October 6: Bitcoin

Scribe: Jake Lerner

October 18, 2015

1 Introduction

Bitcoin is a peer-to-peer electronic cash system, surrounded by a broad social, economic, and technical literature. These notes, however, focus on Bitcoin as a system built on cryptography. After an introductory description of Bitcoin's history and goals, the need for Bitcoin's key cryptographic concepts is established though a series of straw-man electronic currencies. Then, these cryptographic concepts are developed along with the rest of the Bitcoin protocol.

1.1 History

Bitcoin was established in a 2008 white paper by Satoshi Nakamoto. Little is known about Satoshi Nakamoto (which is likely a pseudonym, and means 'clear thinking' or 'direction'), save that they are quite rich, possessing over one million bit coins.

1.2 Why does a Currency need Cryptography?

In short, electronic cash systems are easy to design if you assume there are trusted parties, but when it comes to money, few people can be trusted. This necessitates a cryptographic currency.

Why do we need an electronic currency in the first place? Whereas cold, hard, government issued cash is great for day-to-day purchases, an e-currency would be immediately transferable online and wouldn't require any central, trusted authority like a mint. Credit cards already allow online payment, but these merely use trusted parties to transfer guarantees of centralized physical currency. This means regulations, lack of anonymity, and transaction fees. There are thus clear economic (and, depending on viewpoint, political) benefits to developing an e-currency which doesn't rely on trusted parties.

However, building such a system requires overcoming several technical and socioeconomic hurdles. Technically, the e-currency must prevent users from spending the same currency twice, from spending more currency than they possess, and from spending money possessed by other users. Bitcoin addresses all of these challenges. Socio-economic challenges (which, generally, aren't addressed in these notes) include providing incentives to contribute to whatever collective infrastructure the currency requires, creating value for the currency, and working the currency into existing legal and monetary frameworks.

2 Towards Bitcoin

Here, we develop a series of straw men to demonstrate the need for the key cryptographic concepts of Bitcoin. For each system, we assume Alice wishes to send money to Bob, describe how she does this, and discuss the strengths and weaknesses of the approach. We use S_a(message) to represent a message signed with Alice's private key.

2.1 Strawman 0: Digital Signatures

Currency: S_a(”Alice gives $7 to bob for a hamburger”)

Here, the signed message *is* the currency. The message is provably created by Alice, and the amount or purpose of the transaction can't be altered by anyone besides Alice, because it has been cryptographically signed.

Strengths: Nobody but Alice can spend Alice's money, and Alice can't steal other people's money.

Weaknesses: Alice can spend money twice, or spend money she doesn't have.

2.2 Strawman 1: Digital Signatures with Unique Serials

Currency: S_a(”Alice gives $7 to bob for a hamburger. Transferring coins 900439, 900440, 900441, 900442, 900443, 900444, 900445”)

Alice includes the serial number of each bill she's sending in the signed message; giving each bill an identity allows it to be tracked.

Strengths: As above, but Alice also can't overspend: she can't send Bob coins unless she knows the serial numbers for those coins.

Weaknesses: Alice can double spend, by sending the same coins to different people who don't talk to each other.

2.3 Strawman 2: Digital Signatures with Unique Serials and Central Bank

Currency: $S_bank(S_a$(”Alice gives $7 to bob for a hamburger. Transferring coins 900439, 900440, 900441, 900442, 900443, 900444, 900445”))

A central bank keeps a list of who has what coins, and, after checking that Alice currently owns all coins she is sending to Bob, notes that Bob now owns those coins, and signs the message.

Strengths: Nobody but Alice can spend Alice's money, and Alice can't steal other people's money or double spend money. This system is totally reliable.

Weaknesses: This system relies on a trusted party, which is one of the things we wanted to avoid.

2.4 Strawman 3: Digital Signatures with Unique Serials and Collective Bank

"From this moment on, is where the magic of Bitcoin starts. . ."

Currency: As in Strawman 1, S_a("Alice gives $7 to bob for a hamburger. Transferring coins 900439, 900440, 900441, 900442, 900443, 900444, 900445").

Here, though, Bob has a list of who owns each coin, and verifies Alice owns these.

Strengths: Assuming Bob has complete knowledge, nobody but Alice can spend Alice's money, and Alice can't steal other people's money or double spend money.

Weaknesses: How can Bob trust that his list of who owns which coins is accurate and up to date?

Possible Solutions:

Naive- In a very basic system, every user would be notified of every transaction, and update their lists of who owns which coins accordingly. However, this system puts a huge amount of trust in users: Alice could, for example, not notify Bob that she had already given money to Chris, and then pay Bob with money she had already spent.

Majority Vote- An improved system would require a majority vote from all users to verify a transaction. Bob updates his list based on a majority of other clients, so no user could steal or double spend without controlling a majority of the clients. This system is much stronger, but retains a key weakness – creating clients is very easy, so it would be completely feasible for Alice to forge many identities, and then have those identities provide inconsistent views of the system to Bob and Chris, allowing her to double spend. Requiring consensus among users also fails, as it allows a single malicious user to disrupt the entire system. The key insight behind Bitcoin is a system for communal transaction verification which incentivizes cooperation in the scheme and makes forging a majority extremely difficult: Proof of Work.

Proof of Work: If we had a means for users to prove they had carried out a certain number of computational cycles, we could reward users for contributing cycles towards the system (by verifying other users' transactions) and make it prohibitively expensive to forge a majority (by requiring a majority of cycles to verify a transaction). How can we achieve such a 'proof of work'? With a good one-way hash function, we can do this by asking users to determine an arbitrary nonce which, when hashed with a transaction, produces a hash that begins with some number of zeros (more zeroes requires more work).

