MASTER BITCOIN

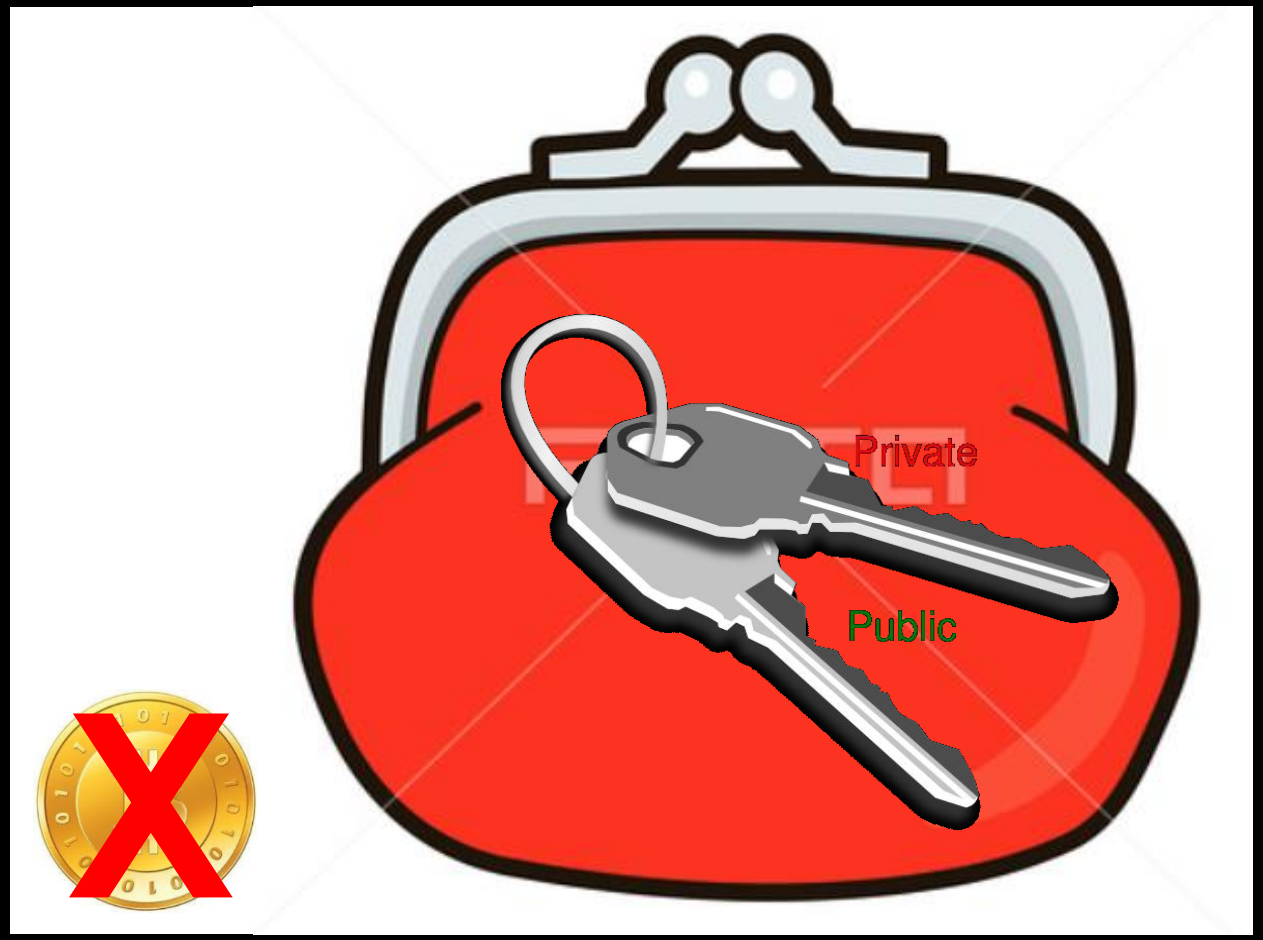
Chapter5 - Wallet

Cheng Ting Tsai

2018.08.09

Wallet

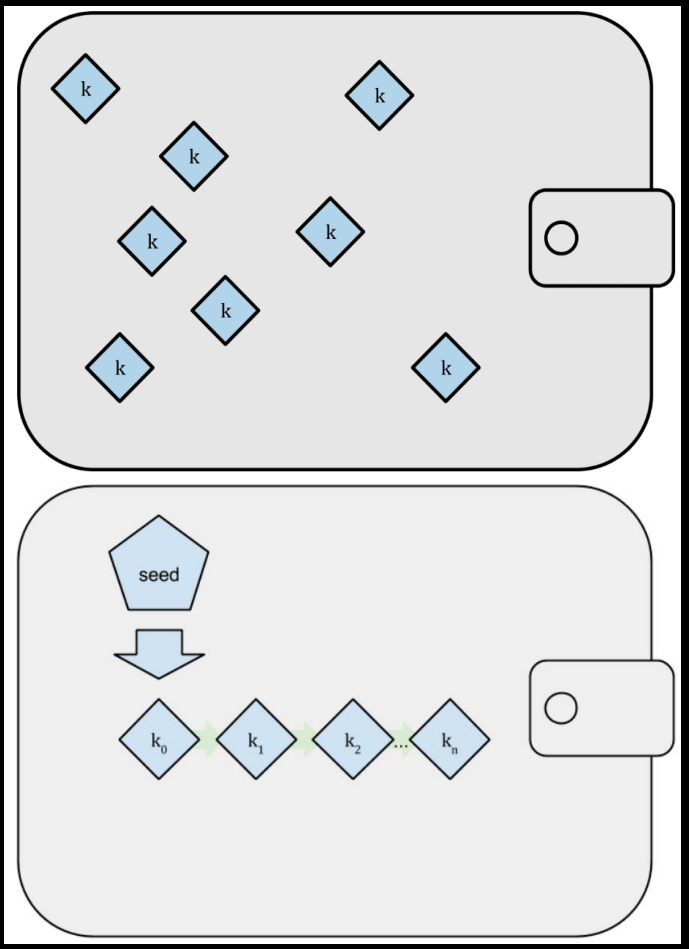
- Wallet stores
  - Key pairs
- Wallet **DO NOT** store
  - Coins (transactions)



