source: Certicom

Stallings, W. (2006). Cryptography and network security, 5/E. Pearson Education. p.321

Public Key

Generating a Public Key

Starting with a private key in the form of a randomly generated number k, we multiply it by a predetermined point on the curve called the *generator point* G to produce another point somewhere else on the curve, which is the corresponding public key K. The generator point is specified as part of the `secp256k1` standard and is always the same for all keys in bitcoin:

$$K = k * G$$

where k is the private key, G is the generator point, and K is the resulting public key, a point on the curve. Because the generator point is always the same for all bitcoin users, a private key k multiplied with G will always result in the same public key K. The relationship between k and K is fixed, but can only be calculated in one direction, from k to K. That's why a bitcoin address (derived from K) can be shared with anyone and does not reveal the user's private key (k).

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.63

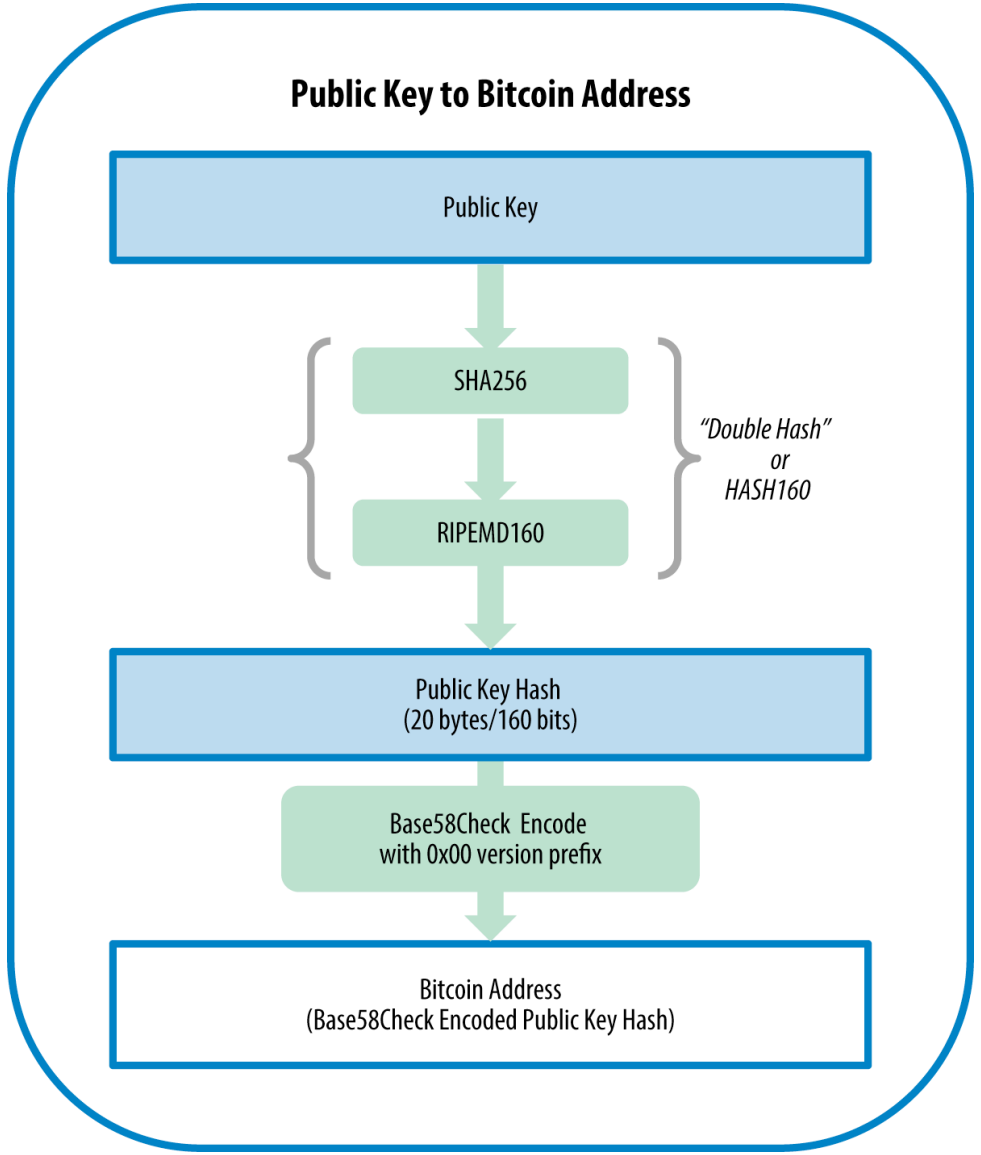
Bitcoin Addresses

Private key:	256 bits
Public key, uncompressed:	512 bits
Public key, compressed:	257 bits
Message to be signed:	256 bits
Signature:	512 bits

Bitcoin Addresses

A Bitcoin address is a string of digits and characters that can be shared with anyone who wants to send you money. Addresses produced from public keys consist of a string of numbers and letters, beginning with the digit "1". Here's an example of a Bitcoin address:

```
1J7mdg5rbQyUHENYdx39WVWK7fsLpEoXZy
```



Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.64

Bitcoin uses ECDSA standard

Elliptic Curve Digital Signature Algorithm

relies on hairy math

will skip the details here --- look it up if you care

good randomness is essential

foul this up in generateKeys() or sign() ?

probably leaked your private key

Hot and Cold Storage

Designed by studiogstock / Freepik

Slides based on Bitcoin and Cryptocurrency Technologies: http://bitcoinbook.cs.princeton.edu/

CBS

Hot and Cold Storage

- As we just saw, storing bitcoins on your computer is like carrying money around in your wallet or your purse. This is called “hot storage”. It’s convenient but also somewhat risky.
- On the other hand, “cold storage” is offline. It's locked away somewhere. It's not connected to the internet, and it's archival. So it’s safer and more secure, but of course, not as convenient.
- This is similar to how you carry some money around on your purse, but put your life's savings somewhere safer.

Arvind Narayanan et. al., (2016). Bitcoin and Cryptocurrency Technologies. p.104

Hot storage

online

convenient but risky

Cold storage

offline

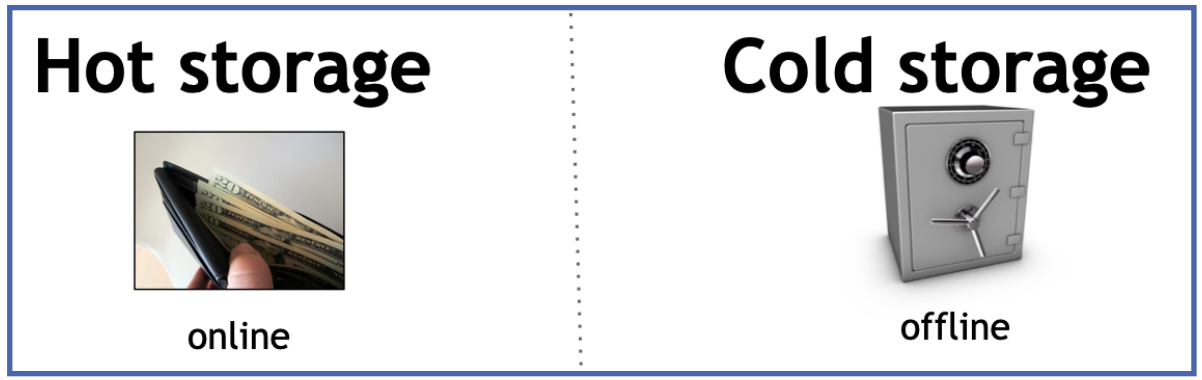
archival but safer

separate keys

Arvind Narayanan et. al., (2016). Bitcoin and Cryptocurrency Technologies. p.104

Payments between Hot and Cold Storage

- To have separate hot and cold storage, obviously you need to have separate secret keys for each — otherwise the coins in cold storage would be vulnerable if the hot storage is compromised.
- You'll want to move coins back and forth between the hot and the cold side, so each side will need to know the other's addresses or public keys.
- Cold storage is not online, and so the hot storage and the cold storage won't be able to connect to each other across any network.



BBC Podcast: The Lazarus Heist:

S2.8 Bitcoin bandits https://www.bbc.co.uk/programmes/w3ct5fc5

S1.4 Billion dollar hack https://www.bbc.co.uk/programmes/p09gy7jm

CBS CENTRE FOR BUSINESS DATA ANALYTICS

Arvind Narayanan et. al., (2016). Bitcoin and Cryptocurrency Technologies. p.104

Hot storage

online

hot secret key(s)

cold address(es)

Cold storage

offline

cold secret key(s)

hot address(es)

payments

Hot storage

online

hot secret key(s)

cold address(es)

payments

Cold storage

offline

Problem:

Want to use a new address (and key) for each coin sent to cold

But how can hot wallet learn new addresses if cold wallet is offline?

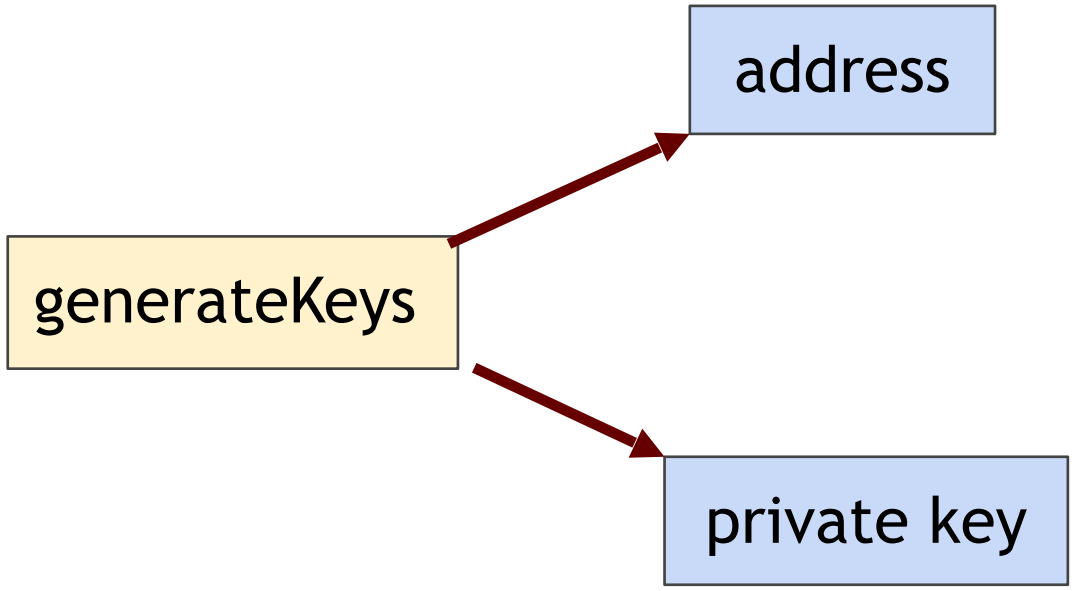
Awkward solution:

