Simple local storage

To send a Bitcoin, you need to now:

- Some information from the public blockchain.
- The owner's secret key.

The goal of local storage is to achieve three properties (which are often in tension):

- Availability: you should be able to spend your coins when you want to.
- Security: no one else should be able to spend your coins.
- Convenience: it shouldn't be hard to spend your coins.

The simplest solution is to just store your secret key in a file (e.g., on your computer or phone). This is very convenient, but only as available and secure as your device (e.g., if the device is lost or wiped, the key is lost, and so are the coins). *(Actually, this is a lot like cash: it's convenient, but only as available or secure as your wallet.)*

The implementation of this idea is known as **wallet software**: gives you a nice user interface, stores all necessary information locally. It's a nice trick to have a separate dress and key for each coin. This leads to better privacy (looks like multiple owners exist) and the software can abstract away all the messy details.

To encode an address, you usually encode it as a text string with base58 notation (like base64, but you remove similarly looking characters, like '0' and 'O'). You can also use a QR code.

Hot and cold storage

There are two primary forms of storage:

- Hot storage: online, convenient but risky.
- Cold storage: offline, archival but safer.

The hot-and-cold-storage approach says that you maintain both hot *and* cold storage, moving money back-and-forth as necessary. Your hot storage lets you use coins day-to-day and your cold storage keeps the remaining coins safe.

Typically, you have hot secret keys and cold addresses in your hot storage, and cold secret keys and hot addresses in your cold storage. If you lose your hot storage, you lose the hot secret keys and those coins disappear. But the majority of your coins (in the cold storage) remain intact. The thief has your cold public addresses, but those aren't useful (except for sending you money).

The hot storage is able to send money to the cold storage, as it has the necessary addresses, but it cannot withdraw.

Note: the cold storage will actually be offline most of the time (e.g., a laptop in a safe).

We'd like to use different addresses for each coin in the cold wallet (for privacy reasons). But how can the hot wallet learn new addresses if the cold wallet is offline? One (awkward) solution: generate a big batch of addresses and keys, and transfer these addresses to the hot storage beforehand.

Hierarchical wallet

A better solution is to use a **hierarchical wallet**. With regular key generation, you generate an address and a private key. Hierarchical key generation, instead, generates some private key generation information and some address generation information. You can then use the former to generate addresses on-demand (e.g., provide an index i and receive the ith address). The same operation can be performed to generate corresponding private keys. It also must be the case that the address generation information doesn't leak any information about the private keys (because that information will be stored on the hot side).

How is this possible? We can use the ElGamal key generation scheme. In the standard procedure, your private key is a and your public key is $g^a \bmod p$, where g and p are public.

For hierarchical, your private information is two values (a, b) and your public information is $(g^a \bmod p, g^b \bmod p)$. The ith private key is equal to $(a + bi)$ and the public key is equal to $(g^a \bmod p)(g^a \bmod p)^i \bmod p = g^{a+bi} \bmod p$.

Note: Bitcoin uses ECDSA, but ElGamal is easier to understand.

You can recur and perform this trick on n levels, which also has some useful applications.

Storing cold information

How do we actually store this information?

1. Information is stored on a device, locked in a safe. This is just physical security.

2. "Brain wallet": encrypt the information under a passphrase that the user remembers.

3. Paper wallet: print the information on paper and lock it up.

4. "Tamperproof (or tamper-evident) device": some device that signs things for you, but doesn't divulge its keys. The adversary's problem reduces to that of stealing or touching the device.

Splitting and sharing keys

The key idea: split a secret key into N pieces such that, given any K pieces, you can reconstruct the secret, but with fewer than K pieces you don't learn anything.

For example, if $N = 2$ and $K = 2$, choose P to be a large prime, define S to be your secret (chosen from $[0, P)$) and choose R to be a random value in $[0, P)$. Split this information into:

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

When you have both pieces, you can calculate S using:

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

A more general scheme involves picking a secret point $(0, S)$ and a random slope R, then sending out $(1, S + R), (2, S + 2R), \ldots$, where all arithmetic is modulo some large prime P. Given any two points, we can draw a straight line and deduce the value of S. No single point reveals any information.

You can generalize to K dimensions. For each dimension, you need an additional parameter (e.g., you need two parameters to define a quadratic and three points to recover it). This supports any K-out-of-N breakdown.

Pros & cons

If you store your secrets separately, the adversary must compromise several shares to get the key. However, to sign, you need to bring the shared secrets together to reconstruct the key. In that moment, you're vulnerable.

Multi-sig

Say there are four individuals and their company stores a lot of Bitcoin. Each of the four participants generate a key-pair and put the secret in safe, private, offline storage. The company's cold-stored coins could use multi-sig so that three of the four keys must sign to release a coin.

Online wallets & exchanges

Online wallets

These are just like local wallets, but they exist "in the cloud". They typically run in your browser. The site stores your keys (perhaps encrypted using your password, or *allegedly* encrypted) and you log in to access your wallet.

These sites could be better at backing up, better at patching security exploits, etc. They're convenient in that they work across devices. However, you lose some control and a lot of transparency. It's also the case that if the site goes away for some reason, you might lose control of your wallet (and, by extension, lose your coins).

Bitcoin exchanges

The core function of a bank is that you give them money (a "deposit") and the bank promises to pay you back later on-demand. Of course, the bank doesn't actually keep your money in the back room (they typically invest it), instead keeping a "fractional reserve" around to pay back as necessary.