Types

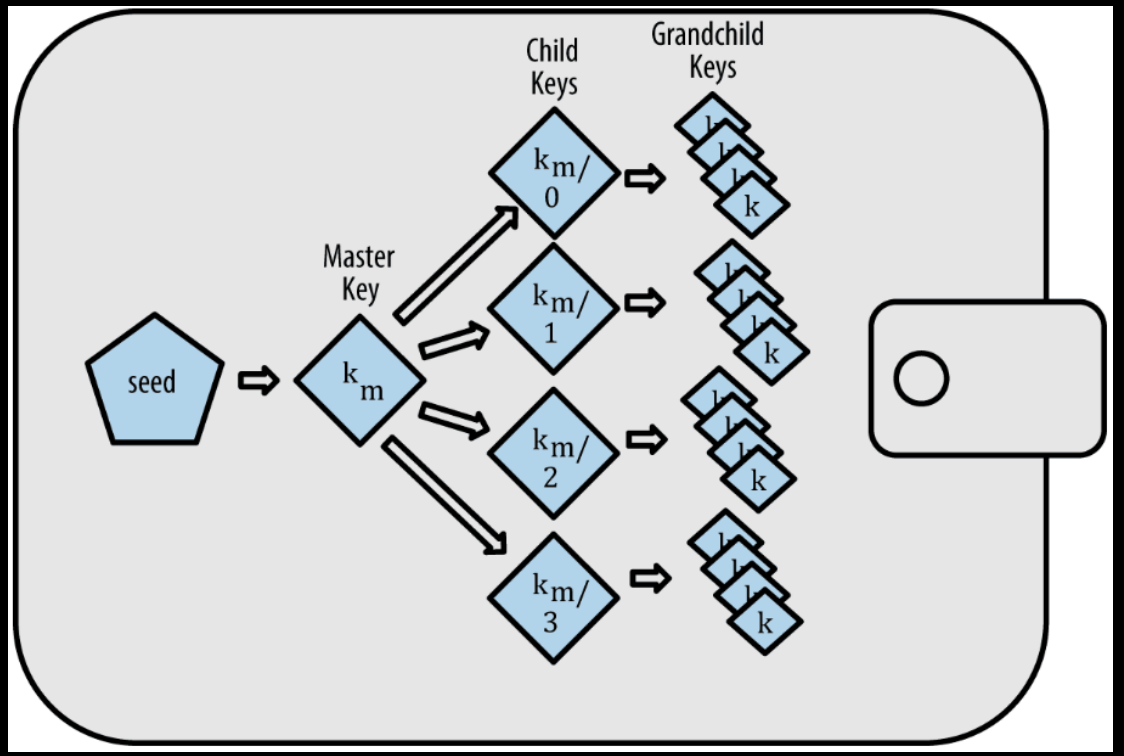
- Nondeterministic wallet
  - Random
  - Independent
  - Just a Bunch Of Keys (JBOK)

- Deterministic wallet
  - Seed
  - Easy for key restoration



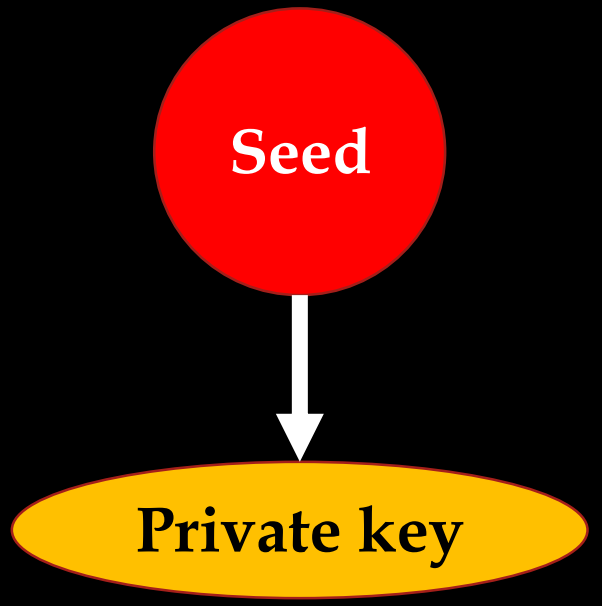
HD Wallets (1/7)

- Data structure
  - Tree
- Root
  - Seed
- Keys production
  - Child key derivation (CKD)



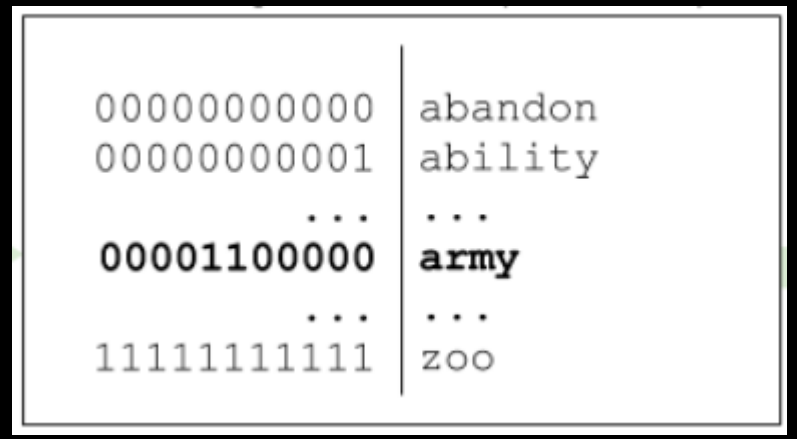
HD Wallets (2/7)

- Seed
  - 512 bits
  - Password-Based Key Derivation Function 2 (PBKDF2)
  - Used to generate the first private key
- Mnemonic Code Words
  - Bitcoin Improvement Proposals 39 (BIP-39)
  - Easily for user to memorize



HD Wallets (3/7)

