CS 3700
Networks and Distributed Systems

Lecture 19: Bitcoin

What is money?

- Many things; two are germane to this discussion:

- Medium for exchange
  - Not valuable for itself; rather for future exchanges

- Store of value
  - Allows one to easily "store" value (instead of objects)

Pros/cons of physical money

- Easily portable
- Cannot double-spend (spend the same $ in two places)
- Cannot repudiate after payment
- No need for trusted 3rd party for transactions
- Semi-anonymous (modulo serial #s, tracking, etc)
- Doesn't work online
- Easy to steal (it's a bearer token)
- Hard to tax / monitor cash transactions
- Government can print more as economy expands/conditions dictate

What about electronic money?

- e.g., Credit cards, Paypal and bank e-checks are similar
- Unlike cash, does work online
- More difficult to steal (sometimes)
- One can repudiate a transaction (credit card *chargeback*)
- Requires trusted 3rd party for transactions
- No privacy: All purchases tracked
- Government can censor/prohibit transactions
- Easy for government to monitor/tax/control

Bitcoin

- Goal: e-cash without a central trusted third party
  - Basically, electronic cash that is closer to offline cash

Outline

- Why is p2p money hard?
- Work though simple designs
- Actual Bitcoin protocol, design
- Security analysis
- Bitcoin in practice

Why is peer-to-peer money hard?

- forgery
- double spending
- theft
- ownership

- Rest of lecture: Build up design of Bitcoin using strawman proposals
  - Will call our protocol "neucoin"

Assumptions, goals

- No “strong identities” (i.e., can’t rely on passports, etc)
  - Would like some anonymity if possible (like cash)
- No central entity with control
  - E.g., US Treasury issues money, etc
- Payments entirely electronic
- Expected properties of money:
  - Cannot generate money you don’t have
  - Can only spend each coin once
  - Clear ownership of each coin
  - No repudiation

How can Alice send to Bob?

- Alice prepares a message:

I, Alice, send one neucoin to Bob

- Problems?
  - Can message be forged? *Yes*
  - Can neucoins be stolen? *Yes*
  - Can Alice double-spend? *Yes*
  - Can we tell who "Alice" is? *No*

- Can cryptography help with message forging and identity?

Introducing cryptography

- Entities are “wallets” — simply a public/private keypair
  - Knowledge of private key gives ownership

- Sending money is giving money to a public key

How can Alice send to Bob? (v2)

- Alice prepares *and signs* a message:

I, Alice's public key, send one neucoin to Bob's public key
🔒 Alice's private key

- Problems?
  - Can message be forged? *No*
  - Can neucoins be stolen? *No, if private key is private*
  - Can Alice double-spend? *Yes*
  - Can we tell separate transactions apart? *No*

- Can serial numbers help with double-spending?

Where do serial numbers come from?

- How do we prevent Alice from "making up" a neucoin?

- We need a *trusted third party* to issue serial numbers
  - Also known as a bank
  - In our context, bank would have well-known public key

- Serial number would be

Serial number 10238

Bank's private key

How can Alice send to Bob? (v3)

- Alice prepares and signs a message *with a specific serial no*:

I, Alice's public key, send neucoin [Serial number 94839 — Bank's private key] to Bob's public key — Alice's private key

- Problems?
  - Can Alice double-spend? *Sort of*
  - Suppose Alice also signed the message

I, Alice's public key, send neucoin [Serial number 94839 — Bank's private key] to Charlie's public key — Alice's private key

- Who owns neucoin 94839?

Preventing double-spending

- Could have the bank also track who owns which coin
  - Bank would have a ledger, be official record
  - Bob can contact bank, verify that Alice owns that coin
  - But, defeats the purpose of Bitcoin (no central bank)

- Instead, *the network is the bank*

- Network collectively keeps track of *all transactions*
  - Called the *public ledger*
  - To verify Alice isn't double-spending, look in the ledger
    - Charlie would notice 94839 wasn't Alice's

In more detail

- Each network node (Bitcoin client) keeps record of all transactions
  - Ledger (blockchain) is public (but pseudonymous)

- Implication: You can download the entire Bitcoin transaction history

- Now, Bob/Charlie can *broadcast* transaction to all nodes
  - Nodes verify transaction, and respond
    - Verify: Correct signature, Alice owns neucoin 94839
  - Nodes also add transaction to the public ledger (*blockchain*)
  - Once "enough" nodes respond, accept transaction

But, what if Alice sends simultaneously?

