How Do Public/Private Keys Work in Bitcoin?

Stacie

Back End Engineering

Previous roles: Enterprise Blockchain, AdTech

@satsie_

Ron

Head of Security

C4 CCSSA Exam Curator

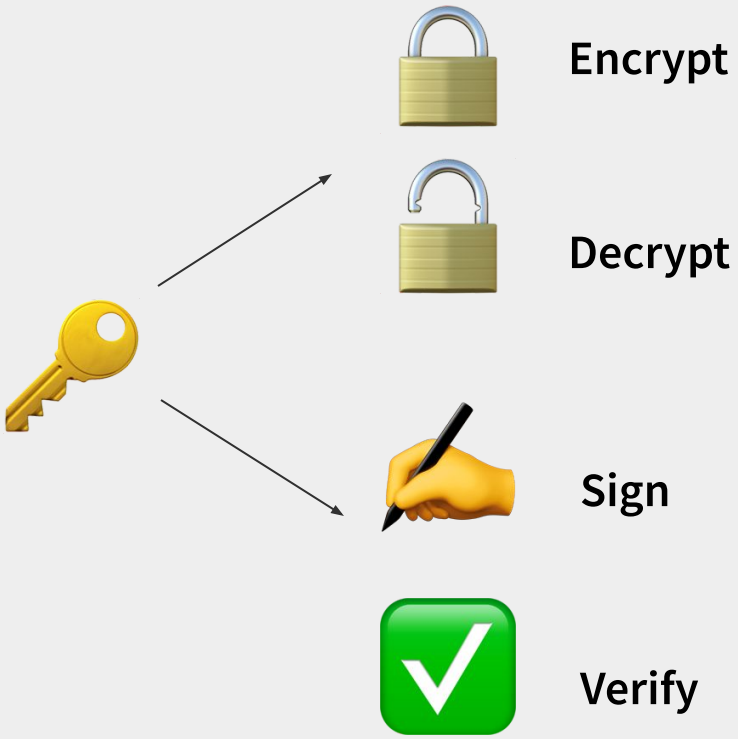
@forwardsecrecy

Cryptography Basics

What are keys?

- A random string of bits
  - MFK4EEACIDAwT9nORmlUb7NZv76Z
    5dYVbX/o9Yzf…
- Used to
  - Encrypt & decrypt messages
  - Create & verify digital signatures
  - And more!

Symmetric vs. Asymmetric Keys



Symmetric: one key

- Passwords / Shared Secret

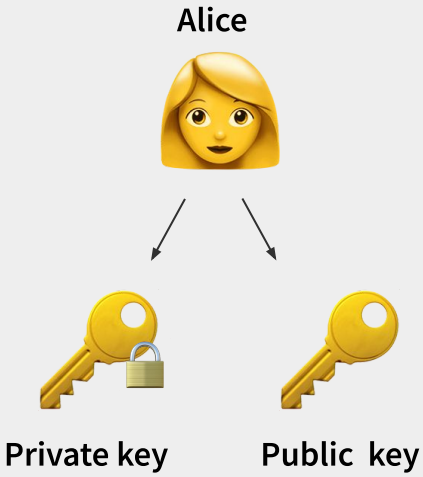
Symmetric vs. Asymmetric Keys

Asymmetric: Separate keys

- Public-key cryptography (keypair)
- **Private key**: Secret. Only you know it!
  - Create signatures, decrypt data.
- **Public key**: The one you share.
  - Verify signatures, encrypt data.