- Bitcoin Improvement Proposals 39 (BIP-39)
  - Generate an entropy (128 or 256 bits)
  - Hash the entropy (SHA256)
  - Append first n bits of hash to entropy
    - n = 4 if length of entropy = 128 bits
    - n = 8 if length of entropy = 256 bits
  - Divide sequence to m parts, n bits for each (m = 12 or 24)
  - Map each parts to a word from the predefined dictionary of 2048 words



00000000000	abandon
00000000001	ability
...	...
00001100000	**army**
...	...
11111111111	zoo

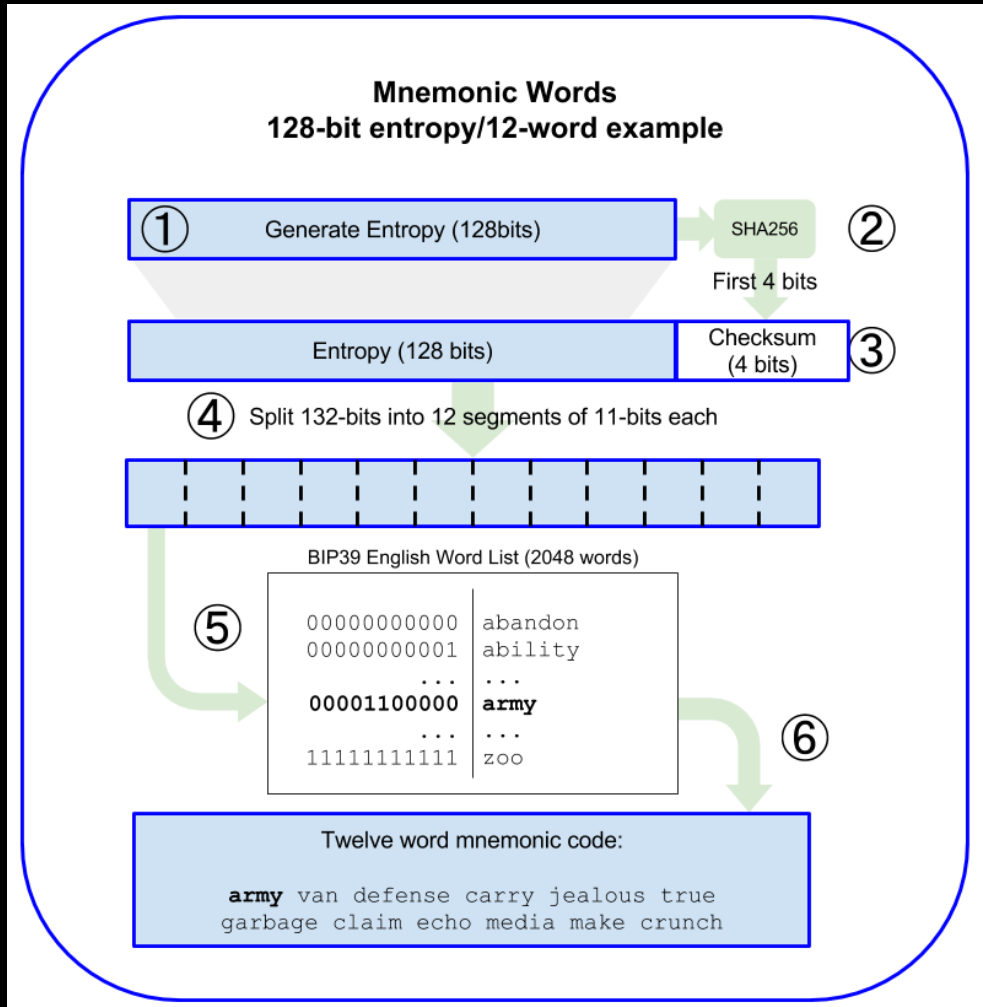
HD Wallets (4/7)

Table 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

HD Wallets (5/7)

- Example
  - E = entropy(128)
  - H = sha_256(E)
  - S = (E | H[first 4 bits])
  - P[] = S to 12 parts, 11 bits each
  - MCW = Map(P[], Dictionary)