- What is Alice sends *both* messages at the same time?
  - Both Bob and Charlie will attempt to verify, accept the transaction
- Idea: Bob and Charlie should wait for N/2 nodes to respond
  - At least half the network must accept the transaction
  - …doesn't seem particularly scalable…
- But, subtle problem: what is a node?
  - Any Bitcoin client
  - What would it take to run multiple nodes?

Sybil attacks

- Alice could introduce “Sybils” (fake nodes under control)
  - Would allow her to respond to Bob/Charlie differently
  - Remember, Bitcoin node is just a process; could lie

- *Fundamental problem* for distributed systems
  - Alice could “fake” many, many nodes
  - Respond selectively to Bob/Charlie
    - Have N/2 respond “OK” to Bob, another N/2 to Charlie

- Implication: Voting (one vote/node) doesn’t work
  - Instead, need something more powerful

Proof-of-work

- Need to tie voting to a resource hard to obtain
  - Identities (passports) are an obvious choice, but defeats purpose
  - Idea: Can we tie voting to *computation resources controlled*?
- Why a good idea?
  - Would obviate need for Sybil prevention
- How can we accomplish this?
  - Use *proofs of work*, via crypto puzzles
  - Proves that entity expended effort, allows voting

Cryptopuzzles

- Recall our discussion of hash functions
  - Hash function: f(X) -> H (e.g., MD5, SHA-1, etc)
    - Input range is arbitrary
    - Output range is fixed-width (e.g., 256 bits)
  - Hash functions are *cryptographically secure* if:
    - Hard to find a pre-image for a given hash value H

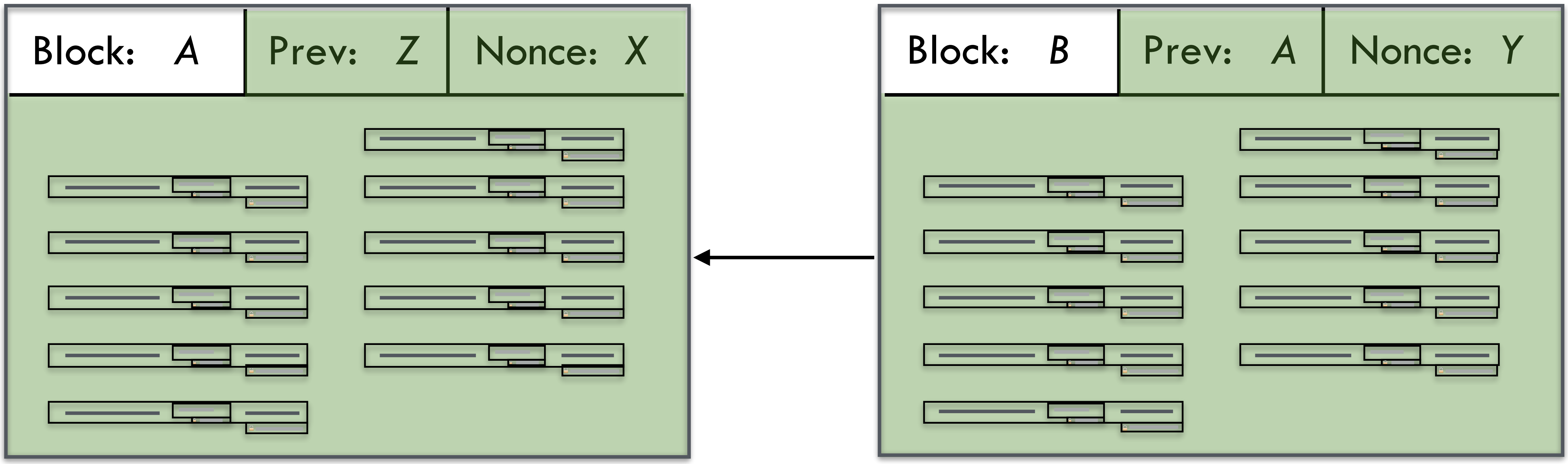
- Implement cryptopuzzle in neucoin as follows:
  - Find a value V such that
    - f(V + [some other fixed data]) < *target*
  - No choice but to "brute force" different values of V
  - Can change difficulty by making *target* bigger/smaller

Proof-of-work in Bitcoin

- Essentially, idea is to
  - Ensure you can only add an entry to the ledger if you've done work
  - Changes "one node/one vote" to "one CPU/one vote"
    - Much harder for Alice now
    - She must have access to LOTS of CPUs to out vote honest users

- How to implement this in Bitcoin?
  - First, introduce the notion of "blocks"
  - Essentially groups of transactions
    - Nodes receive transaction broadcasts, add to current block

