6.857: Computer and Network Security February 24, 2014

Bitcoin

Lecturer: Justin Holmgren

1 Introduction

In today's lecture I will be discussing a slightly simplified version of the Bitcoin crypto-currency. To start, consider how one would go about designing a digital currency, slowly making it more distributed and robust.

The first, and simplest approach, would be to have a central party (like a bank) manage all accounts. This would be somewhat analogous to how banking works in the real world. When Alice wants to send $10 to Bob, Alice visits her bank. The bank asks her to show some identification (authentication) and checks that she actually has at least $10 in her account (authorization). If both of these are true, the bank will subtract $10 from Alice's account and add $10 to Bob's account.

There is a simple way to implement the same scheme digitally. Alice and Bob will both have private keys, as well as a public key which is known to the bank. When Alice wants to transfer money to Bob, she signs the following message using her secret key, and sends it to the bank: "Bank, please give $10 to Bob. Love, Alice". The bank can authenticate Alice by verifying the signature, and can check authorization by looking in its accounts database. *Note*: In order to prevent replay attacks, i.e. Bob recording Alice's message and repeatedly sending it to the bank, each message from Alice should also have a unique number in it. The bank should then not process duplicate messages twice.

Fundamentally in this scheme, Alice and Bob are both trusting the bank. The bank could in principle empty Alice's account, or give tons of money to everybody except them, thereby devaluing Alice and Bob's money. It seems plausible that public key cryptography can help with this situation. After all, anybody can verify that Alice really did sign a message promising $10 to Bob. Unfortunately, this isn't sufficient. Everybody also needs to be able to verify that Alice has $10 to give. Essentially, Alice's message is the equivalent of a check. Before Bob takes the check to the bank, he doesn't know whether or not the check will clear.

A hypothetical construct which would enable this is a public ledger - a way for everybody to see which checks have been written, and to whom, and in what order. If everybody agrees on the ordering of the checks, then everybody can agree on how much money different accounts have at different points in time, and also on whether a given check "succeeds" or not.

This seems bad for anonymity, and it is, but there is a small bit of consolation in the fact that users will only be identified by their public keys. In this sense, Bitcoin is a *pseudonymous* currency, but because different transactions can be linked to the same pseudonym, it is not fully anonymous.

2 Public Ledger via Proof of Work

Bitcoin's main technical contribution is implementing a distributed public ledger that is robust against computationally bounded attackers. It will achieve eventual agreement in the form of a list of transactions which is ideally append-only, and the probability of any participant changing their mind about a transaction decreasing exponentially with the age of the transaction. So one mantra to keep in mind is that old transactions are set in stone, while recent ones can be volatile. A transaction is essentially just a signed check as decribed in the introduction.

2.1 Block Chain

Bitcoin clients maintain lists of blocks of transactions called block chains with several properties which together imply that a client can become more and more certain about the validity of a transaction as it gets older.

1. It is difficult to extend a block chain with a new block of transactions to yield a longer valid block chain. Finding a valid extension of a block chain is called "mining" a block. The difficulty of mining a block is set such that the expected time for any Bitcoin participant to mine a block is 10 minutes. Furthermore, the time to next mined block is a memoryless (exponential) distribution, and the probability of any one participant mining a block first is proportional to that participant's hash rate.

2. Honest participants always attempt to extend the longest block chain they are aware of and they communicate the longest chain they have seen to the rest of the network (via a gossip protocol).

3. An honest participant always regards the longest block chain they have seen as valid. Note sometimes this may change in a way other than simply being extended.

When the honest participants comprise a large majority of the computational power in the network, the old end of the block chain will almost certainly be unique. More formally, given a longest block chain, the probability that the kth from most recent block will ever not be a part of the longest block chain decreases exponentially with k. The longest ever orphaned fork of the blockchain was four blocks long, and standard practice is to consider a transaction valid after six "confirmations".