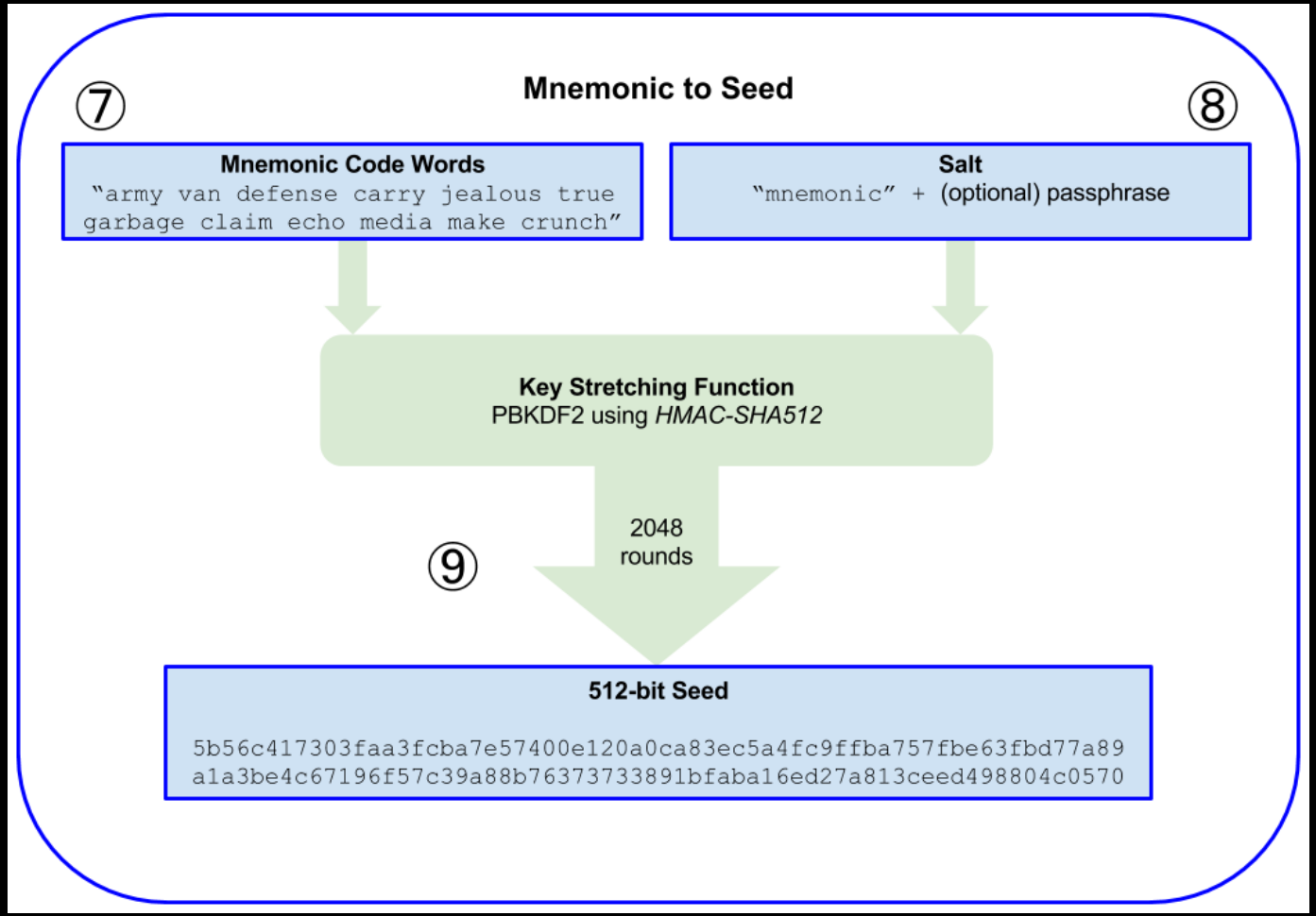


HD Wallets (6/7)

- Seed
  - Mnemonic Code Words | Salt (optional)
  - PBKDF2: HMAC-SHA512

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

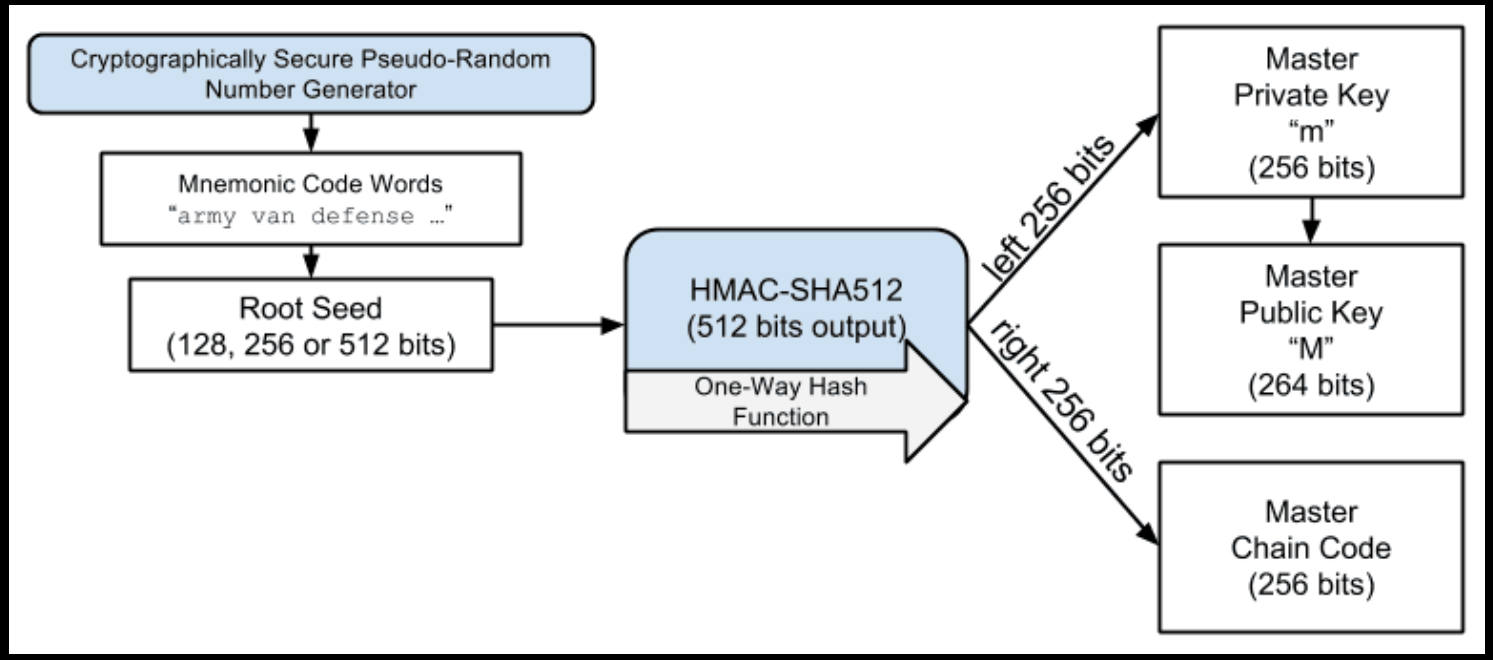
HD Wallets (7/7)



Short Break ~ =u=/

Keys (1/11)

