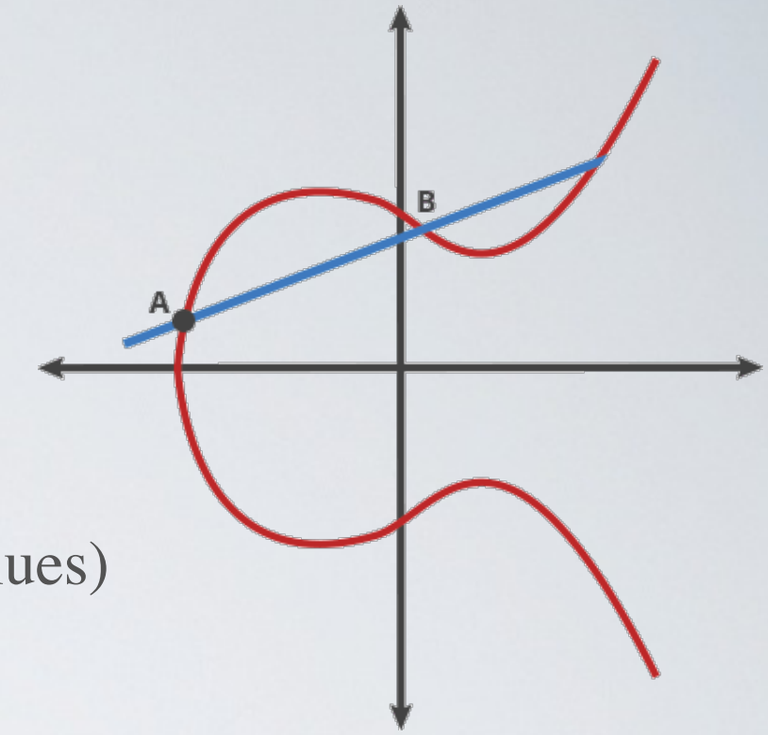
Key Management with Crypto Wallets

Thierry Sans

Elliptic Curve Cryptography



Use Elliptic-curve for generating a cryptographic public-key pair

The algorithm is based on two public pieces:

- The curve equation $y^2 = x^3 + ax + b$ (a and b are fixed values)
- The generator point (fixed value)

➡ See fixed values for secp256: https://en.bitcoin.it/wiki/Secp256k1

When generating a key pair

1. the user "choose a random number" as private key (256 bits for secp256)
2. then derived the public key from the curve

✓ The key size is small compared to RSA : secp256 bits has the same entropy as RSA 3072 bits

✓ Can be used for digital signature (ECDSA algorithm)

✓ Can be used for key agreement (ECDH algorithm)

https://blog.cloudflare.com/a-relatively-easy-to-understand-primer-on-elliptic-curve-cryptography/

The best private key is a random one

- Problem: computers cannot provide "pure" randomness
  - Pseudo-random number generator (based on a seed)
  - True-random number generator that uses data coming from the hardware I/O

✓ Some solution: https://www.youtube.com/watch?v=1cUUfMeOijg

The problem with purely random number

The private key for a crypto wallet must be backed up securely:

- Problem : 256 bits ~ 44 characters (356 bits) encoded in base64

Password-Based Key Derivation Function

- ➡ Either to generate a private key from a password
- ➡ Or generate a (symmetric) encryption key to store the random private key

Example: $\mathbf{PBKDF2}(\mathbf{hashFunction}, \mathit{password}, \mathit{salt}, \mathit{nbIter}, \mathit{kLen})$ that has several round of hashing to mitigate brute force attacks

- ◉ Problem : users tend to choose weak passwords

Seeds and Mnemonic Codes (BIP-39)