Blocks



- Block is group of transactions
  - *Block (ID)* is the hash of all other fields (in green)
  - *Prev* is the *ID* of the previous block
  - *Nonce* is a number chosen to make the *ID* small "enough"
    - Changing *nonce* changes the *ID* of the block unpredictably

Blockchain

- Next block must have *ID* < *target*
  - *target* changed so that 1 block/10 minutes, on average

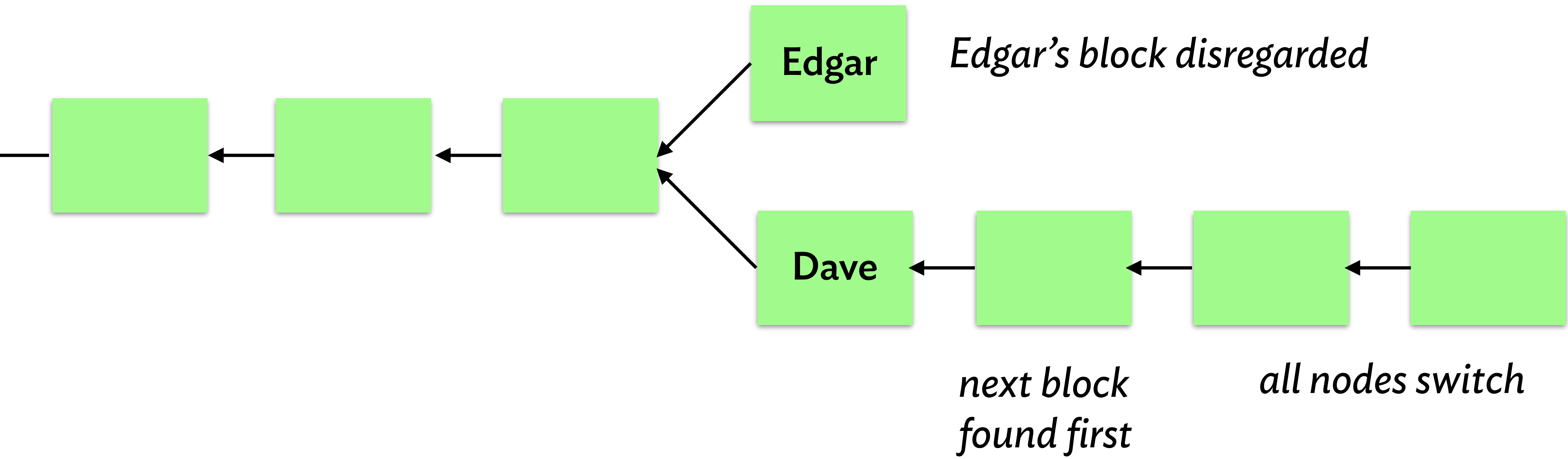
- So, at any time, all nodes “searching” for next block
  - Searching == trying different *Nonces*
  - Hoping to get lucky, find block with *ID* < *target*

- When node discovers such a block, it broadcasts to the network
  - Other nodes verify
  - Start searching for the next block (with new block as *Prev*)
  - “Blockchain” is all of these blocks together
    - Starting with special *genesis block*

What if two blocks found simultaneously?

- But, what if two nodes find *different* blocks at the same time?
  - Say, nodes Dave and Edgar?
- Both Dave and Edgar broadcast
  - Some nodes start working on Dave's "fork", others on Edgar's
  - Bad, right!
- In Bitcoin, nodes always believe "longest" chain
  - Chain the represents the most work
  - Eventually, either Dave's or Edgar's fork will find *next* block first
    - When that is broadcast, all nodes switch to longer chain

Blockchain split



- In case of split, network searches for new blocks in both chains
- First chain to be lengthened "wins"
  - All nodes switch
- Other block is ignored; and transactions go back into queue

Creation of new coins

- But, this seems like a LOT of work for the nodes
  - Running hashes is CPU-intensive
  - Why do they do this?

- Bitcoin solves incentives in two ways:
  - Transactions can provide a *transaction fee*
    - Amount of to be paid to node who “wins”
  - New blocks introduce new coins
    - Node who wins also claims fixed amount of bitcoin as a prize
    - Currently, 25 BTC (today, ~$5,000!)
    - Called *coinbase* transaction, simply another transaction

Coinbase transactions

- Elegantly solves problems of:
  - Where do bitcoins come from?
  - Who gets initial bitcoins?