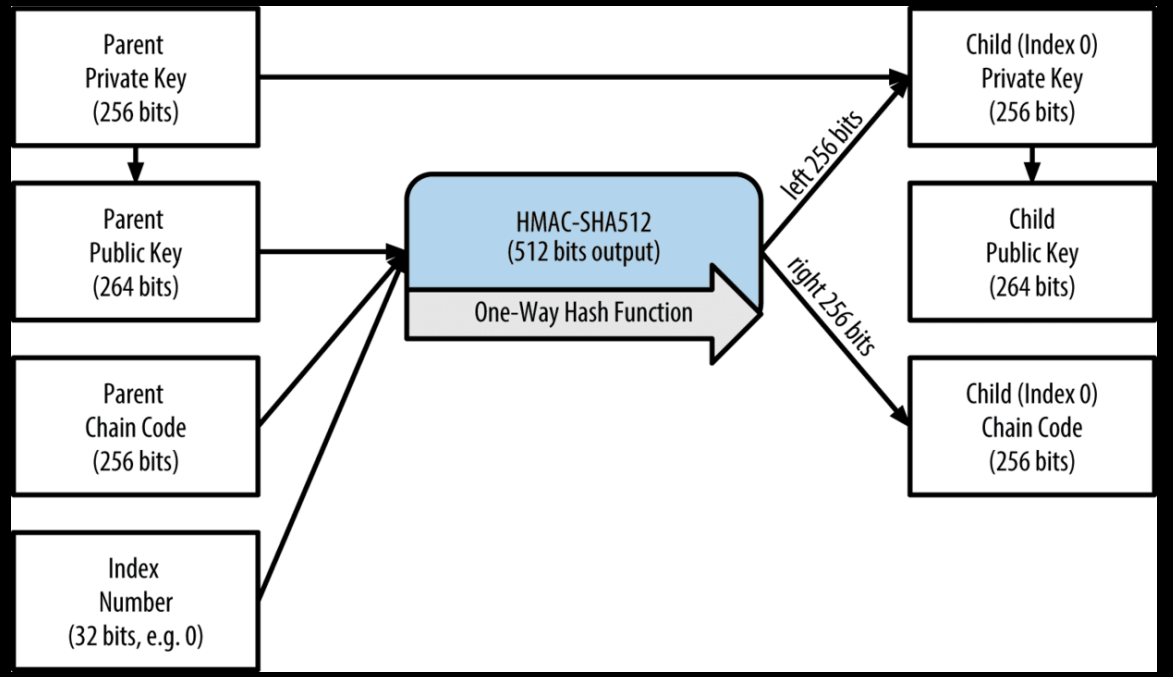
- First private key (m)
  - Seed
  - HMAC-SHA512
  - Left 256 bits
- Chain code
  - Right 256 bits
  - Entropy for child keys
- Public key (M) = m*G



Keys (2/11)

- Child private keys
  - Parameters
    - Parent public key
    - Parent chain code
    - Child key index
    - Parent private key (optional)

Hash by HMAC-SHA512

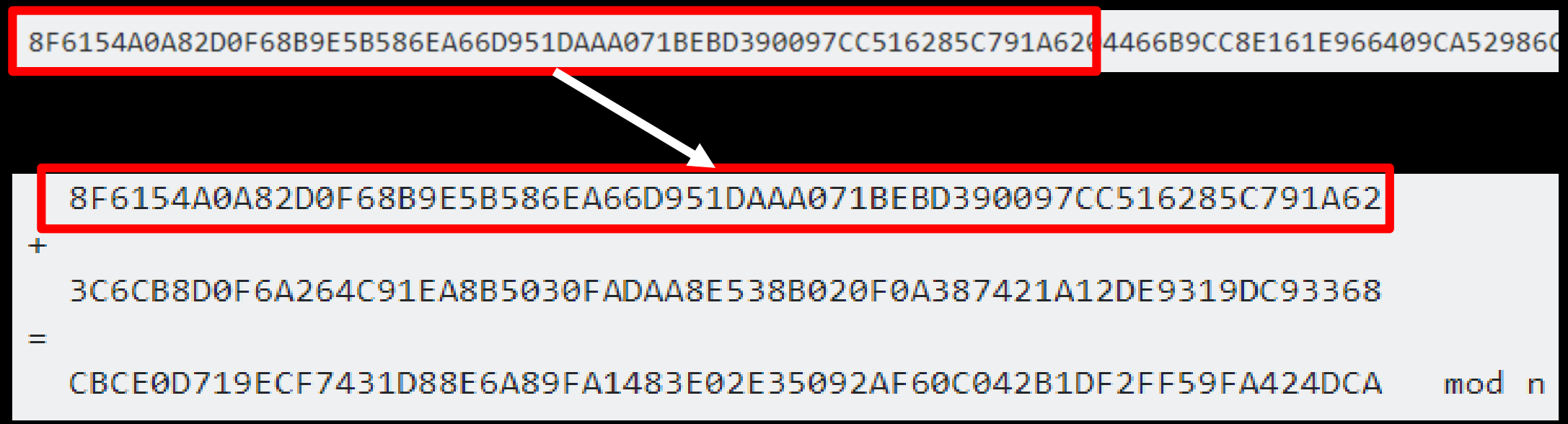


- **Child private key = Left 256 bits of hash + parent private key**

Keys (3/11)

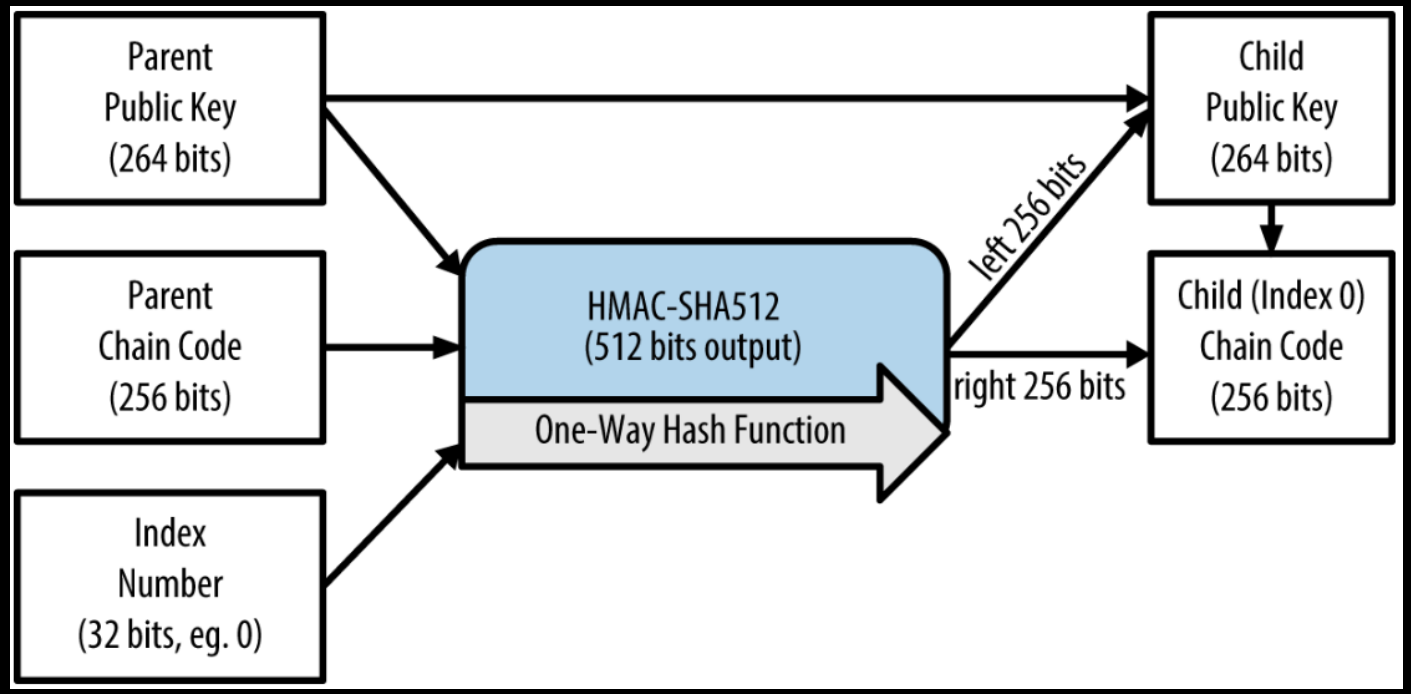
- Child private key = Left 256 bits + parent private key