Generate a big batch of addresses/keys, transfer to hot beforehand

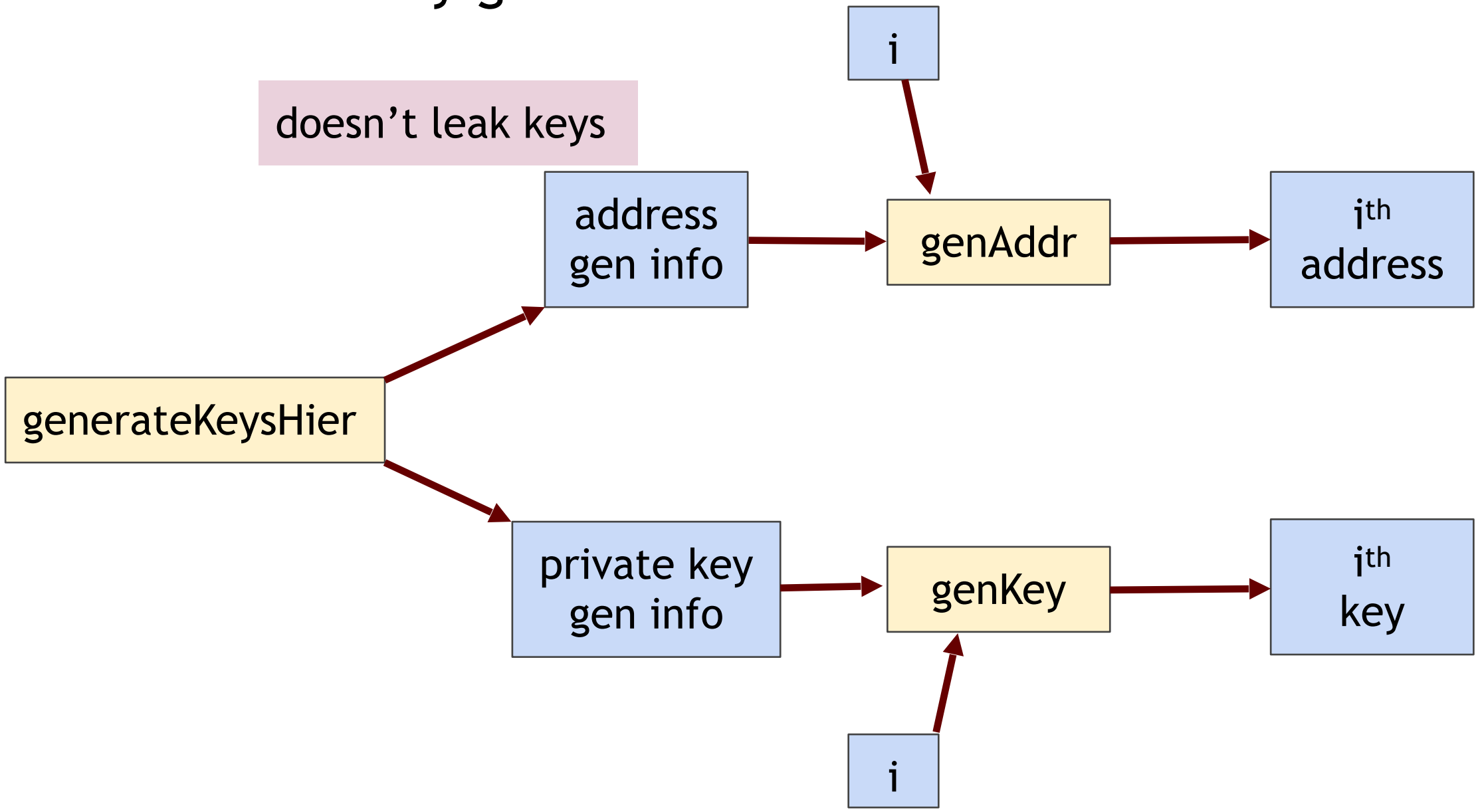
Better solution:

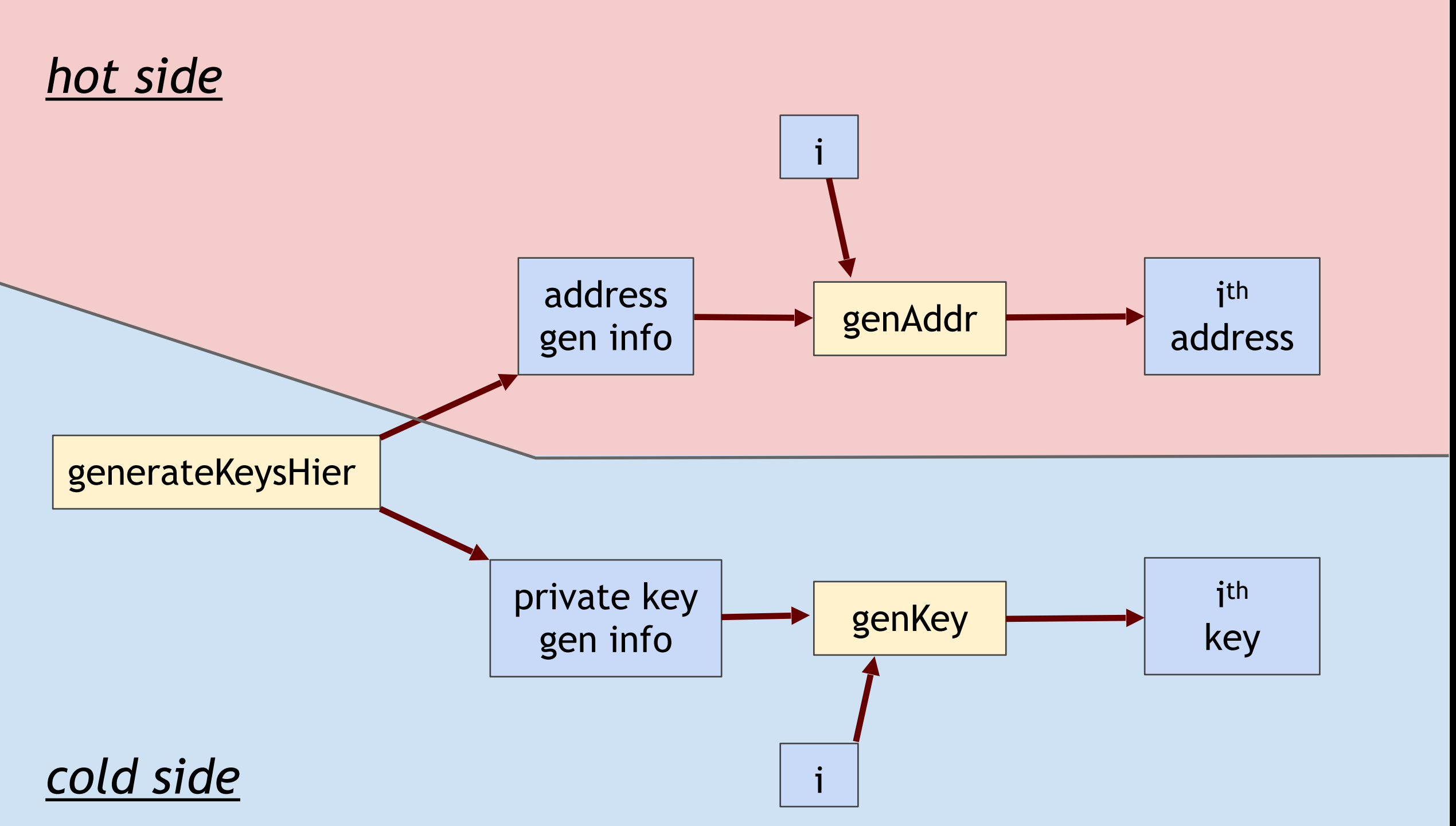
Hierarchical wallet

Regular key generation (Awkward solution)



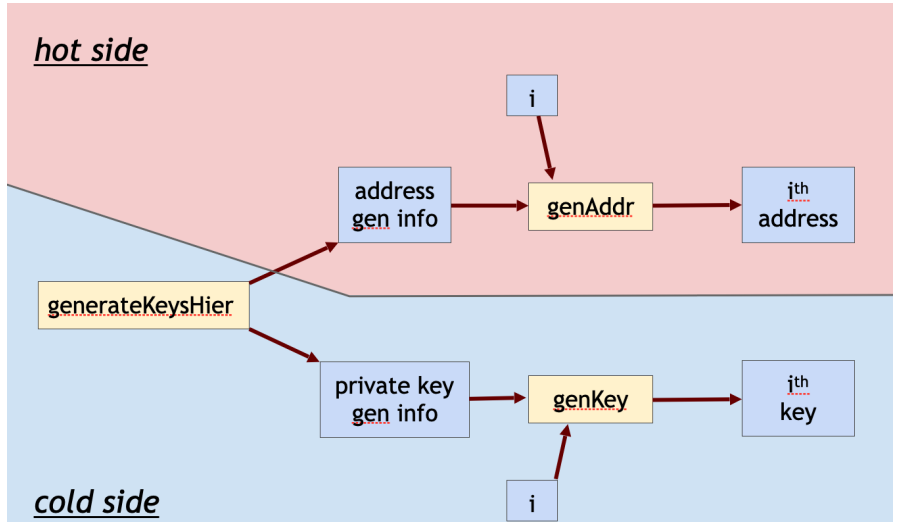
Hierarchical key generation:




hot side
i
address
gen info
genAddr
ith
address
generateKeysHier
private key
gen info
genKey
ith
key
i
cold side

Hot and Cold Storage

- A more effective solution is to use a hierarchical wallet. It allows the cold side to use an essentially unbounded number of addresses and the hot side to know about these addresses, but with only a short, one-time communication between the two sides.
- In a hierarchical wallet, key generation works differently. Instead of generating a single address, we generate what we'll call address generation info, and rather than a private key, we generate what we'll call private key generation info.
- Given the address generation info, we can generate a sequence of addresses by passing integer *i* as input to generate i^{th} address and key
- It is the cryptographic magic that both i^{th} address and i^{th} key both match exactly like pair of keys generated old fashioned way



Private key generation info:	k, x, y
i^{th} private key:	$x_i = y + H(k \mid\mid i)$
Address generation info:	k, g^y
i^{th} public key:	$g^{x\_i} = g^{H(k \mid\mid i)} \cdot g^y$
i^{th} address:	$H(g^{x\_i})$

Arvind Narayanan et. al., (2016). Bitcoin and Cryptocurrency Technologies. p.104

Several Ways to store cold info

(1) Info stored in device, device locked in a safe

(2) “Brain wallet” encrypt info under passphrase that user remembers

(3) Paper wallet print info on paper, lock up the paper

(4) In “tamperproof” device device will sign things for you, but won’t divulge keys (e.g. trezor hardware wallet)

Wallets Types

https://alexey-shepelev.medium.com/hierarchical-key-generation-fc27560f786

Slides based on

1) Bitcoin and Cryptocurrency Technologies: http://bitcoinbook.cs.princeton.edu/

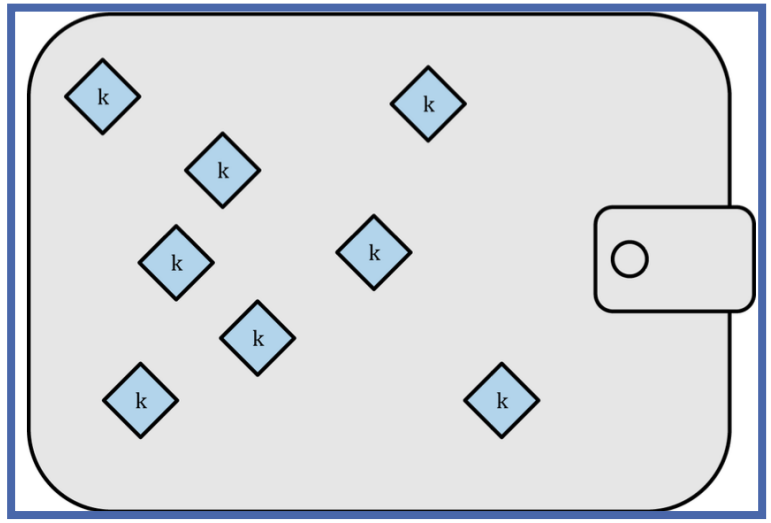
2) Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.”.