- Generate a private key from a mnemonic seed phrase i.e from a sorted series of words in a given wordlist

useful human fox portion peasant grab

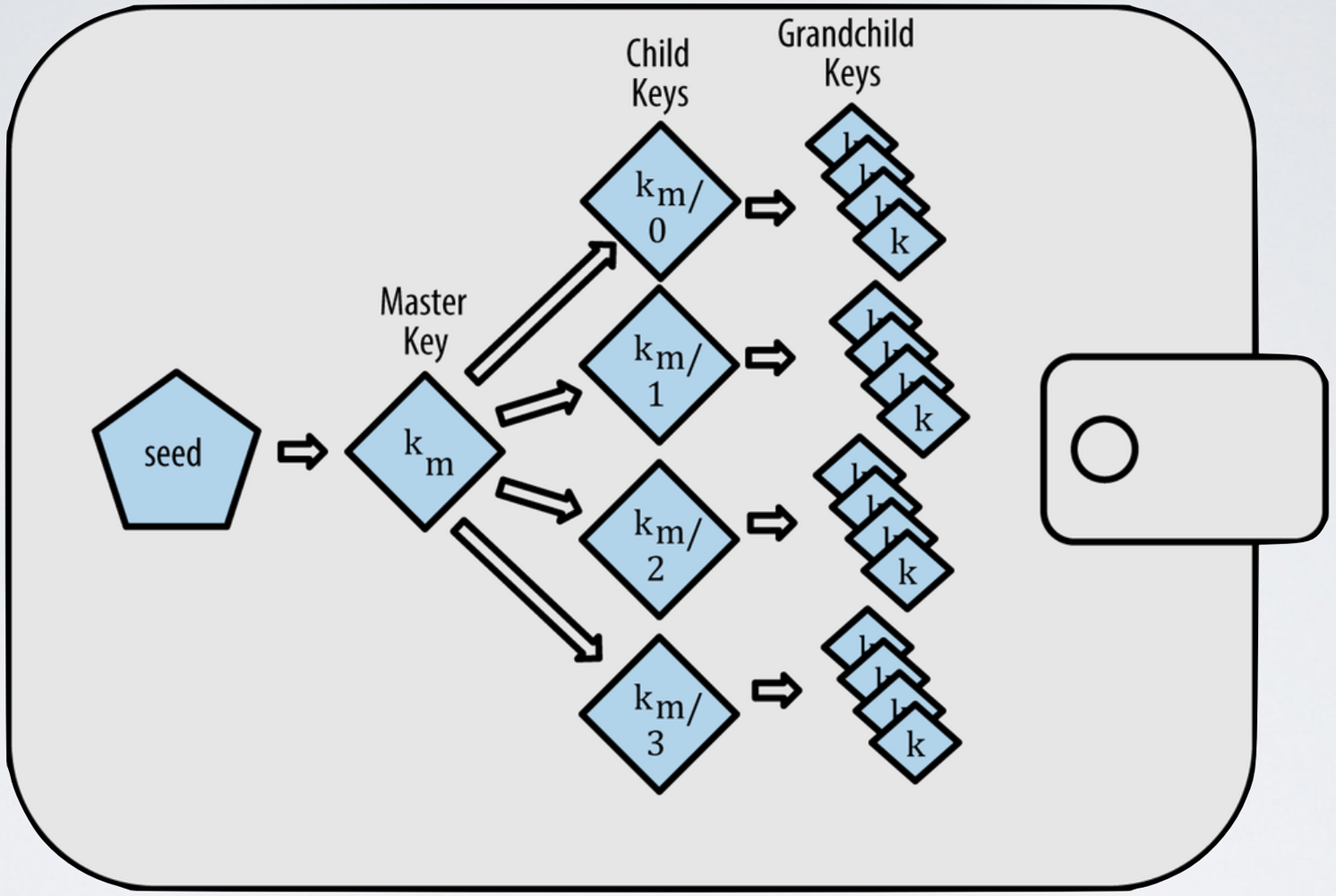
shaft best basic nerve ivory lyrics

Word list are made of 2048 words with the following characteristics to make remembering the sentence much easier:

1. Each word can be unambiguously identified after typing the first four letters
2. Word pairs that look similar are avoided
3. The wordlist is sorted so that lookup for the word index is efficient

- 12-word mnemonic ~ 128 bits of entropy
- 24-word mnemonic ~ 256 bits of entropy