3 Bitcoin Design

Bitcoin assumes an open (and changing) set of peers, each with (eventual) access to all bit coin transitions. Bitcoin assumes the threat model that any peer could misbehave, but a majority of the computation cycles are trusted, and follow Bitcoin protocol. Each bit coin is a chain of transactions representing it's own history of sale. Transactions are bundled into 'blocks', and a series of blocks over time represents a 'block chain', which takes (provable) work to assemble, so that the longest block chain can be considered the 'most recent' and, in some sense 'correct', block chain. We'll go into two key aspects of this design, transactions and block chains, and then discuss how Bitcoin incentivizes work and handles privacy.

3.1 Transaction Model

A Transaction record contains the public key of the new owner and a hash of the previous transaction, signed by the current owner of the bit coin (Really, a Transaction construct contains a series of currency transactions, but here we'll assume it contains only one). Each of these items is vitally important:

New Owner's Public Key: Including the new owner's public key in the transaction fulfills the same function as including the name ”Bob” in the signed messages of our straw men examples - it indicates the recipient of the message. Using a public key allows the new owner to PROVE the coin is theirs, as only they can sign it.

Previous Hash: Including the hash of the previous transaction functions much like the serial number in our straw man examples. If Alice attempts to doubles spend the coin, she'll have two transaction records with the same hash.

Signature: As in our straw man examples, Alice signs the transaction with her private key, proving herself to be behind the transaction. In the real world, this means stealing private keys can be extremely lucrative.

In its entirety, an arbitrary transaction T_N between sender S and recipient R looks like

$$\mathrm{T}_N = (PK_R, \mathrm{HASH}(T_{N-1}))SIGN_S$$

where PK_R is the public key of the recipient, $HASH(T_N)$ is a hash of the previous transaction (say, transaction number N), and $SIGN_S$ represents the digital signature of the sender. If recipient R subsequently wished to the spend their newly received money to some third party Q, that transaction would look like

$$\mathrm{T}_{N+1} = (PK_Q, HASH(T_N))SIGN_R$$

To make this transaction, R would send this to a few peers, who would then send the transaction to the rest of the peers. Each peer checks that the coin has not been double spent. To do this, the peer fetches T_N from it's block chain, and makes sure that the sender of T_{N+1} is the recipient of T_N (in this case, R). This is checked by using R's public key from T_N to verify R's signature in T_{N+1}. Then, the peers check against double spending, by making sure there is nothing in their block chain that chains after T_N. Even after being checked by peers, however, the transaction isn't committed. First, it must become part of the Block Chain.

3.2 Blocks and Block Chains

After checking a transaction, each peer bundles it in a 'block' with other transactions it has recently heard of and verified. This Block consists of a hash of the previous block, a set of transactions, a timestamp, and nonce (for proof of work). When this is all hashed, the result should begin with some number of zeroes, depending on how much work is required.

Every peer stores the entire block chain, and works to create the next block. If the peer succeeds in creating the next block, it broadcasts that block to the entire network, and all other peers add the block to their own block chain if the new block is valid. To check that a newly received block is valid, each peer must check proof of work and transaction validity of the block. Proof of work is easily checked by making sure the hash of the block must begin with the required number of zero bits. Transaction validity is more difficult, and requires individually checking the validity of each transaction included in the block, as described in the above section.

If there are multiple valid blocks, a peer must keep the one with the longest block chain, or if both are the same length, the peer will keep a both blocks until a longest block chain emerges. Thus, there is no notion of the 'current' block chain, merely the block chain which is the longest (and thus has the most proof of work). Because most cycles are honest, and the probability of being able to create blocks with a minority of cycles decreases dramatically with the number of blocks created, there will generally only be a difference in the last few blocks, and forks tend to be quickly resolved. Thus, in Bitcoin a transaction is considered fully committed only when there are a few blocks in the block chain after the block containing the transaction in question.

3.3 Incentives

Why would Bitcoin Peers do so much work to commit other users' transactions? Bitcoin provides peers with two incentives for generating valid blocks. First, at the beginning of each block is a transaction which grants some amount of money to the user who 'mined' (created) the block. These will decrease over time, to gradually be replaced by transaction fees. With transaction fees some portion of a transaction (chosen by the sending party) goes to the peer that creates a block containing that transaction. Right now, there is still 'room' in blocks for more transactions so there is always an incentive to include transactions with even trivially small transaction fees in a block. These two incentives combine to make block creation sufficiently lucrative that huge amounts of energy (electrical, human, computational) have been put into generating new blocks. Lots of validation work is required even for peers that don't 'win' by creating blocks. However, this work is necessary to have an up to date chain that will allow that user to create future blocks, and so is still incentivized.

3.4 Privacy

Many people use Bitcoin because they think it's anonymous. While it's true that you're using a public key rather than real identity, or something explicitly linked to real identity (like a credit card), there's substantial research showing that people can easily be linked to public keys. Some wallet and vendor features, like wallets with multiple keys and vendors with Bitcoin tumblers, can help provide more privacy to users, but these methods are usually still traceable back to users if sufficient effort is invested. Fundamentally, the 'point' of Bitcoin is not to be a private or anonymous e-currency, but to be an e-currency with no trusted party. However, current research into using zero-knowledge proofs for Bitcoin validation (Zerocoin and Zerocash) could result in real privacy for Bitcoin transactions in the future.