CBS

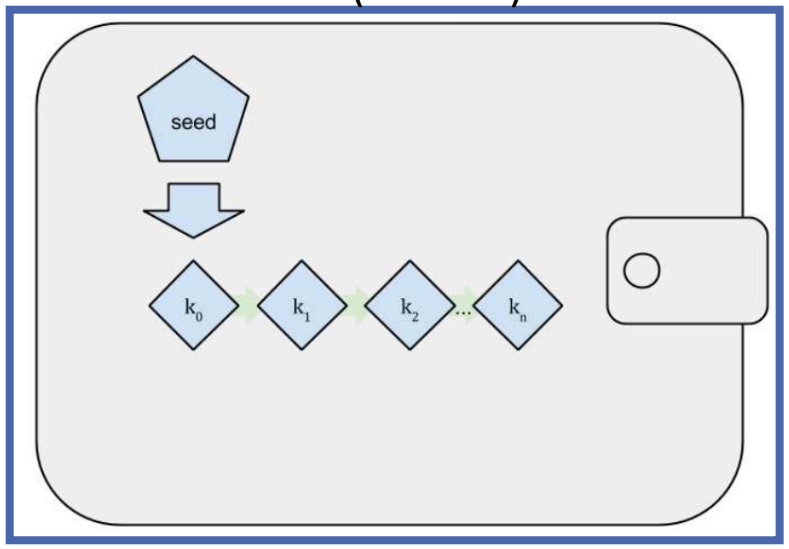
Wallets

- There are two primary types of wallets, distinguished by whether the keys they contain are related to each other or not.
- The first type is a **nondeterministic wallet**, where each key is independently generated from a random number, where keys are not related to each other.
- The second type of wallet is a **deterministic wallet**, where all the keys are derived from a single master key, known as the seed and keys in this type of wallet are related to each other. (**Hierarchical key generation**)

Nondeterministic (Random) Wallets



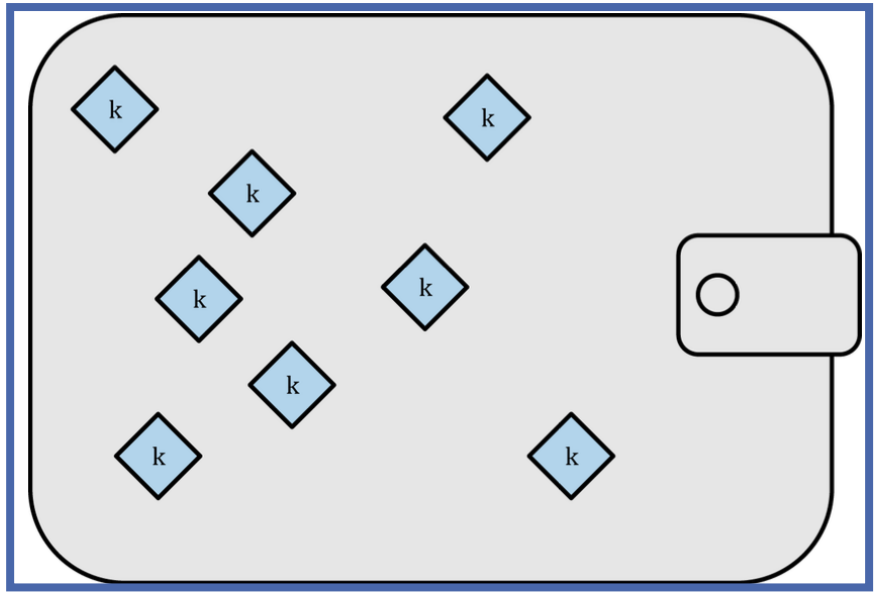
Deterministic (Seeded) Wallets

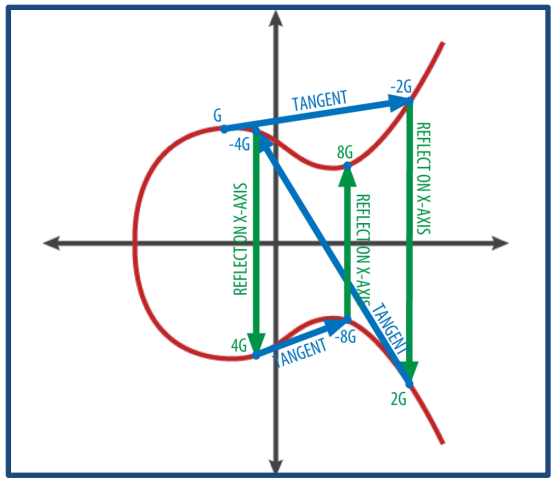


Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.95

Nondeterministic Wallets

- In non-deterministic wallets, each key is independently generated from a random number.
- Wallets are collections of randomly generated private keys. E.g., the original Bitcoin Core client pre-generates 100 random private keys, each key used only once.
- This type of wallet is also known as a JBOK wallet from the phrase “Just a Bunch Of Keys.”
- The disadvantage of random keys is that you must keep copies of all of them, meaning that the wallet must be backed up frequently.
- Each key must be backed up, or the funds it controls are lost if the wallet becomes inaccessible.





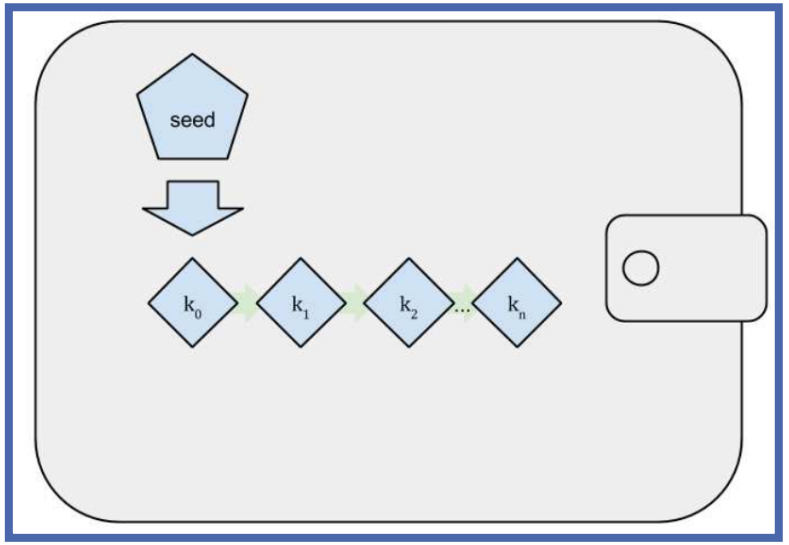
$$K = k * G$$

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.”. p.94

Deterministic Wallets

- Deterministic, or “seeded,” wallets are wallets that contain private keys that are all derived from a common seed through the use of a one-way hash function.
- The seed is a random number that is combined with other data, such as an index number or 'chain code' to derive the private keys.
- In a deterministic wallet, the seed is sufficient to recover all the derived keys, and to backup at creation time is sufficient
- The seed is also sufficient for a wallet export or import, allowing for easy migration of all the user’s keys between different wallet implementations.
- Most commonly used derivation method uses a tree-like structure known as a hierarchical deterministic or HD wallet.

Deterministic (seeded) Wallets

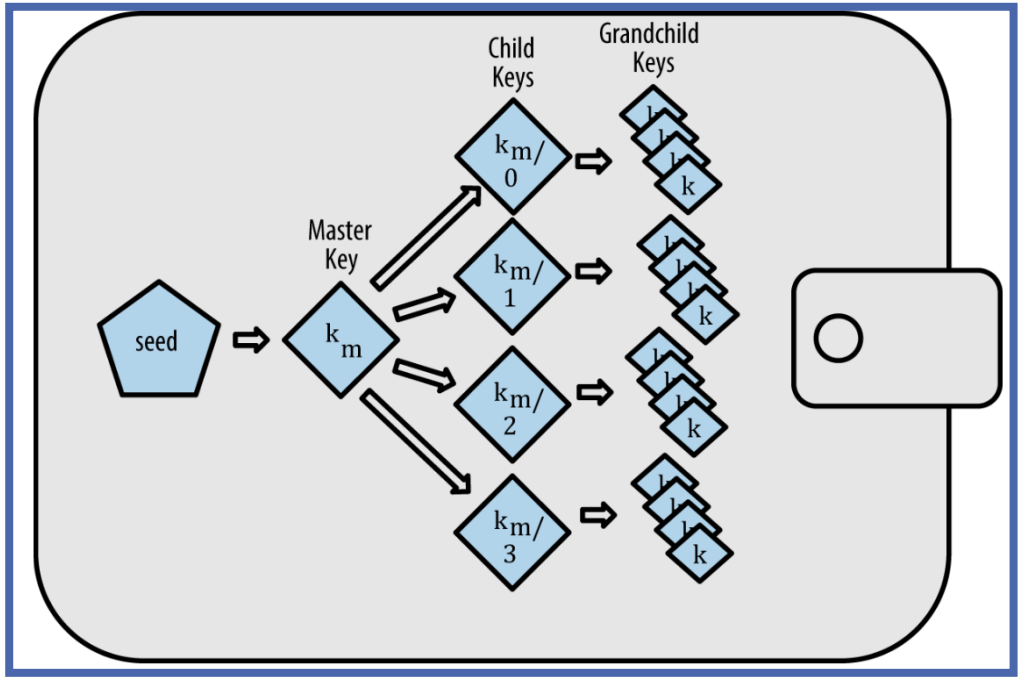


Private key generation info:	k, x, y
i^{th} private key:	$x_i = y + H(k \,//\, i)$
Address generation info:	k, g^y
i^{th} public key:	$g^{x\_i} = g^{H(k \,//\, i)} \cdot g^y$
i^{th} address:	$H(g^{x\_i})$

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.”. p.95

Hierarchical Deterministic (HD) Wallet

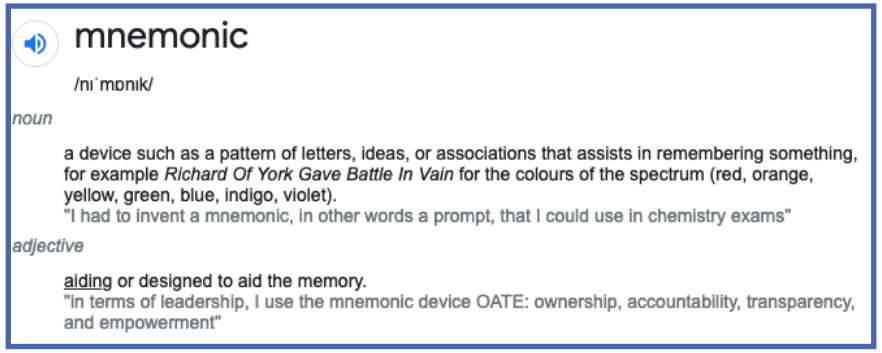
- Most advanced form of deterministic wallet is the HD wallet defined by the BIP-32 (BIP: Bitcoin Improvement Proposal) standard.
- HD wallets contain keys derived in a tree structure, such that a parent key can derive a sequence of children keys, each of which can derive a sequence of grandchildren keys, to an infinite depth.
- Tree structure is used to express additional organizational meaning, i.e. a specific branch of subkeys is used to receive incoming payments and a different branch used to receive change from outgoing payments.
- Branches of keys can also be used in corporate settings, allocating different branches to departments, subsidiaries, specific functions, or accounting etc.



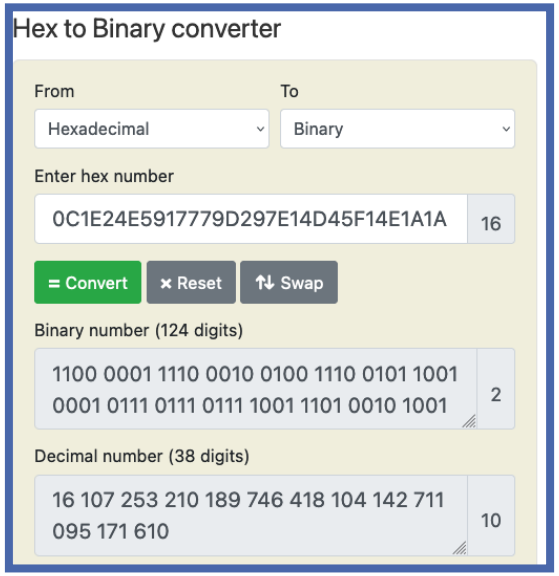
Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.95

Seeds and Mnemonic Codes

- HD wallets are a very powerful mechanism for managing many keys and addresses and random seed is most important.
- A standardized way of creating seeds from a sequence of English words that are easy to transcribe, export/import across wallets.
- This is known as a mnemonic and the standard is defined by BIP-39.
- Most bitcoin wallets (other cryptocurrencies also) use this standard and can import and export seeds for backup and recovery using interoperable mnemonics.