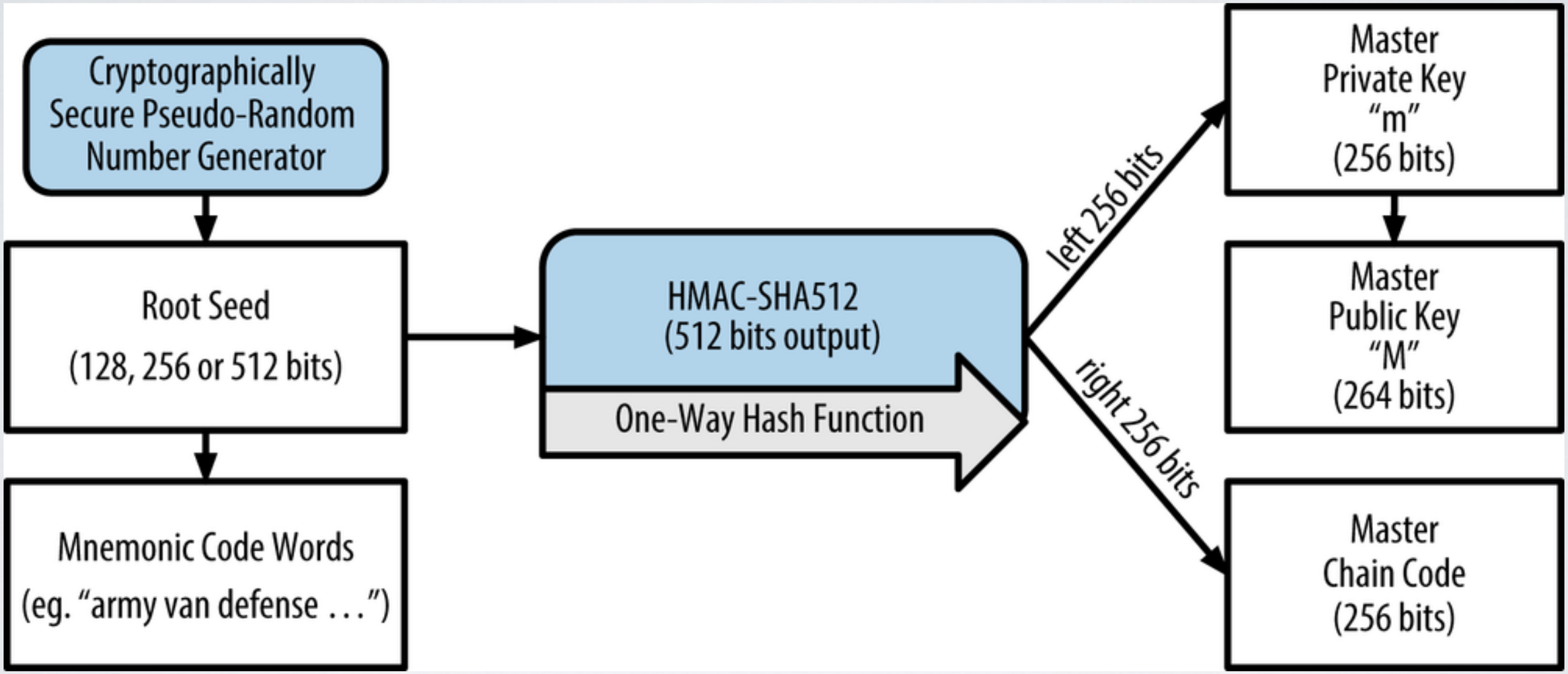
Hierarchical Deterministic Keys (BIP 32)



- Each node in the tree can have a maximum of 2^{32} (4,294,967,296) child nodes
- The depth of the tree is potentially infinite