HMAC-SHA512



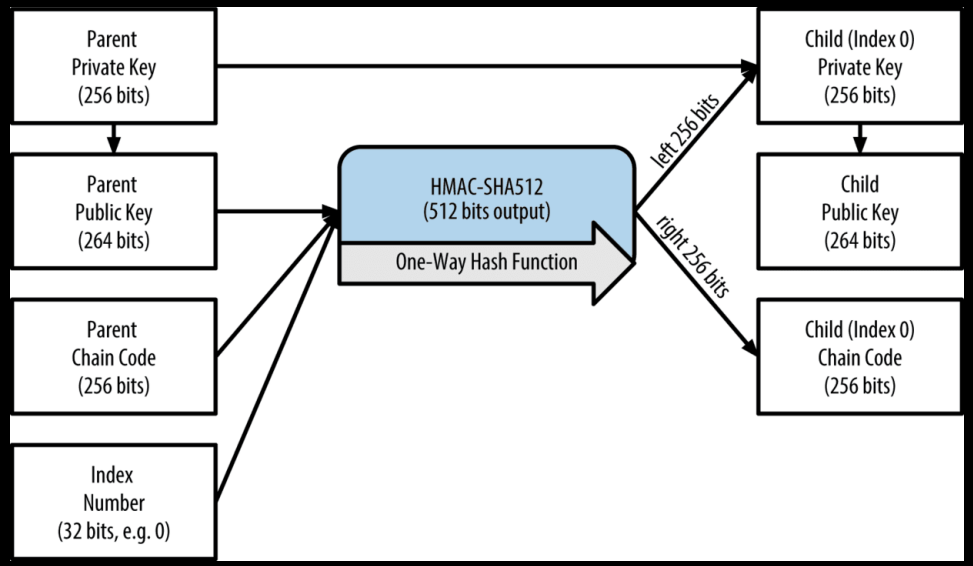
Keys (4/11)

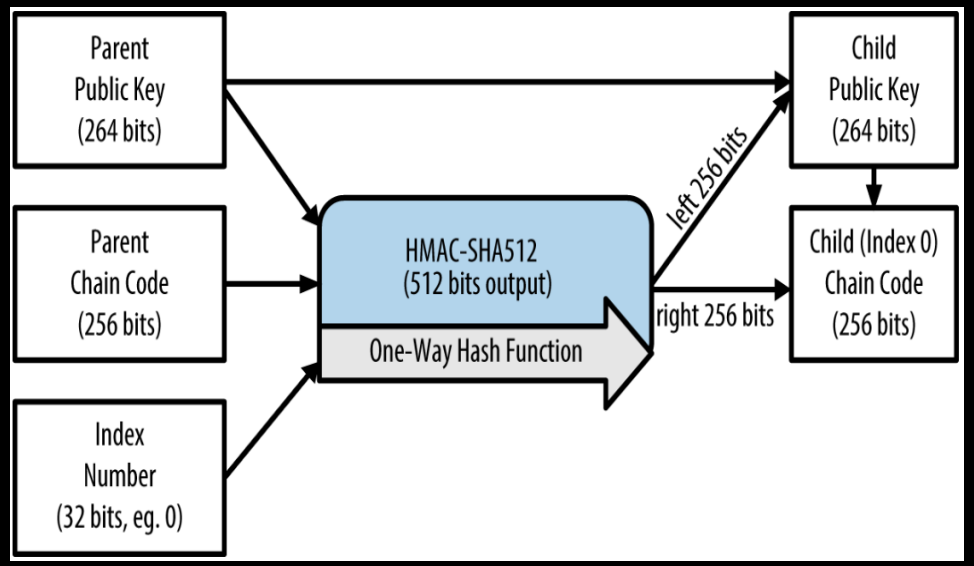
- Child public keys only
  - Parameters
    - Parent public key
    - Parent chain code
    - Child key index
  - HMAC-SHA512
- **Safety**
  - Can receive
  - Can't pay



Keys (5/11)

- Index
  - $0 \sim 2^{31} - 1$ (0x00000000 to 0x7FFFFFFF)
  - The other half of index is for a more advanced usage.