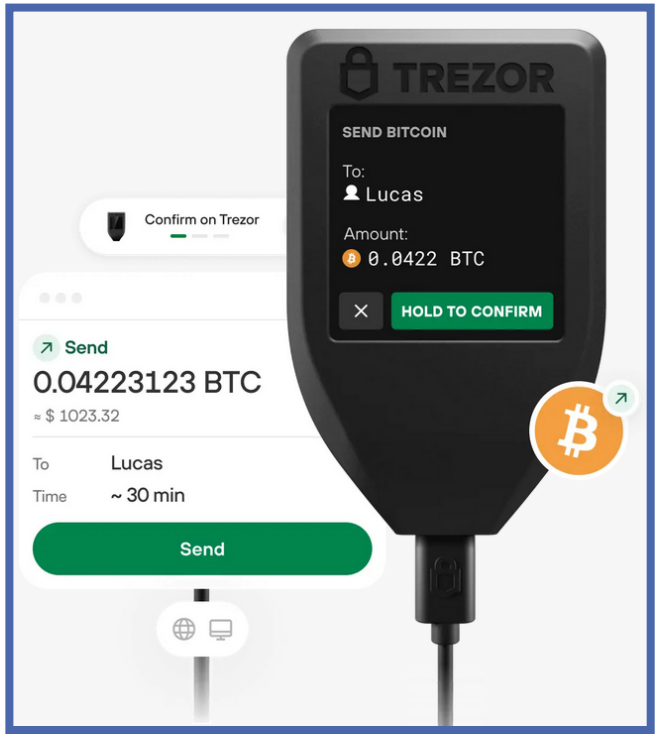

```
0C1E24E5917779D297E14D45F14E1A1A

army van defense carry jealous true
garbage claim echo media make crunch
```



Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.95

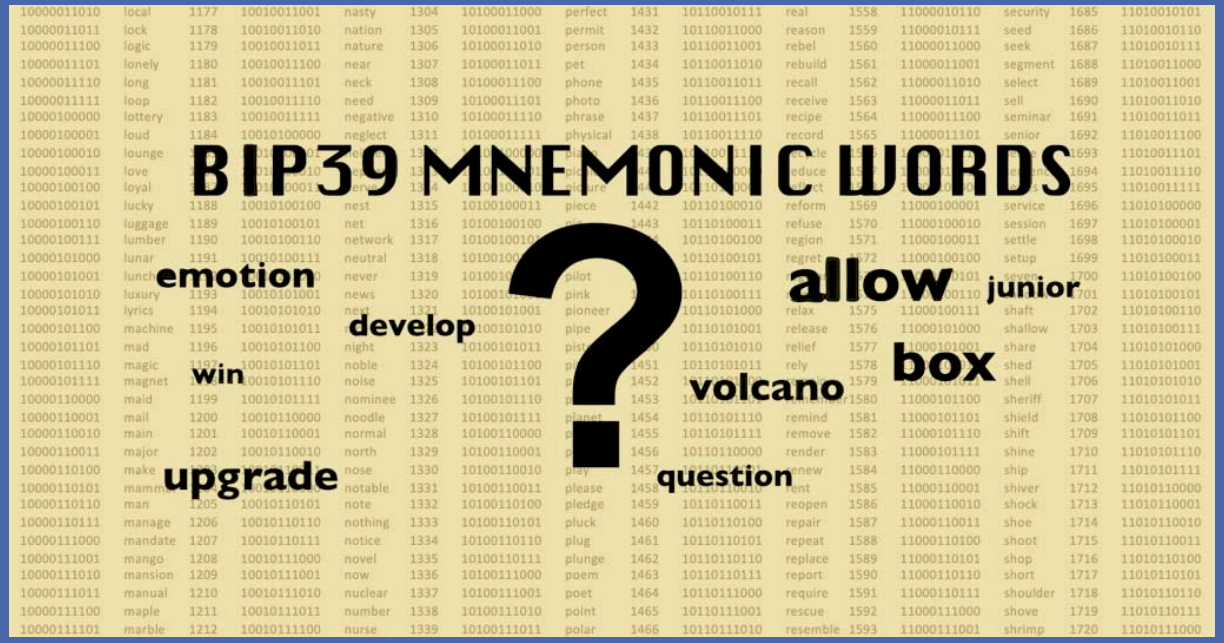
Hardware Wallets

- Trezor is a bitcoin hardware wallet
- The Trezor is a simple USB device with two buttons that stores keys (in the form of an HD wallet) and signs transactions.
- The device generates a mnemonic and seed from a built-in hardware random number generator.
- During this initialisation phase, the wallet displayed a numbered sequence of words, one by one, on the screen
- Sequence of words is important, so mnemonic paper backups have numbered spaces for each word.



Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.95

Mnemonic Codes and Generation



https://coinguides.org/generating-custom-mnemonic-seed-phrases/

Slides based on

1) Bitcoin and Cryptocurrency Technologies: http://bitcoinbook.cs.princeton.edu/

2) Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.".

CBS

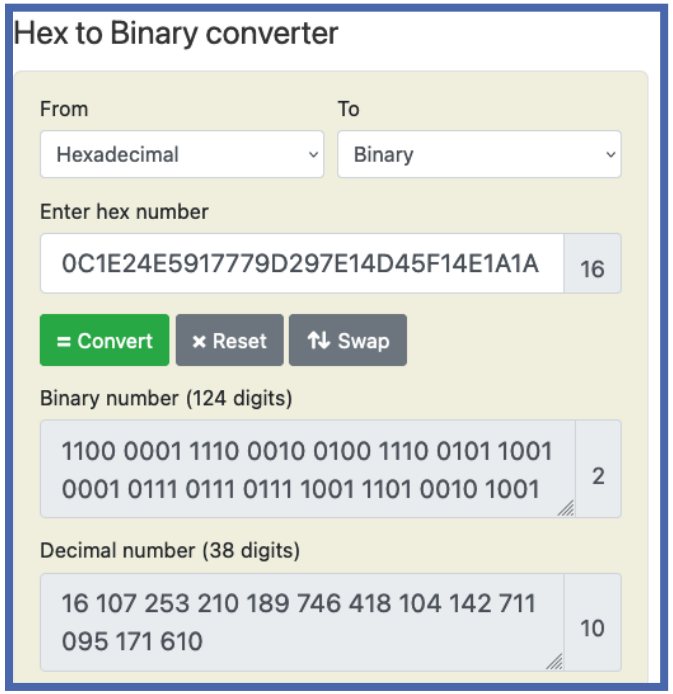
Mnemonic Code Words (BIP-39)

- Mnemonic code words are word sequences that represent (encode) a random number used as a seed to derive a deterministic wallet.

```
0C1E24E5917779D297E14D45F14E1A1A

army van defense carry jealous true
garbage claim echo media make crunch
```

- The sequence of words (12 to 24 words) is sufficient to re-create the seed and from there re-create the wallet and all the derived keys.

- Mnemonic words make it easier for users to back up wallets because they are easy to read and correctly transcribe, as compared to a random sequence of numbers.



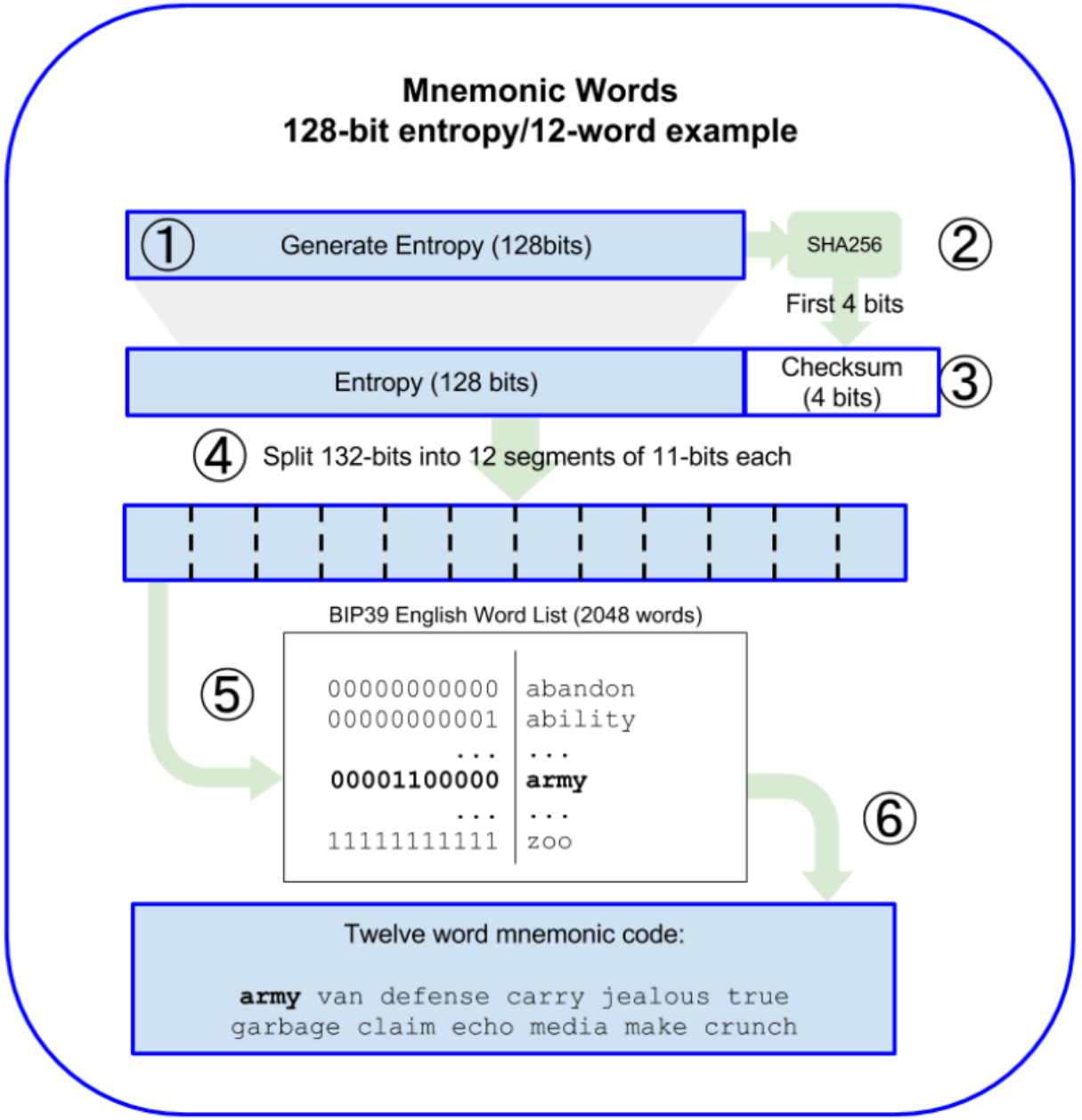
Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.95

Generating Mnemonic Words

1. Create a random sequence (entropy) of 128 to 256 bits.
2. Create a checksum of the random sequence by taking the first (entropy-length/32) bits of its SHA256 hash.
3. Add the checksum to the end of the random sequence.
4. Divide the sequence into sections of 11 bits.
5. Map each 11-bit value to a word from the predefined dictionary of 2048 words.
6. The mnemonic code is the sequence of words.

Table 5-2. Mnemonic codes: entropy and word length

Entropy (bits)	Checksum (bits)	Entropy + checksum (bits)	Mnemonic length (words)
128	4	132	12
160	5	165	15
192	6	198	18
224	7	231	21
256	8	264	24



Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.101

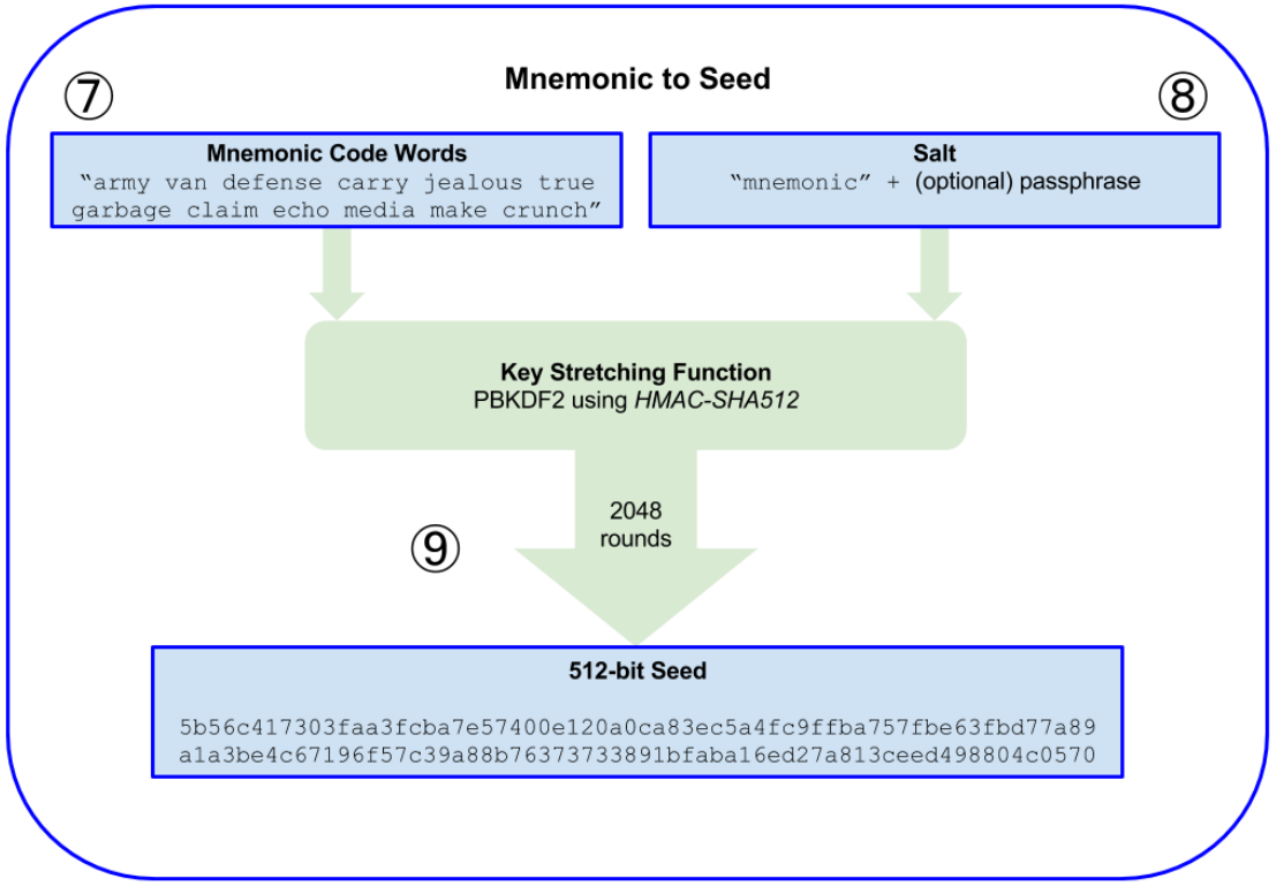
https://getcoinplate.com/blog/official-bip39-word-list-mnemonic-in-english-verified/

Mnemonic Words to Seed

7. The first parameter to the PBKDF2 key-stretching function is the mnemonic produced from step 6.

8. The second parameter to the PBKDF2 key-stretching function is salt. The salt is composed of the string constant "mnemonic" concatenated with an optional user-supplied passphrase string.