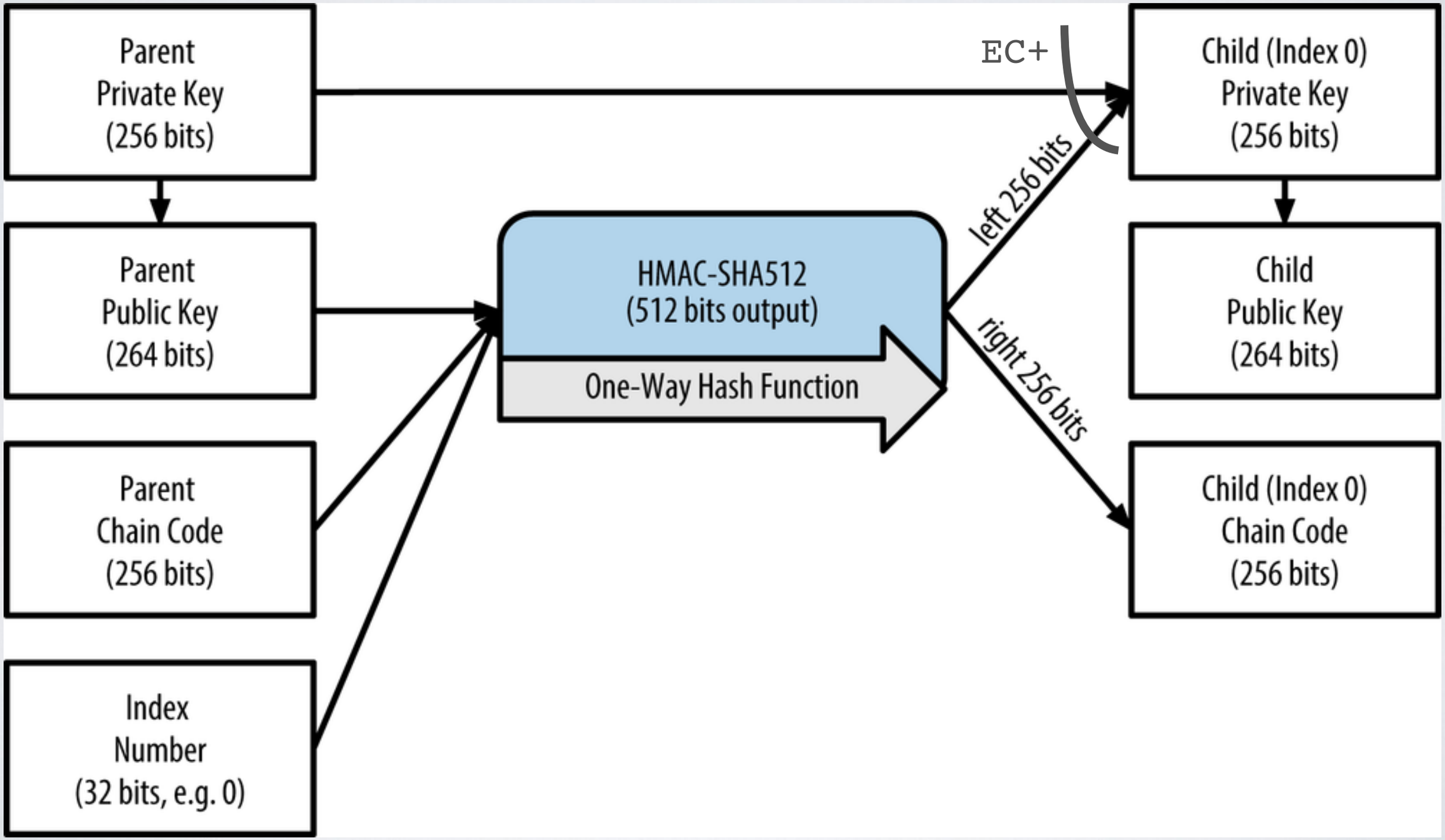
https://www.oreilly.com/library/view/mastering-bitcoin/9781491902639/ch04.html