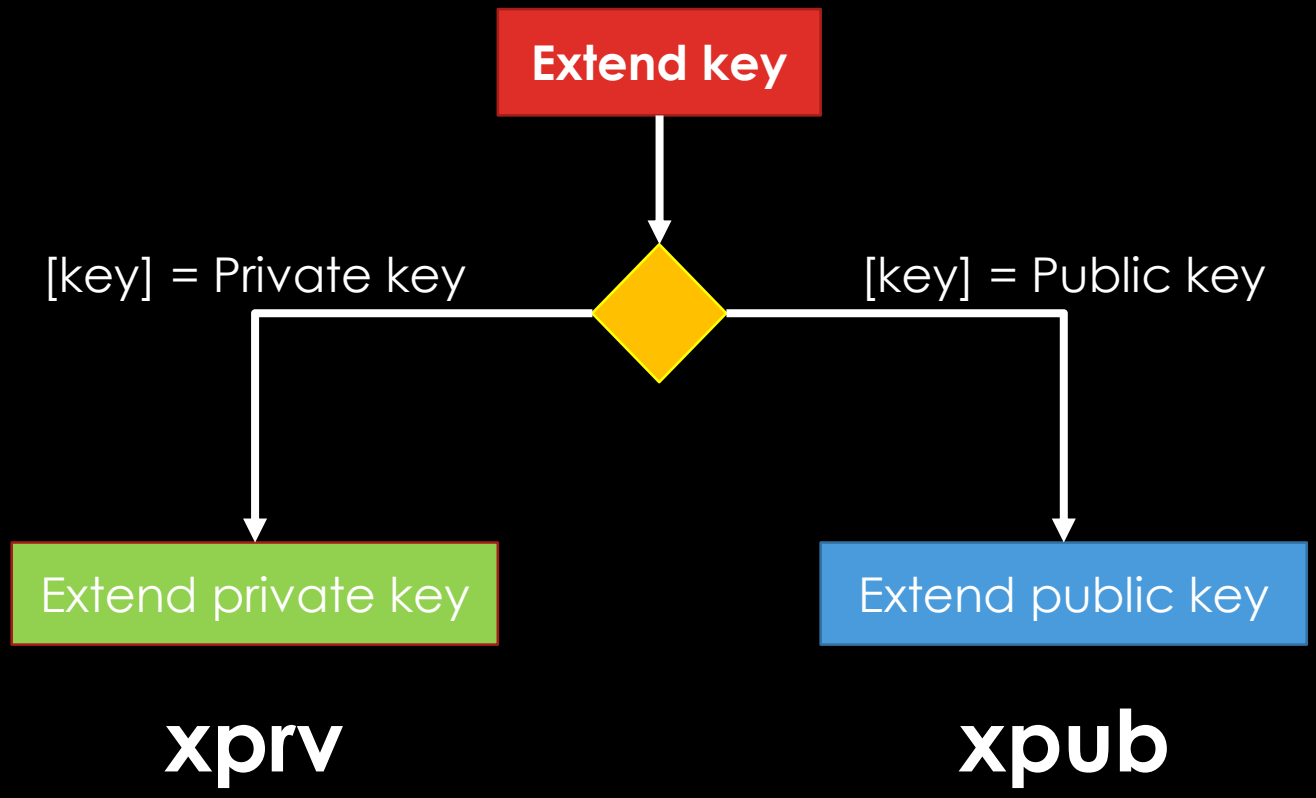
first 4 bytes of the `hash160` of the *public key*

Keys (6/11)

- Extended key `[ magic ][ depth ][ parent fingerprint ][ key index ][ chain code ][ key ]`
  - **Magic:** mainnet or testnet
  - **Depth:** depth in tree
  - **Parent fingerprint:** first 4 bytes hash160 public key
  - **Key index:** key index
  - **Chain code:** chain code
  - **Key:** private key or public key

- Easy to use
  - Combine key features that used for creating child key

Keys (7/11)


Extend key
[key] = Private key
[key] = Public key
Extend private key
Extend public key
xprv
xpub

Keys (8/11)

- Another problem comes out!
  - If one gets
    - Parent xpub (contains public key and chain code)
    - At least one child private key
  - One can get all the children information
  - One can also get a chance to reach the parent private key
  - Because
    - **Child private key = Left 256 bits of hash + parent private key**
  - **Chain code + (private key) can do everything!!**

Keys (9/11)

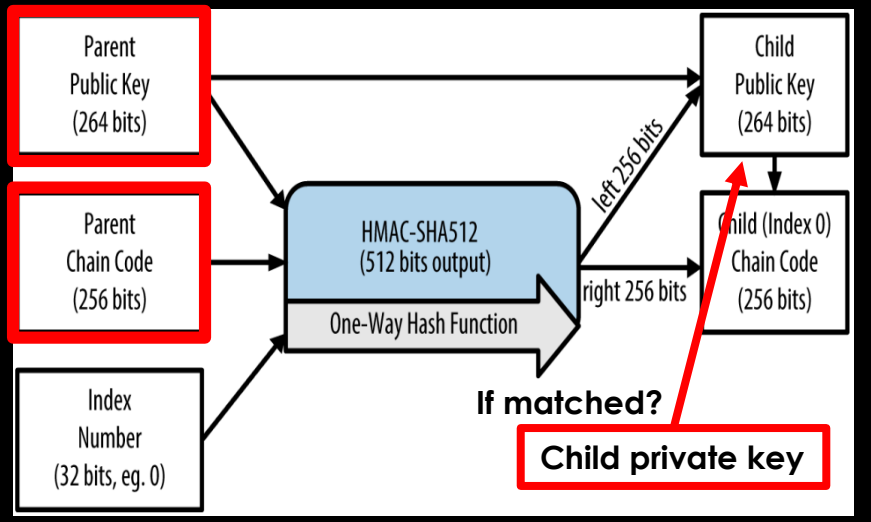
xpub	: public extended key
P	: public key
p	: private key
C	: chain code
N	: child public key
n	: child private key
I	: index of child

- Find p given xpub, n:
  - Get P and C from xpub
  - Use P and C to find child public key N with index i

  - Try different i above steps until N fulfills N = n*G
  - If N = n*G, then i is found

  - Use i, P, C to find 512 hash code
  - n - Left 256 of hash = p
  - FOUND p
  - Then p, P, C can be used to find every children

Unknown
known



Keys (10/11)