9. PBKDF2 stretches the mnemonic and salt parameters using 2048 rounds of hash ing with the HMAC-SHA512 algorithm, producing a 512-bit value as its final output. That 512-bit value is the seed.



Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.101

Mnemonic Words to Seed

Table 5-3. 128-bit entropy mnemonic code, no passphrase, resulting seed

Entropy input (128 bits)

`0c1e24e5917779d297e14d45f14e1a1a`

Mnemonic (12 words)

`army van defense carry jealous true garbage claim echo media make crunch`

Passphrase

(none)

Seed (512 bits)

`5b56c417303faa3fcba7e57400e120a0ca83ec5a4fc9ffba757fbe63fbd77a89a1a3be4c67196f57c39a88b76373733891bfaba16ed27a813ceed498804c0570`

Table 5-4. 128-bit entropy mnemonic code, with passphrase, resulting seed

Entropy input (128 bits)

`0c1e24e5917779d297e14d45f14e1a1a`

Mnemonic (12 words)

`army van defense carry jealous true garbage claim echo media make crunch`

Passphrase

SuperDuperSecret

Seed (512 bits)

`3b5df16df2157104cfdd22830162a5e170c0161653e3afe6c88defeefb0818c793dbb28ab3ab091897d0715861dc8a18358f80b79d49acf64142ae57037d1d54`

https://iancoleman.io/bip39/

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.103

Mnemonic Words to Seed

The optional passphrase creates two important features:

- A second factor (something memorized) that makes a mnemonic useless on its own, protecting mnemonic backups from compromise by a thief.
- A form of plausible deniability or "duress wallet," where a chosen passphrase leads to a wallet with a small amount of funds used to distract an attacker from the "real" wallet that contains the majority of funds.

However, it is important to note that the use of a passphrase also introduces the risk of loss:

- If the wallet owner is incapacitated or dead and no one else knows the passphrase, the seed is useless and all the funds stored in the wallet are lost forever.
- Conversely, if the owner backs up the passphrase in the same place as the seed, it defeats the purpose of a second factor.

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.104

Hierarchical Key Generation

https://cointelegraph.com/explained/what-are-hierarchical-deterministic-hd-crypto-wallets

Slides based on

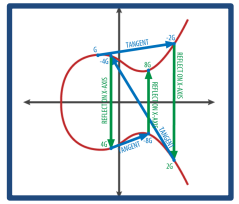
1) Bitcoin and Cryptocurrency Technologies: http://bitcoinbook.cs.princeton.edu/

2) Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.”.

CBS

HD Wallet from Random Seed

- HD wallets are created from a single root seed, which is a 128-, 256-, or 512-bit random number.
- Most commonly, this seed is generated from a mnemonic as detailed in the previous section.
- Every key is deterministically derived from this root seed, making it possible to re-create the entire HD wallet.
- Easy to back up, restore, export, and import HD wallets containing thousands or even millions of keys by simply transferring only the mnemonic that the root seed is derived from.
- The master private key (m) then generates a corresponding master public key (M) using the normal elliptic curve multiplication process k * G.



$$K = k * G$$

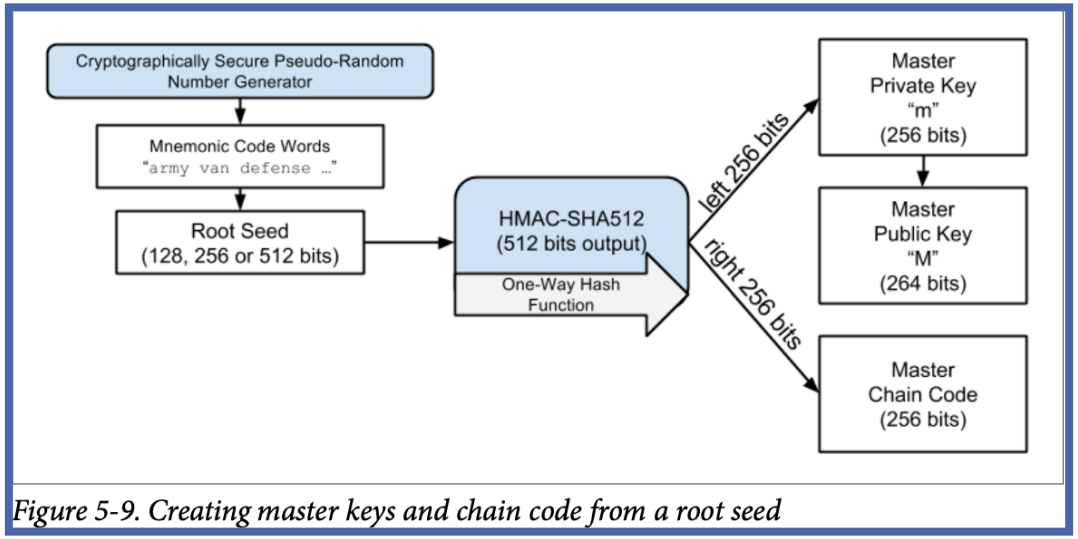
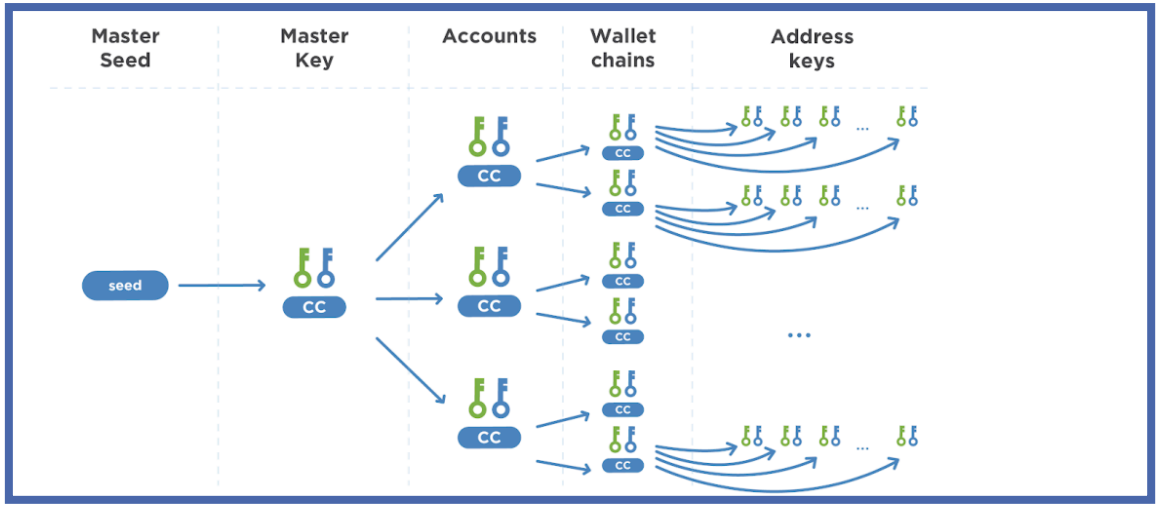


Figure 5-9. Creating master keys and chain code from a root seed



https://alexey-shepelev.medium.com/hierarchical-key-generation-fc27560f786

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.106

Derivation of private child key

- HD wallets use a child key derivation (CKD) function to derive child keys from parent keys, based on a one-way hash function that combines:
  1. A parent private or public key (ECDSA uncompressed key)
  2. A seed called a chain code (256 bits)
  3. An index number (32 bits)

- The chain code introduces deterministic random data to the process so that knowing the index and a child key is insufficient to derive other child keys.

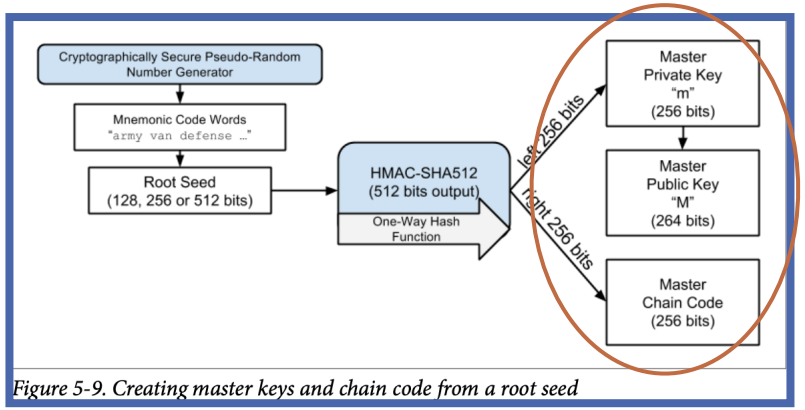


Figure 5-9. Creating master keys and chain code from a root seed

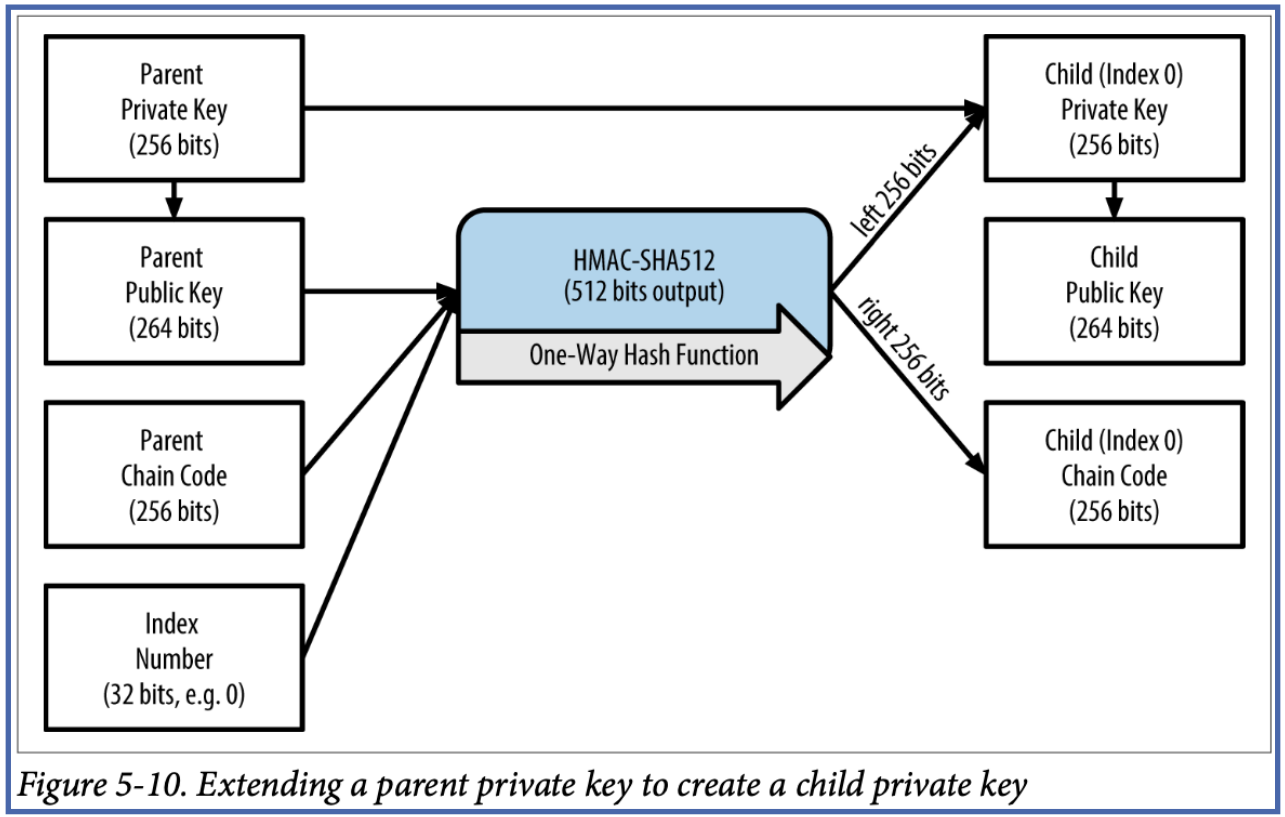


Figure 5-10. Extending a parent private key to create a child private key

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.106

How we derive private child key

- Knowing 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 tree's root) is made from the seed, while subsequent child chain codes are derived from each parent chain code.
- These three items (parent key, chain code, and index) are combined and hashed to generate children keys
- The parent public key, chain code, and the index number are combined and hashed with the HMAC-SHA512 algorithm to produce a 512-bit hash.
- This 512-bit hash is split into two 256-bit halves. The right-half 256 bits of the hash output become the chain code for the child.
- The left-half 256 bits of the hash are added to the parent private key to produce the child private key.