2.2 Economic Incentives

So far I've described a protocol and argued for security assuming that all of the participants in the Bitcoin protocol act honestly - that is, in accordance with the protocol. In fact, the security of Bitcoin relies on a large amount of computation acting honestly. But because there is a cost to computation (electricity) and the potential for money to be made, expecting everybody to act honestly is questionable. In fact, Bitcoin rewards people for participating by allowing the miner of a block to give a fixed Bitcoin reward to themselves. This reward does not come out of other Bitcoin accounts; these are "fresh" Bitcoins. At the moment, the reward is 25 BTC, which as of March 2014 conversion rates is about $15,000 USD. Bitcoin is designed to have a capped supply of Bitcoins, so the reward halves every four years.

Miners also can receive a specified transaction fee from whichever transactions they incorporate into their block. The transaction fee is set by whomever issues the transaction and is optional. An economically motivated miner should therefore include whichever transactions have the highest associated transaction fees.

At an economic equilibrium, the reward per mined block should be equal to the expected cost of mining a block.

2.2.1 Collusion / Contracts?

I claim that in the presence of contracts and/or collusion, there are economic incentives for cheating behavior in Bitcoin. For example, if I buy a yacht for 10,000 BTC and the transaction is 6 blocks old, I can promise to pay 100 BTC to whomever mines a block which helps to reverse my payment. This is beneficial to me (I'll only have to pay out around 600 BTC), and there is now an equilibrium dishonest behavior in which everybody tries to reverse my transaction. Furthermore, even if I didn't buy a yacht, I can pool funds with the other senders of transactions in my block, and collectively issue a similar bribe.

3 Distributed Mining

Even assuming that there is no collusion and there are no contracts, the economic incentives behind Bitcoin seem shaky. If I have a small computer, I will nearly never be able to mine a Bitcoin block, even though my expected payout / cost ratio is the same as anyone else's. Fortunately, Bitcoin easily lends itself to reducing the variance in payout with *mining pools*.

The idea behind distributed mining is that people will band together in mining pools with a trusted pool operator. Instead of looking for blocks which pay themselves, everybody looks for blocks which pay the pool operator. Incidentally while searching for such blocks, it is much more likely that a block satisfying a lower difficulty level (hash has a fewer number of starting zeros) will be found. Such a block can be sent to the pool operator as evidence that a miner has been contributing to the pool. In case a miner does find an actually valid block, note that he cannot convert it into a block which pays himself. Instead, his best action is to send it to the network, resulting in money for the pool operator which will be distributed to all the contributors in finding the block.

4 Bitcoin Scripting and Cryptocontracts

Earlier I said that transactions from Alice to Bob are just messages, signed by Alice, containing Bob's public key as well as the amount which Alice wishes to send. That was actually a simplification, and in fact Bitcoin allows a much more general way of sending money. Rather than specify Bob's public key, Alice's transaction includes a program in a Bitcoin scripting language. Now a transaction can spend these Bitcoins if it also provides an input which will make Alice's program accept. For example, in a standard transaction, the program will verify that the input is a valid signature (under Bob's public key) of the hash of the transaction.

However, it's possible to use much more interesting scripts for a transaction. One example models a real-life type of exchange known as escrow. If Alice is buying a book from Eve online, a plain Bitcoin transaction leaves her vulnerable to Eve never shipping the book. If this happens, the transaction is still irreversible. One solution is for Alice to send her money instead to a trusted third party known as an escrow agent. This escrow agent will ask Eve for proof that she sent the book (perhaps a video of Eve putting the package in the mail). If Eve provides such such proof, the escrow agent will give Eve the money. Otherwise, the escrow agent will return the money to Alice. In Bitcoin, Alice can make a transaction which allows Eve to spend the Bitcoins with a signed message from the escrow agent saying "OK", and allows Alice to spend the Bitcoins with a signed message from the escrow agent saying "Not OK". This totally eliminates the possibility of the escrow agent absconding with the money. Furthermore, if Alice and Eve can't agree to trust one particular escrow agent, the program can also use a majority vote system which enables spending give a majority decision by n escrow agents.