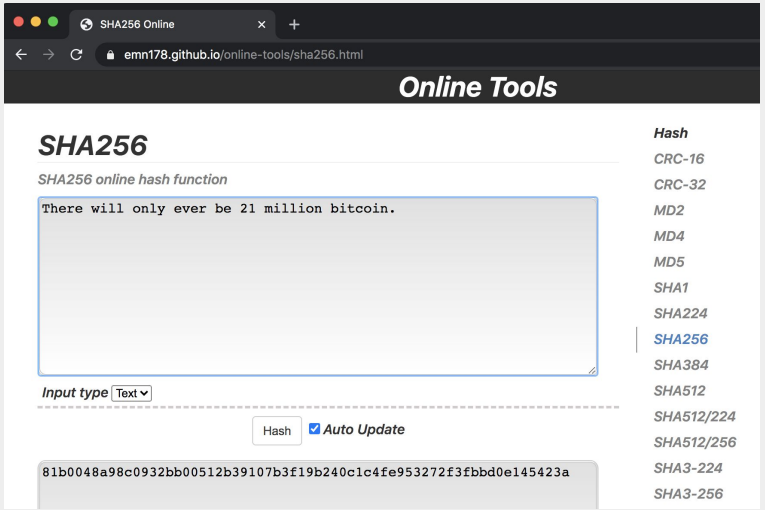
✔️ Private key -> Public key

❌ Public key -> Private key



Hashing

One way function that produces a fingerprint (hash) of a piece of data.




```
stacie@Stacies-MBP ~ $ echo -n "There will only ever be 21 million bitcoin." | shasum -a 256
81b0048a98c0932bb00512b39107b3f19b240c1c4fe953272f3fbbd0e145423a  -
```

What IS a digital signature?

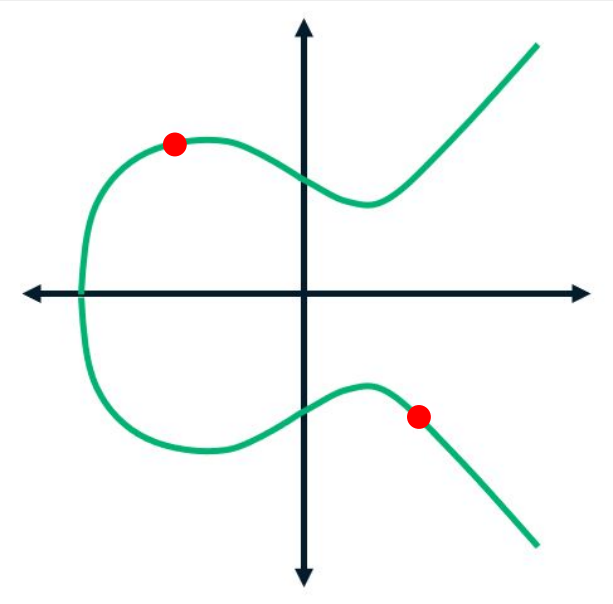
- Proves that message or document was not tampered with
- Contents are hashed (**HashA**)
- Hash is encrypted with sender private key
- Content + Encrypted hash (**HashA**) sent to recipient
- Recipient generates own content hash (**HashB**)
- Recipient decrypts sender hash (**HashA**) using sender public key
- **HashA** == **HashB** 

Common Encryption Schemes and Hashing Algorithms

- **DES**: Symmetric, old, unsecure.
- **AES**: Symmetric, highly secure, commonly used. Encrypts blocks of data with multiple rounds.
- **RSA**: Asymmetric, very secure & popular. Based on factoring the product of two very large prime numbers.
- **SHA (Hashing)**
  - Secure Hash Algorithm
  - SHA1 vs SHA256

How does all this work in Bitcoin?

Bitcoin Uses ECDSA



- **ECDSA/ECC** (Elliptic Curve Digital Signature Algorithm/Elliptic Curve Cryptography)
- Does NOT encrypt
- Based on elliptic curves - the set of points that satisfy
  - $y^2 = x^3 + ax + b$
- Same level of security as RSA, but with shorter key lengths
- Shorter keys = less bandwidth, storage, and processing power

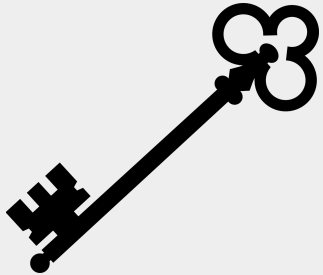
Don’t get too attached to ECDSA!

Taproot (BIP341) upgrade includes Schnorr Signatures (BIP340)

Provably Secure w/ Proofs
Discrete Log Less Assumptions
Simpler and Smaller Footprint
64 byte signatures vs. 71 bytes
Patented until 2008
Bitcoin will be one of the first
More privacy with signature aggregation

What does it mean to own Bitcoin?