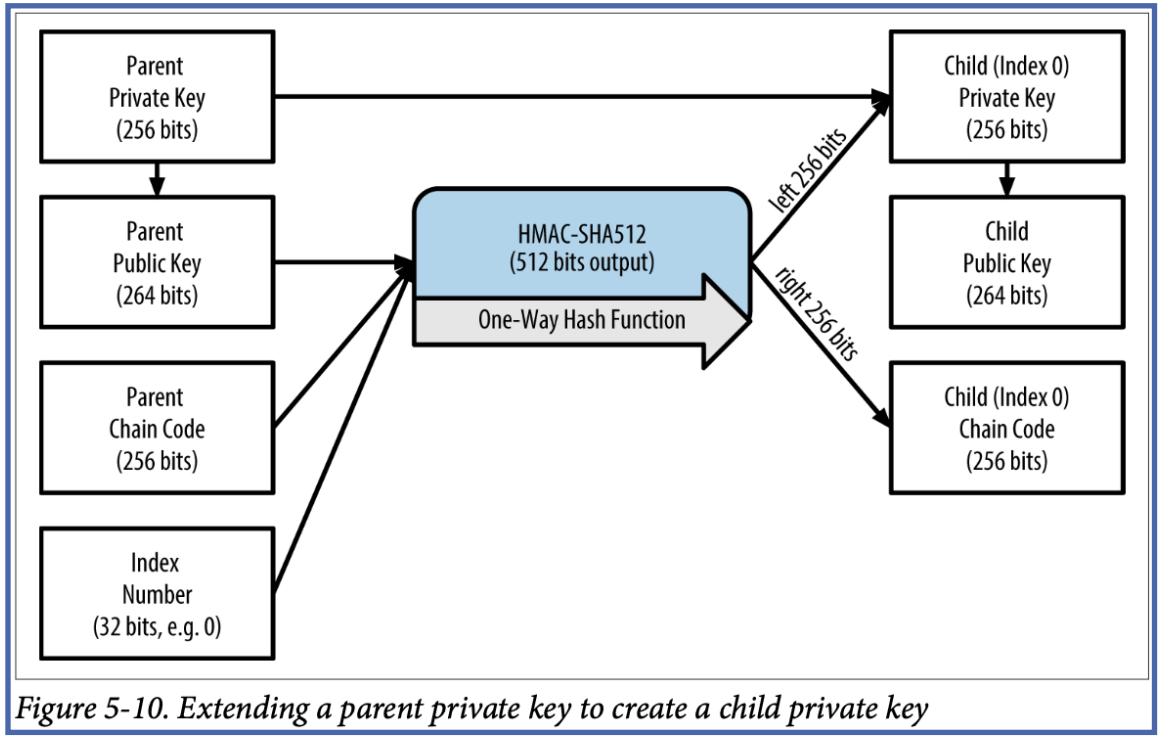


Figure 5-10. Extending a parent private key to create a child private key

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.106

Derivation of private child key

- Index set to 0 to produce the “zero” (first by index) child of the parent.
- Changing the index allows us to extend the parent and create the other children in the sequence, e.g., Child 0, Child 1, Child 2, etc.
- Each parent key can have 2,147,483,647 (2^{31}) children (2^{31} is half of the entire 232 range available because the other half is reserved for a special type of derivation
- Repeating the process one level down the tree, each child can in turn become a parent and create its own children, in an infinite number of generations.

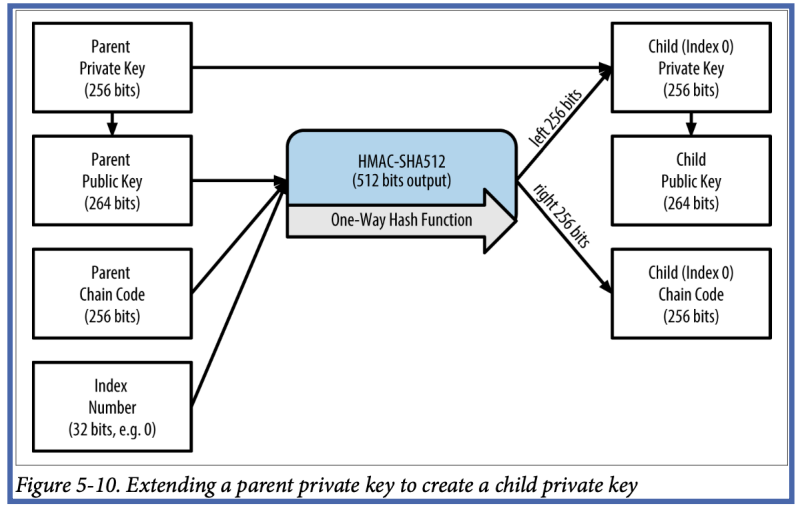
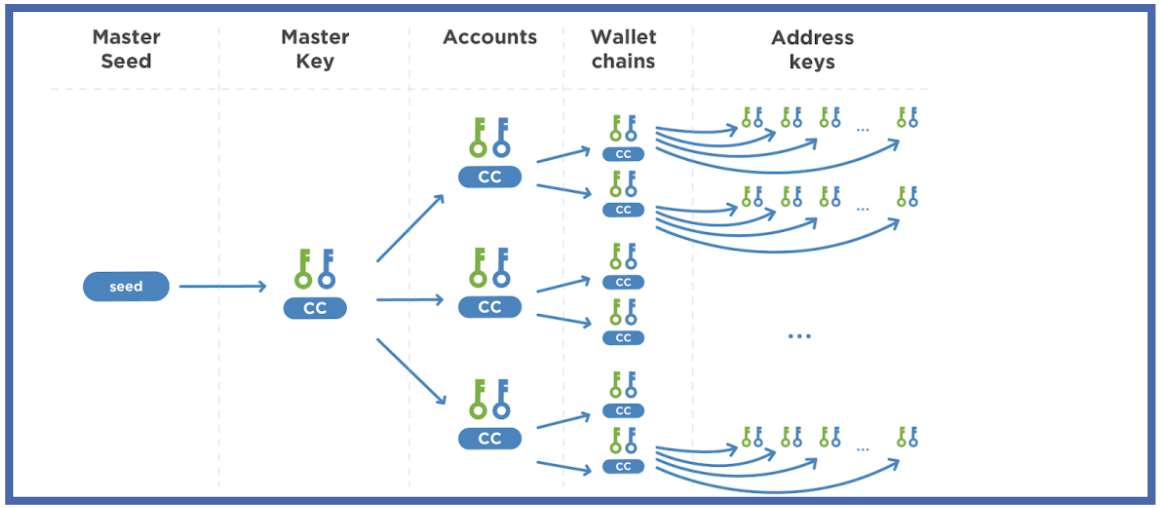


Figure 5-10. Extending a parent private key to create a child private key



https://alexey-shepelev.medium.com/hierarchical-key-generation-fc27560f786

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.”. p.106

Using Derived Child Keys

- Child private keys are indistinguishable from nondeterministic (random) keys.
- Because the derivation function is a one-way function, the child key cannot be used to find the parent key. The child key also cannot be used to find any siblings.
- Only the parent key and chain code can derive all the children. Without the child chain code, the child key cannot be used to derive any grandchildren either. You need both the child private key and the child chain code to start a new branch and derive grandchildren.
- So what can the child private key be used for on its own? It can be used to make a public key and a bitcoin address. Then, it can be used to sign transactions to spend anything paid to that address.

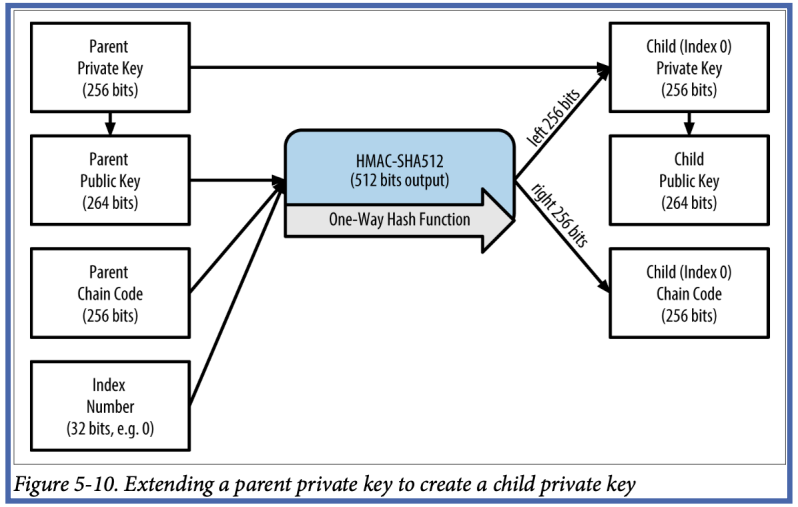
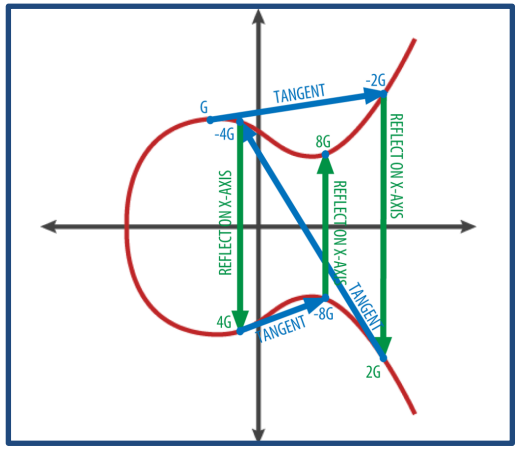


Figure 5-10. Extending a parent private key to create a child private key



$$K = k * G$$

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.106

Optional / Extra Slides

Image by MichaelWuensch from Pixabay

CBS

Extended Public Child Key Derivation

- A very useful characteristic of HD wallets is the ability to derive public child keys from public parent keys, without having the private keys. Remember ***cold addresses for hot storage***

- This gives us two ways to derive a child public key: either from the child private key, or directly from the parent public key.

- An extended public key can be used to derive all of the public keys (and only the public keys) in that branch of the HD wallet structure.

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.110

Extended Public Child Key Derivation

- This shortcut can create very secure public keys–only deployments where a server or application has a copy of an extended public key and no private keys whatsoever.
- That kind of deployment can produce an infinite number of public keys and bitcoin addresses but cannot spend any of the money sent to those addresses.
- Meanwhile, on another, more secure server, the extended private key can derive all the corresponding private keys to sign transactions and spend the money.
- One common application of this solution is to install an extended public key on a web server that serves an ecommerce application.