Creating master keys and chain code from a root seed

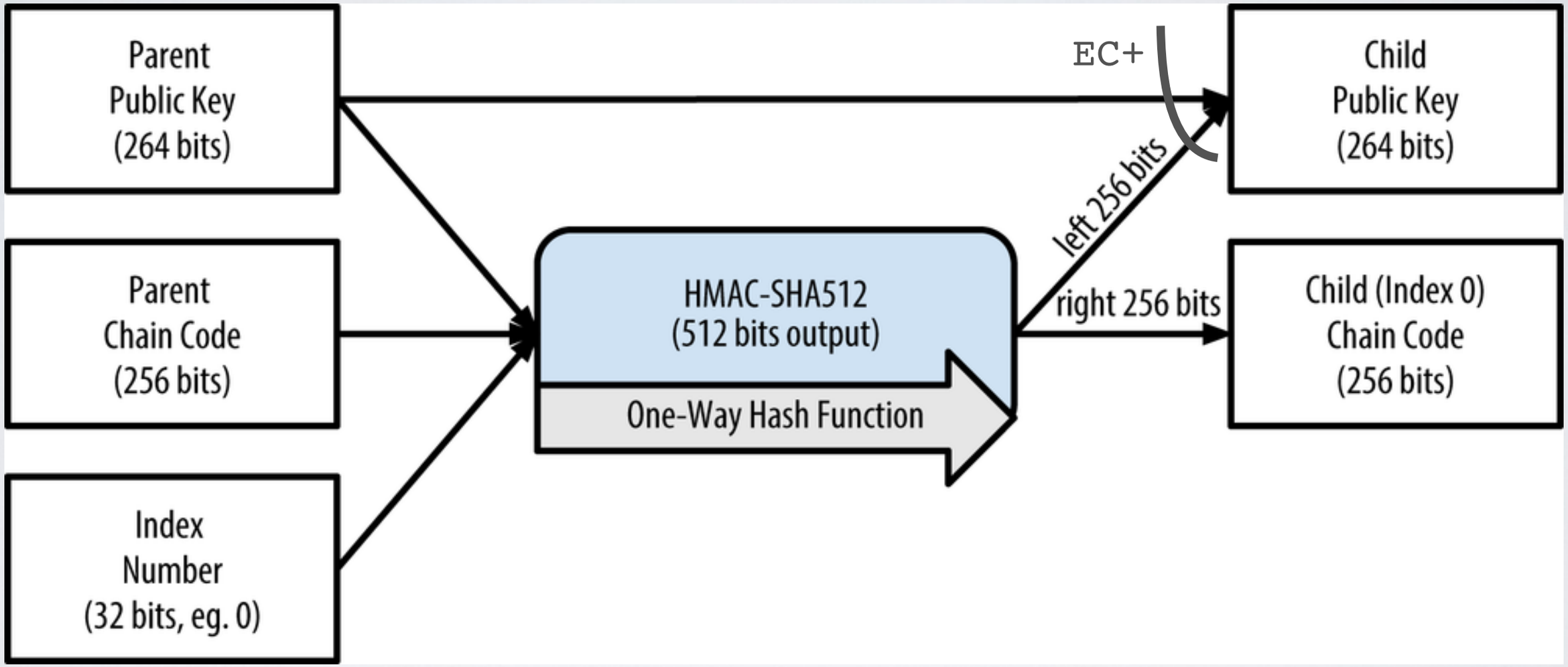


Child Key Derivation (CKD)