- Bitcoin addresses = how you plan to spend the bitcoin
  - public key or
  - by script
- Spending bitcoin = signing a message
  - Message: *“I am transferring ownership of this bitcoin to someone else’s address (public key/script)”*
- Owning bitcoin = having the private key(s) that correspond to the address
- “Not your keys, not your coins”
- Freedom and Sovereignty
- Security Awareness

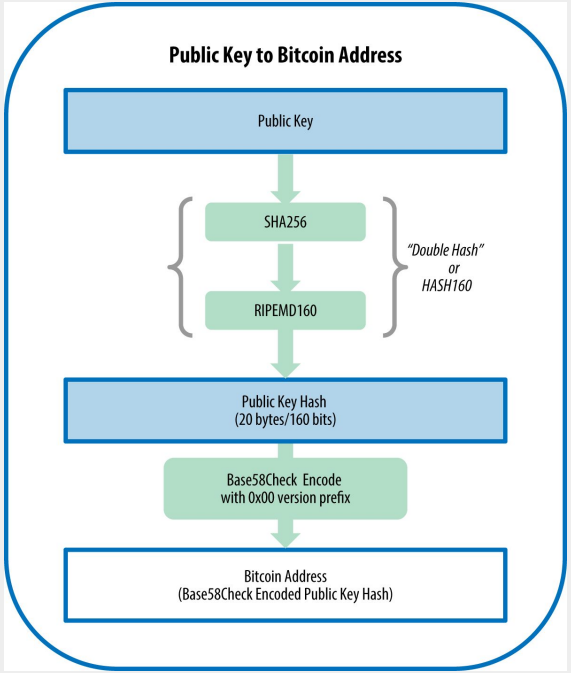


Key Generation Demo

Try it at home!

```
bitcoin-cli getnewaddress
bitcoin-cli dumpprivkey <address>
```

How do we go from a keypair to a Bitcoin address?



Address derivation for non-segwit addresses
Source: Mastering Bitcoin by Andreas Antonopoulos, Chapter 4

- Addresses = hashes
- Input: public key or redeem script
- If we didn't hash, addresses would be very long!
- Can't go backwards and derive a public key from an address
- Final step: convert to Base58Check/Bech32 (segwit) encoding
  - Human readable portion
  - Base58/Bech32 encoding of the hash
  - Checksum