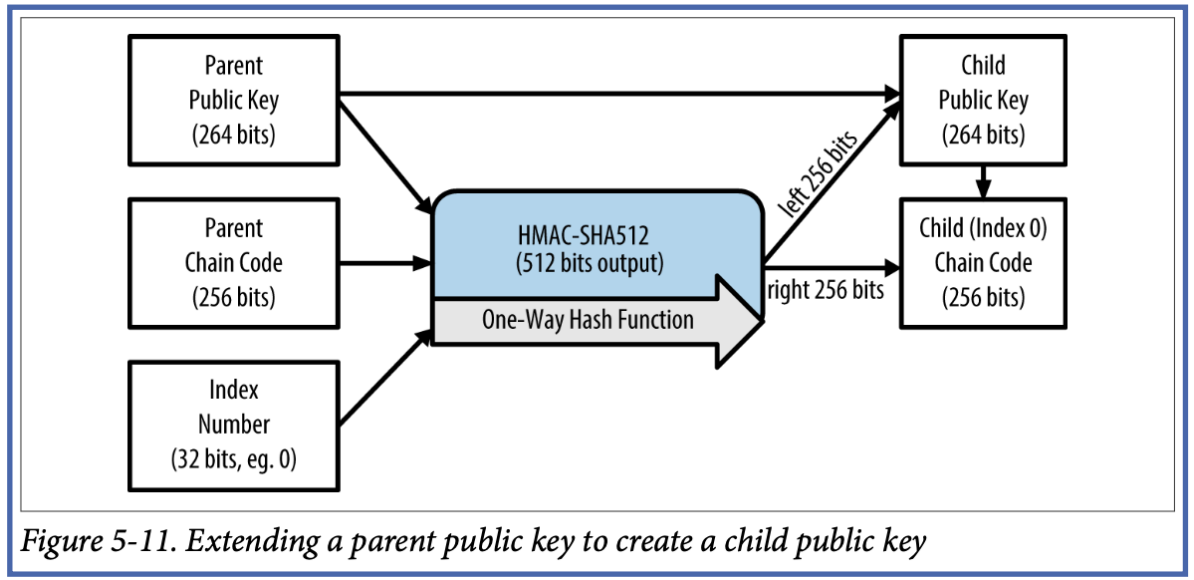


Figure 5-11. Extending a parent public key to create a child public key

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.110

Extended Public Child Key Derivation

- Another common application of this solution is for cold storage or hardware wallets.
- In that scenario, the extended private key can be stored on a paper wallet or hardware device (such as a Trezor hardware wallet), while the extended public key can be kept online.
- The user can create “receive” addresses at will while the private keys are safely stored offline.
- To spend the funds, the user can use the extended private key on an offline signing bitcoin client or sign transactions on the hardware wallet device (e.g., Trezor).

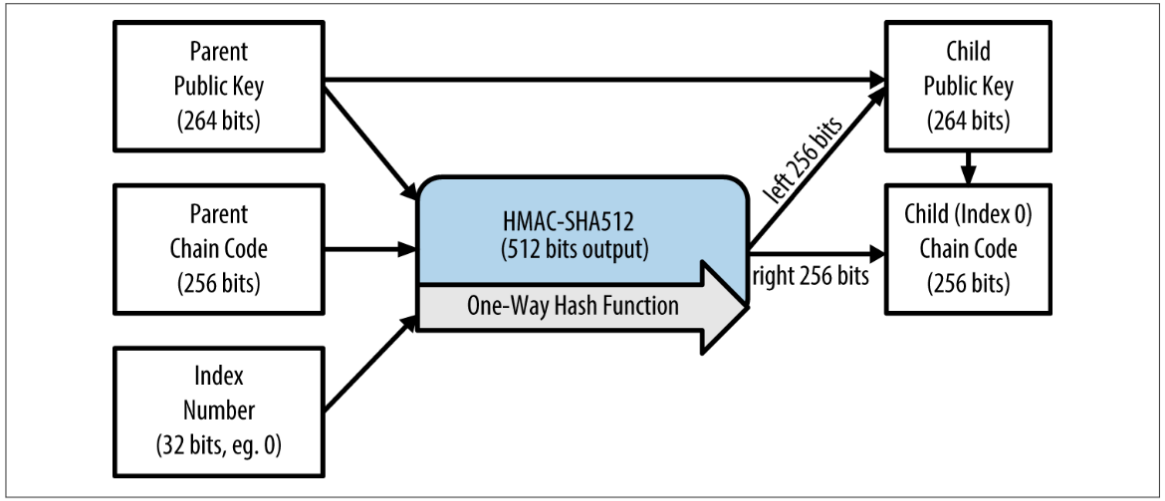


Figure 5-11. Extending a parent public key to create a child public key

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.”. p.110

Hardened Child Key Derivation - I

- The ability to derive a branch of public keys from an extended public key (xpub) is very useful but comes with a potential risk.
- Access to an xpub does not give access to child private keys. However, because the xpub contains the chain code, if a child private key is known, or somehow leaked, it can be used with the chain code to derive all the other child private keys.
- A single leaked child private key, together with a parent chain code, reveals all the private keys of all the children. Worse, the child private key together with a parent chain code can be used to deduce the parent private key.
- To counter this risk, HD wallets use an alternative derivation function called hardened derivation, which “breaks” the relationship between parent public key and child chain code.

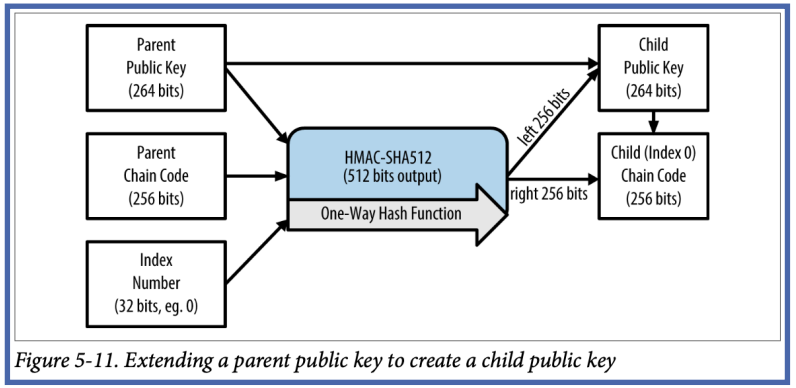
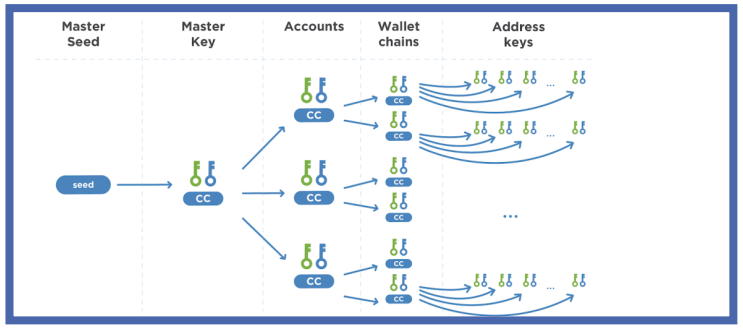


Figure 5-11. Extending a parent public key to create a child public key



https://alexey-shepelev.medium.com/hierarchical-key-generation-fc27560f786

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.”. p.110

Hardened Child Key Derivation - II

- The hardened derivation function uses the parent private key to derive the child chain code, instead of the parent public key.
- This creates a “firewall” in the parent/child sequence, with a chain code that cannot be used to compromise a parent or sibling private key.
- The hardened derivation function looks almost identical to the normal child private key derivation, except that the parent private key is used as input to the hash function, instead of the parent public key

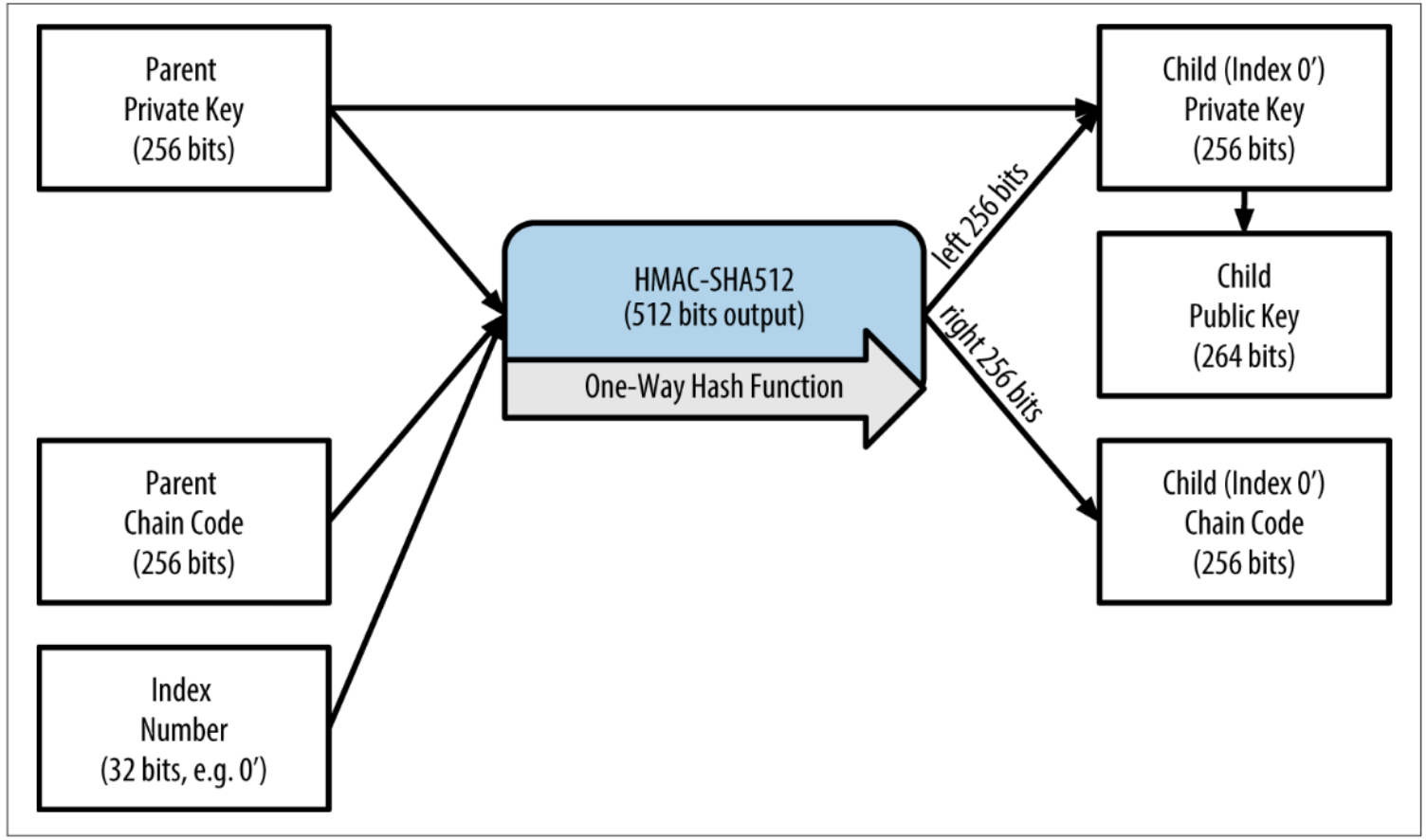


Figure 5-13. Hardened derivation of a child key; omits the parent public key

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.”. p.110

Hardened Child Key Derivation - III

- When the hardened private derivation function is used, the resulting child private key and chain code are completely different from what would result from the normal derivation function.
- The resulting “branch” of keys can be used to produce extended public keys that are not vulnerable, because the chain code they contain cannot be exploited to reveal any private keys.
- Hardened derivation is therefore used to create a “gap” in the tree above the level where extended public keys are used.
- In simple terms, if you want to use the convenience of an xpub to derive branches of public keys, without exposing yourself to the risk of a leaked chain code, you should derive it from a hardened parent, rather than a normal parent.
- As a best practice, the level-1 children of the master keys are always derived through the hardened derivation, to prevent compromise of the master keys.