Extending parent private key to derive child private key



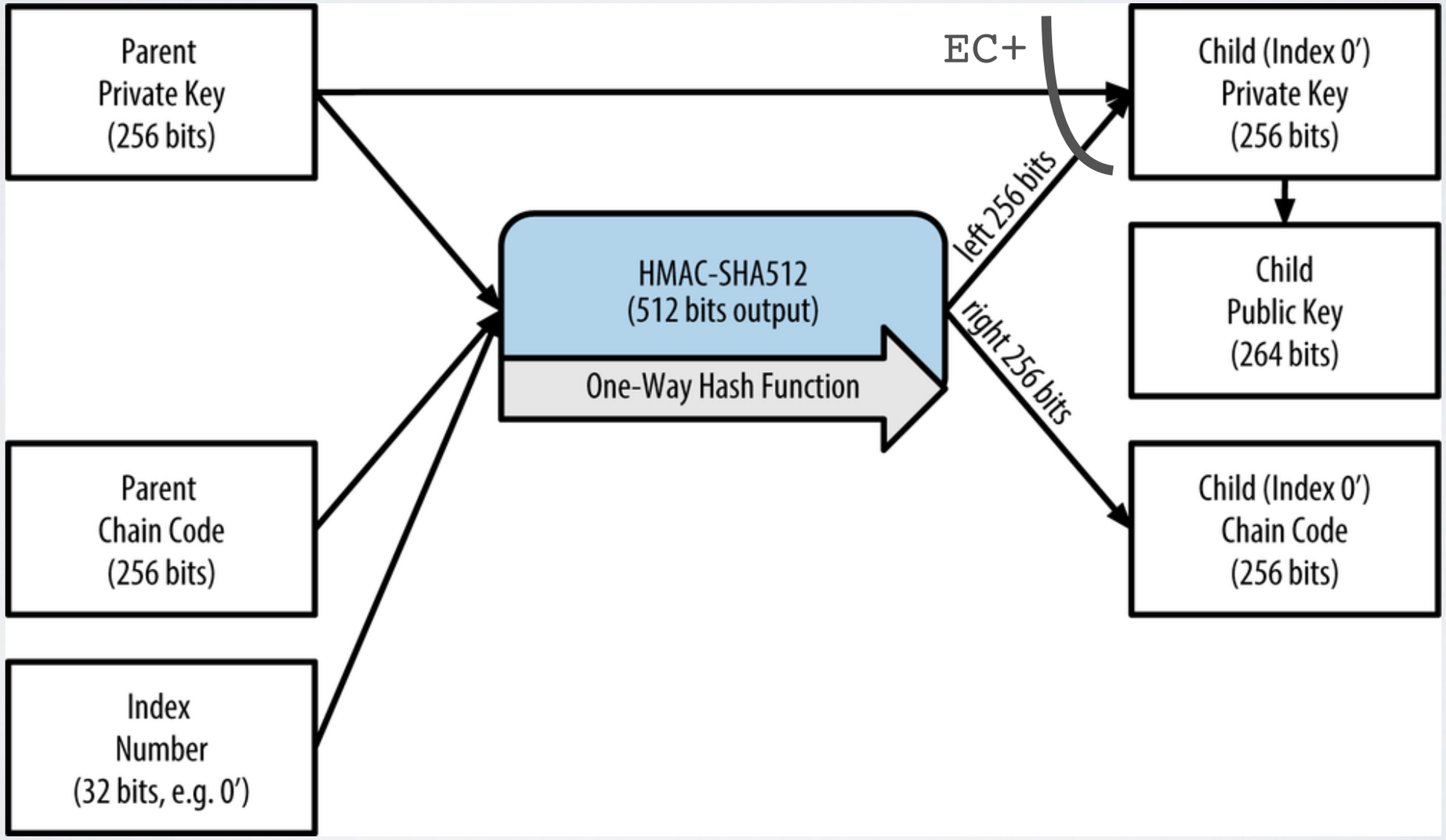
Child Key Derivation (CKD)
Extending parent public key to derive child public key



https://medium.com/@robbiehanson15/the-math-behind-bip-32-child-key-derivation-7d85f61a6681

Hardened Key Derivation

- ✓ Access to parent's public key does not to child private key
- ◉ However, access to a child private key can be used with the chain code to derive all the other child private keys
- ➡ Hardened derivation uses the parent private key to derive the child chain code, instead of the parent public key



https://medium.com/@blainemalone01/hd-wallets-why-hardened-derivation-matters-89efcdc71671

Hierarchical Deterministic Key Identifier

Path naming convention

- each level of the tree separated by a `/` character
- hardened keys are marked with `'` character
- derived private keys start with `m` character
- derived public keys start with `M` character

HD path	Key
`m/0`	The first (`0`) child private key from the master private key (`m`)
`m/0/0`	The first child 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 of the 18th grandchild of the 24th child

HD Wallets (BIP-44)

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

HD path	Key
`M/44'/60'/0'/0/0`	The receiving address public for the first Ethereum account
`M/44'/0'/3'/1/14`	The 15th change-address for the 4th Bitcoin account
`m/44'/2'/0'/0/1`	The second private key in the Litecoin main account
`m/44'/1'/1'/0/0`	The second account on Testnet (all coins)

Registered coin types : https://github.com/satoshilabs/slips/blob/master/slip-0044.md