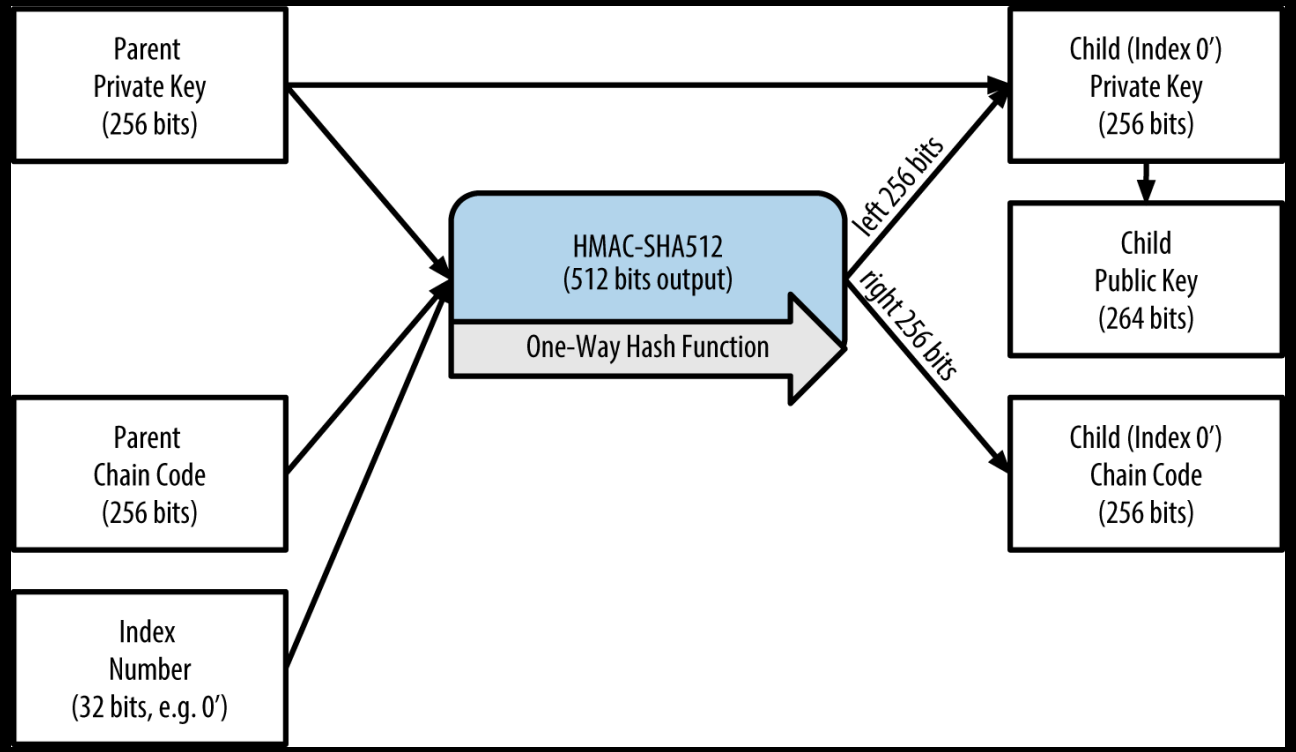
- Hardened child key derivation

- Parent public key
- Parent chain code
- Child key index

Hash by HMAC-SHA512

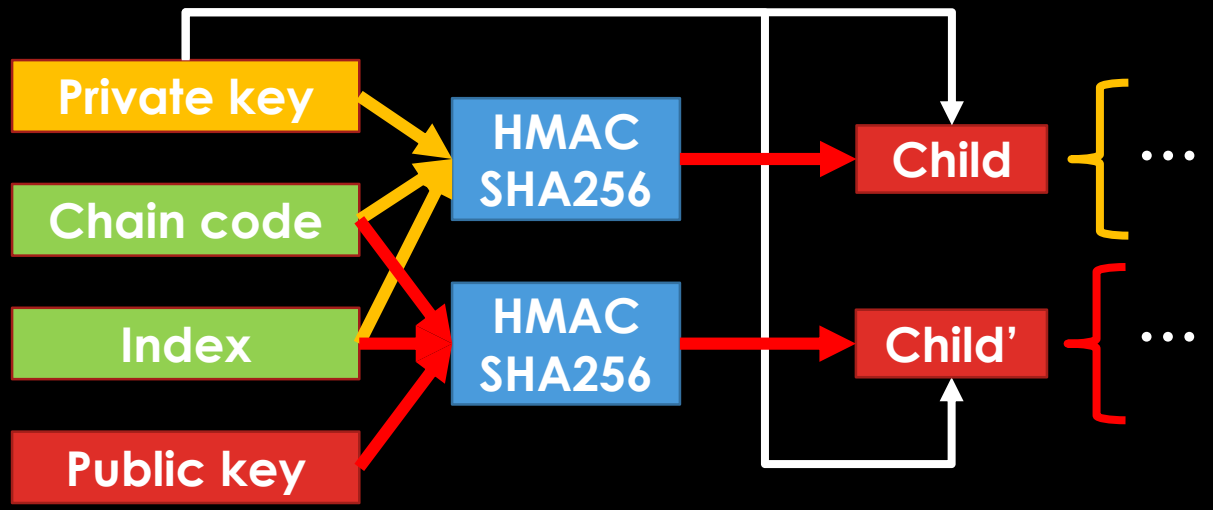
- Parent private key
- Parent chain code
- Child key index

Hash by HMAC-SHA512



Keys (11/11)

- Hardened child key derivation
  - Safe
  - Private key is used
  - Gap
- Index
  - $2^{31} \sim 2^{32} - 1$
  - 0x80000000 to 0xFFFFFFFF
  - $0' \sim (2^{31}-1)'$
  - $n' = 2^{31} + n$



HD Wallet Key Identifier (Path)

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 from the first child (m/0)
m/0'/0	The first normal grandchild from the first *hardened* child (m/0')
m/1/0	The first grandchild private key from the second child (m/1)
M/23/17/0/0	The first great-great-grandchild public key from the first great-grandchild from the 18th grandchild from the 24th child

Navigating HD Wallet Structure

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

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

Thanks For Your Attention!!