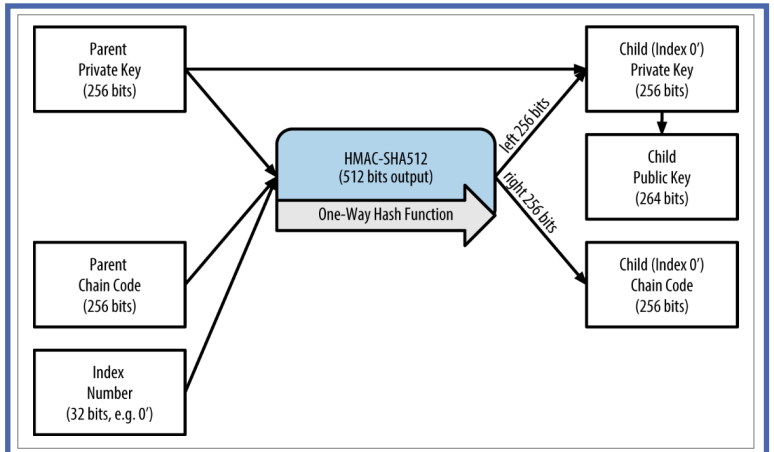
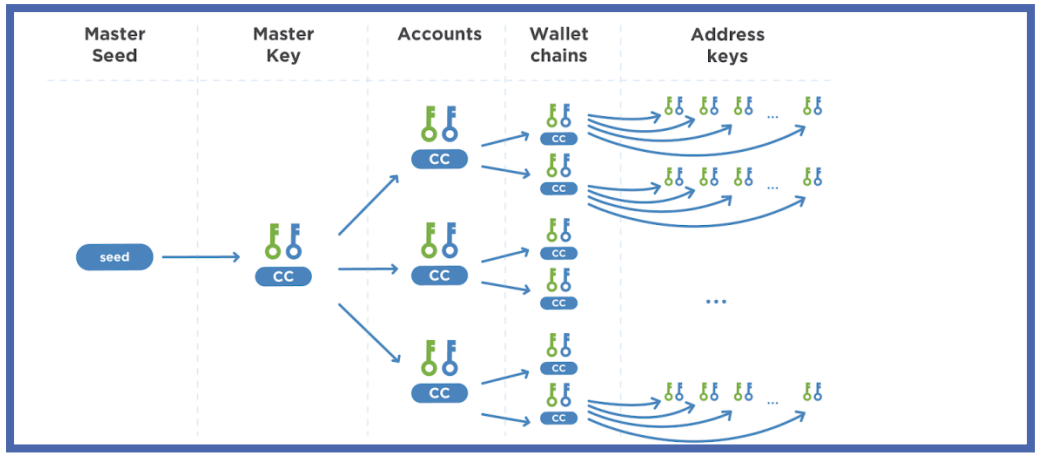


Figure 5-13. Hardened derivation of a child key; omits the parent public key



https://alexey-shepelev.medium.com/hierarchical-key-generation-fc27560f786

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.”. p.110

Hardened Child Key Derivation - Index Numbers

- The index number used in the derivation function is a 32-bit integer.
- To easily distinguish between keys from normal derivation versus hardened derivation,
  - Index numbers between 0 and $2^{31}-1$ are used only for normal derivation.
  - Index numbers between 2^{31} and $2^{32}-1$ are used only for hardened derivation.
- Normal: 0, 1.., hardened: 0', 1'..

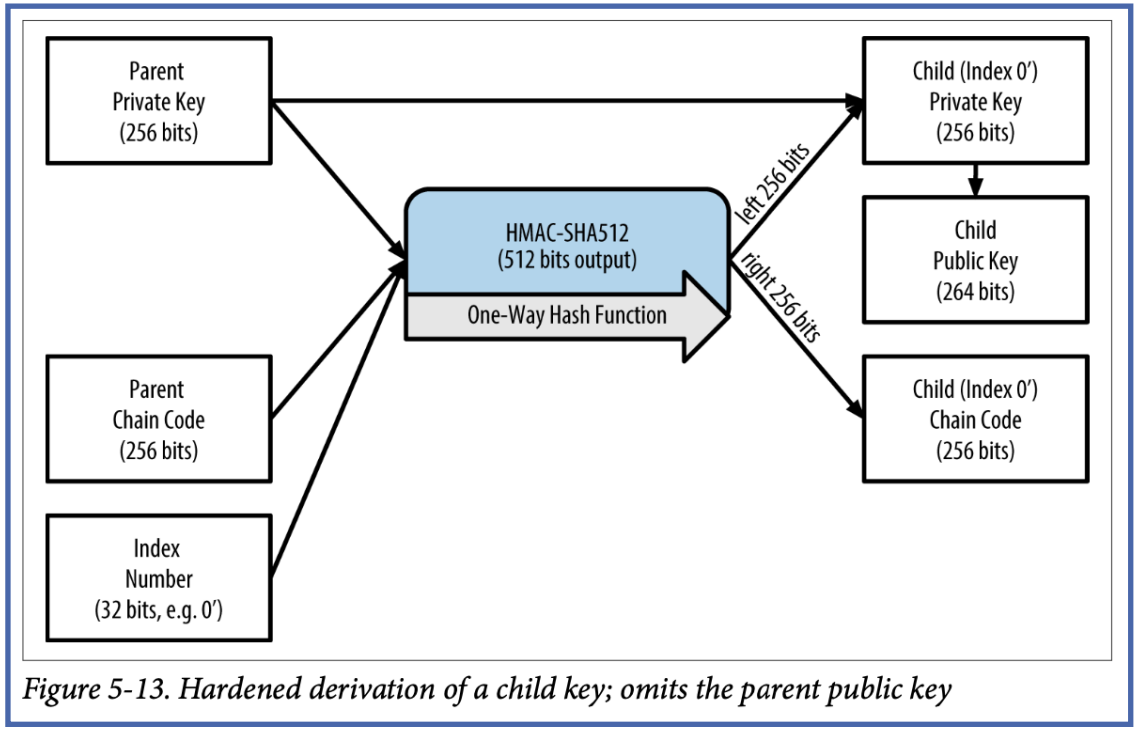


Figure 5-13. Hardened derivation of a child key; omits the parent public key

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.110

Key Derivation Path

Table 5-6. HD wallet path examples

HD path	Key described
m/0	The first (0) child private key from the master private key (m)
m/0/0	The first grandchild private key of the first child (m/0)
m/0'/0	The first normal grandchild of the first *hardened* child (m/0')
m/1/0	The first grandchild private key of the second child (m/1)
M/23/17/0/0	The first great-great-grandchild public key of the first great-grandchild child

BIP-44 specifies the structure as consisting of five predefined tree levels:

```
m / purpose' / coin_type' / account' / change / address_index
```

The first-level "purpose" is always set to `44'`. The second-level "coin_type" specifies the type of cryptocurrency coin, allowing for multicurrency HD wallets where each currency has its own subtree under the second level. There are three currencies defined for now: Bitcoin is m/44'/0', Bitcoin Testnet is m/44'/1', and Litecoin is m/44'/2'.

https://iancoleman.io/bip39/

Table 5-7. BIP-44 HD wallet structure examples

HD path	Key described
M/44'/0'/0'/0/2	The third receiving public key for the primary bitcoin account
M/44'/0'/3'/1/14	The fifteenth change-address public key for the fourth bitcoin account
m/44'/2'/0'/0/1	The second private key in the Litecoin main account, for signing transactions

Antonopoulos, A. M. (2014). Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.". p.110

Splitting and Sharing Keys

Slides based on Bitcoin and Cryptocurrency Technologies: http://bitcoinbook.cs.princeton.edu/

Image by pch.vector on Freepik

CBS

Secret sharing

Idea: split secret into N pieces, such that

given any K pieces, can reconstruct the secret

given fewer than K pieces, don't learn anything

Example: N=2, K=2

P = a large prime

S = secret in [0, P)

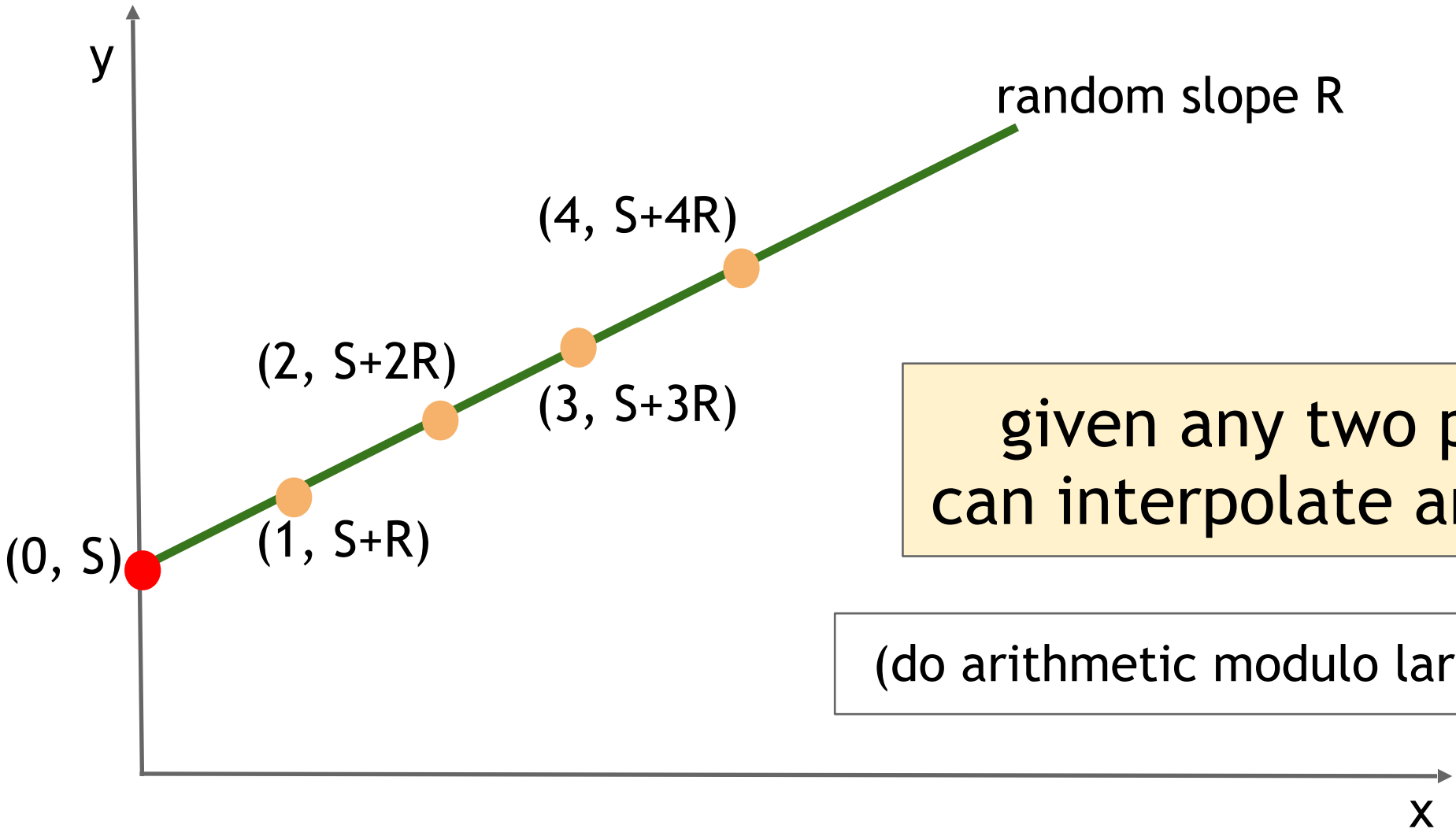
R = random in [0, P)

split:

$X_1 = (S+R) \bmod P$ $X_2 = (S+2R) \bmod P$

reconstruct:

$(2X_1-X_2) \bmod P = S$



given any two points,
can interpolate and find S

(do arithmetic modulo large prime P)

Secret sharing

Equation	Random parameters	Points needed to recover S
$(S + RX) \bmod P$	R	2
$(S + R_1X + R_2X^2) \bmod P$	R_1, R_2	3
$(S + R_1X + R_2X^2 + R_3X^3) \bmod P$	R_1, R_2, R_3	4

etc.

support K-out-of-N splitting,
for any K, N

Secret sharing

Good: Store shares separately, adversary must compromise several shares to get the key.

Bad: To sign, need to bring shares together, reconstruct the key. ⇐ vulnerable

Multi-sig

Recall multi-sig from Lecture 3.

Lets you keep shares apart, approve transaction without reconstructing key at any point.

Example

Andrew, Arvind, Ed, and Joseph are co-workers.
Their company has lots of Bitcoins.

Each of the four generates a key-pair,
puts secret key in a safe, private, offline place.

The company's cold-stored coins use multi-sig, so that three of the four keys must sign to release a coin.

Thank you!

Raghava Mukkamala

rrm.digi@cbs.dk

https://www.cbs.dk/staff/rrmdigi

https://raghavamukkamala.github.io/

https://cbsbda.github.io/