Bitcoin exchanges operate similarly. They accept deposits in Bitcoin and fiat currency, promising to pay you back on demand. These exchanges let customers make and receive Bitcoin payments, or buy and sell Bitcoins for fiat currency by matching up buyers and sellers.

Suppoes your account holds \$5000 and 3 BTC. You tell the exchange that you want to buy 2 BTC for \$580 each. As a result, your account holds \$3840 and 5 BTC. But there's no *actual* transaction that appears on the blockchain. Instead, the exchange is just altering their promise: instead of paying back \$5000 and 3 BTC, they promise to pay \$3840 and 5 BTC.

These are change to the Bitcoin ecosystem that are occurring off-chain. The only evidence that you have of any change is the promise of the exchange. This is similar to when you interact with an actual bank.

Pros

Exchanges connect the BTC economy to the fiat currency economy and provide an easy means of transferring value back-and?forth. Many of these exchanges can also make transfers to your regular bank account, adding an additional level of convenience.

Cons

The main worry is **risk**. Exchanges face many of the same risks as banks, the first being *runs*: if everyone comes to get their money out at the same time, the exchange is ruined. (The real banking economy has FDIC, but Bitcoin exchanges do not.)

There's also the problem of Ponzi Schemes. In effect, this is the risk that the individual running the exchange is a crook, spending your money on yachts and vacations.

Exchanges are also frequent targets of attacks from hackers. With Bitcoin, if an attacker gains access to your secret keys or finds an error in the transaction engine, they have all they need to exploit the service.

Bank regulation

Bitcoin exchanges fail at a 45% rate, but of course real banks do not. For traditional banks, governments do a number of things:

- Impose minimum reserve requirements.
- Regulates behavior and investments.
- Insures depositors against losses (FDIC).
- Acts as a lender of last resort.

Proof of reserve

Bitcoin exchanges can prove that they hold a certain fractional reserve–this fraction can even be 100%.

To do so, they start by publishing valid payment-to-self of some amount (e.g., to show that they have 10,000 BTC, they can pay themselves 10,000 BTC and observe that the transaction made it into the blockchain). This proves that they have at least X BTC on them at any time.

They can also prove how many demand deposits they're holding. To do so, they build a Merkle tree in which each leaf is a user account ID and the number of coins in that account. Each has pointer includes the total value in its subtrees. Due to collision-free properties of our hash functions, such a tree cannot be spoofed, and we can prove that we hold a certain account in the tree using just $\log n$ hashes.

The overall **proof-of-reserve** technique is to show that you have at least X amount of reserve currency. Then, you show that your customers have at most Y amount deposited (although this could be a lie as you could include fake accounts). Therefore, Y could be smaller in reality and X could be larger in reality. This guarantees that the reserve fraction $\geq \frac{X}{Y}$. (Even if X is larger, the fraction still holds. And if Y is actually smaller, the fraction *still* holds.)

Payment services

Say a customer wants to pay with Bitcoin. The merchant will want to receive dollars with a low risk, simple-deployment system.

This need is typically met by some sort of service (e.g., Coinbase). They give you some HTML button that lets users pay with Bitcoin. The end result is that the customer pays Bitcoin, the merchant gets dollars (minus a small percentage, which goes to the payment service), and the payment service gets some Bitcoins, paying out dollars and absorbing some risk.

Transaction fees

Recall that the transaction fee of a Bitcoin transaction is calculated as the (value of inputs) − (value of outputs). The fee goes to the miner.

It costs peers some small amount to relay your transaction and it costs miners to record your transition. The transaction fee compensates for (some of) these costs. Generally, a higher fee means that the transaction will be forwarded and recorded faster.

However, there's no fee if all of the following are true:

- The transaction is less than 1000 bytes in size.
- Each of the outputs are 0.01 BTC or lager.
- The priority large enough, where priority is calculated as the sum of the input ages multiplied by the input value divided by the transaction size.

Otherwise, the (consensus) fee is 0.001 BTC per 1000 bytes. The transaction size can be approximated as $148N_{in} + 34N_{out} + 10$.

By "consensus", we simply mean that the plurality of miners implement this ruleset; however, others might demand higher fees or take no fees at all. In general, if you don't pay the consensus fee, your transaction will take longer to record, and miners prioritize transactions based on fees *and* the priority formula.

Currency exchange markets

There are many exchanges where you can trade dollars for Bitcoin. There are also services that can match you up to find buyers or sellers near you that you can go meet in person.

The market typically matches buyers and sellers. The supply is equivalent to the number of coins in circulations, along with demand deposits (although this is subject to debate, depending on whether or not demand deposits can be spent). The number of coins in circulation is fixed, currently at 13.1 million.

The demand for Bitcoin comes from multiple sources. Firstly, Bitcoin is used to mediate fiat-currency transactions (e.g., when a merchant wants to paid using Bitcoin eventually). Secondly, Bitcoin is demand as an investment (e.g., if the market thinks demand will go up in the future).

We can come up with a simple model of transaction demand. Let T be the total transaction value mediated via BTC (in dollars per second), D be the duration that BTC is needed by a transaction (in seconds), and S be the supply of BTC. We have $\frac{S}{D}$ BTC available per second to service a new transaction and $\frac{T}{P}$ BTC needed per second, where P is the price of a Bitcoin in dollars. Based on market economics, we'll have equality between these two factors, such that $P = \frac{TD}{S}$ in equilibrium.

We consider D to be something of a constant. Over short time intervals, S is also fairly constant. Thus, the price of Bitcoin is proportional to the transaction volume mediated by Bitcoin.