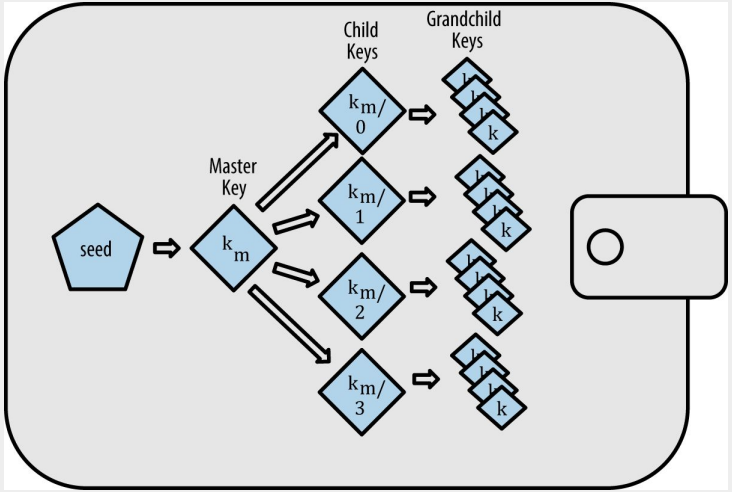
Different Address Types

Example use	Leading symbol(s)	Example
Pubkey hash (P2PKH address)	1	17VZNX1SN5NtKa8UQFxwQbFeFc3iqRYhem
Script hash (P2SH address)	3	3EktnHQD7RiAE6uzMj2ZifT9YgRrkSgzQX
Private key (WIF, uncompressed pubkey)	5	5Hwgr3u458GLafKBgxtssHSPqJnYoGrSzgQsPwLFhLNYskDPyyA
Private key (WIF, compressed pubkey)	K *or* L	L1aW4aubDFB7yfras2S1mN3bqg9nwySY8nkoLmJebSLD5BWv3ENZ
BIP32 pubkey	xpub	xpub661MyMwAqRbcEYS8w7XLSVeEsBXy79zSzH1J8vCdxAZningWLdN3zgtU6LBpB85b3D2yc8sfvZU521AAwdZafEz7mnzBBsz4wKY5e4cp9LB
BIP32 private key	xprv	xprv9s21ZrQH143K24Mfq5zL5MhWK9hUhhGbd45hLXo2Pq2oqzMMo63oStZzF93Y5wvzdUayhgkkFoicQZcP3y52uPPxFnfoLZB21Teqt1VvEHx
Testnet pubkey hash	m *or* n	mipcBbFg9gMiCh81Kj8tqqdgoZub1ZJRfn
Testnet script hash	2	2MzQwSSnBHWHqSAqtTVQ6v47XtaisrJa1Vc
Testnet Private key (WIF, uncompressed pubkey)	9	92Pg46rUhgTT7romnV7iGW6W1gbGdeezqdbJCzShkCsYNzyyNcc
Testnet Private key (WIF, compressed pubkey)	c	cNJFgo1driFnPcBdBX8BrJrpxchBWXwXCvNH5SoSkdcF6JXXwHMm
Testnet BIP32 pubkey	tpub	tpubD6NzVbkrYhZ4WLczPJWReQycCJdd6YVWXubbVUFnJ5KgU5MDQrD998ZJLNGbhd2pq7ZtDiPYTfJ7iBenLVQpYgSQqPjUsQeJXH8VQ8xA67D
Testnet BIP32 private key	tprv	tprv8ZgxMBicQKsPcsbCVeqqF1KVdH7gwDJbxbzpCxDUsoXHdb6SnTPYxdwSAKDC6KKJzv7khnNWRAJQsRA8BBQyiSfYnRt6zuu4vZQGKjeW4YF
Bech32 pubkey hash or script hash	bc1	bc1qw508d6qejxtdg4y5r3zarvary0c5xw7kv8f3t4
Bech32 testnet pubkey hash or script hash	tb1	tb1qw508d6qejxtdg4y5r3zarvary0c5xw7kxpjzsx

Source: https://en.bitcoin.it/wiki/List_of_address_prefixes

HD Wallets (BIP-32) & Seed Phrases (BIP-39)

- For privacy and security, address reuse is strongly discouraged
- Every new address = a new pair of public/private keys
- 100 keypairs = 100 backups
- Seeds come from **BIP-32: Hierarchical Deterministic Wallets**
- Seed phrases come from **BIP-39: Mnemonic Words**



Source: Mastering Bitcoin by Andreas Antonopoulos, Chapter 5

Demo: BIP-32 & BIP-39
https://iancoleman.io/bip39

Secure Key Generation

- Whom is generating?
- Secure Environment
- Offline/Airgap
- Proper Entropy
  - True Random Number Generator (TRNG)
  - Deterministic Random Bit Generator (DRBG)
  - Dice & Cards
  - NOT /dev/random
- Standards
  - NIST SP 800-90A, TRNG, DRBG, DIEHARD, Crypt-X, NIST STS

To summarize

- Public keys are used to create addresses.
- Private keys are used to create digital signatures.
- Digital signatures are used to spend bitcoin.
- Seeds are the root of every Bitcoin wallet.
- A seed is a private key. **HIGHLY SENSITIVE**.
- Seed phrases are human readable versions of seeds.

Additional Resources

- “Mastering Bitcoin” by Andreas Antonopolous - Chapters 4 & 5
  - https://github.com/bitcoinbook/bitcoinbook
- ECC
  - https://blog.cloudflare.com/a-relatively-easy-to-understand-primer-on-elliptic-curve-cryptography/
  - https://hackernoon.com/what-is-the-math-behind-elliptic-curve-cryptography-f61b25253da3
  - “A Course in Number Theory and Cryptography” by Neal Koblitz - “Elliptic Curves” chapter
  - “Programming Bitcoin” by Jimmy Song
- Segwit & Bech32
  - https://www.youtube.com/watch?v=NqiN9VFE4CU&ab_channel=SFBitcoinDevelopers
- Taproot and Schnorr
  - https://bitcoinops.org/en/schorr-taproot-workshop/
- CryptoCurrency Security Standard (CCSS)
  - https://cryptoconsortium.github.io/CCSS/Details/
- Key Generation Demo Script
  - https://github.com/Casa/keyfest/tree/main/2021/keygen-workshop

Q & A