- Successful node claims reward

- Amount drops over time
  - Halves every 210,000 blocks
  - Currently 25 BTC (was 50 BTC until 2012)
  - Will become 0 in year 2140; 21 million total coins
  - At that point, only transaction fees will incentivize nodes

Can we get rid of coin serial numbers?

- Final annoyance: where do bitcoin serial numbers come from?
  - Answer: There aren't any

- Idea: "bitcoins" don't matter; transactions do
  - All transactions given an *ID* (simply a hash of attributes)
  - When transferring a bitcoin
    - Simply state *ID* where you received the bitcoin
    - Makes it easy to verify signature, ownership

- What if you don't want to transfer *all* of the previous transaction(s)?
  - Multiple recipients: Pay yourself change :)

Real bitcoin transactions

- Real transactions have multiple inputs/multiple outputs
  - Each input is simply the identifier of a previous transaction
    - All value must be included
    - Nodes verify no other transaction refers to this one
  - Each output is an amount, and a public key
    - Signed by owner's private key
  - Implicit output: Difference between Sum(input) and Sum(output)
    - If exists, can be claimed by node that finds next block

Outline

- Why is p2p money hard?
- Work though simple designs
- Actual Bitcoin protocol, design
- Security analysis
- Bitcoin in practice

Is Bitcoin “secure”?

- Can I “fake” a transaction? (i.e., steal your bitcoins)
  - No, I need access to your private key
- Can I edit the blockchain? (i.e., remove an old transaction)
  - No, as hash function protects all previous transactions
  - Can’t find a “preimage” (alternate history)
- Can I create money out of thin air?
  - No, only allowed “new” coins are coinbase transactions
  - Other nodes would not accept new block
- Can I repudiate transaction? (i.e., deny that I paid you)
  - No, message has your signature (only you could generate)

What about double-spending?

- Can I double-spend?
  - Sort of — could publish two transactions with same input
  - But, network will only eventually accept one of them

- Recipient should wait until transaction appears in blockchain
  - Not really a guarantee, though
  - A longer chain could appear, nullify transaction

- Ultimately, rely on hardness of generating a blockchain
  - Faster than honest nodes working on fork containing transaction

What if I control many CPUs?

- Say, if I control 51% of the network's CPU capacity?

- In this case, I could re-write the blockchain
  - Remove transactions from existence
  - Requires dedicating all my resources to finding "alternate" chain
    - Once found (and longer than "real" chain), publish
    - Honest nodes will switch to my chain
    - All transactions in honest chain will be disregarded

- So, need to have diversity of nodes in the network to avoid

What about incentives?

- Why do nodes accept transactions?
  - Transaction fees; monetary reward
- Why do nodes “accept” a new block?
  - Couldn’t they just ignore it and keep “mining” the old one?
  - No incentive: Mining is guessing, so it’s not like they are “close”
  - Also, all other nodes will switch to new block
    - Any mined block would be worthless

Outline

- Why is p2p money hard?
- Work though simple designs
- Actual Bitcoin protocol, design
- Security analysis
- Bitcoin in practice

Using bitcoin

- Basically, two options: Desktop Client or Online Wallet Service
- Client: You participate as node in the network
  - Private key on your machine (lose it, lose your coins)
- Wallet: You give your private key to a company/site
  - Log in to site to view "balance", make transactions; easy to use
  - They have your key

- What's up with the stolen bitcoins?
  - All from Wallet sites
  - Hackers break in, get private keys, transfer bitcoins to themselves

Bitcoin wallets

- Essentially a public key
  - Referred to as "wallet address"

- Single user can have many wallets
  - All you need is to generate another keypair

- Best practice: Generate new wallet *for every transaction*!
  - Makes correlating transactions much harder
  - Users worried about government tracking, etc

- Many users "launder" bitcoins using "mixers"

“Mining” bitcoins

- You can download and run “mining” software
  - Your node will search for next block, etc
  - You could win!
    - But you won’t

- Today: mining isn’t worth the electricity cost for your machine
  - Real miners use ASICs (dedicated hardware)
    - Run hashes *really fast* and *really power-efficient*
  - Many mining pools set up in Iceland (cheap power+cooling)

Mining pools

- Problem: Bitcoin is a lottery
  - You are extremely unlikely to win
  - Can we make it more "fair"?
    - Nodes "get out" what they "put in"?

- Solution: *Mining pools*
  - Groups of nodes that work together
  - Split proceeds when any node finds the next block (more fair)

- Lots of mining pools today
  - Some represent up to 25% of mining capacity!

Proof-of-work in mining pools

- How to evenly distribute coins in a mining pool?
  - How to determine what nodes "put in"?
  - Nodes could lie, say "I worked really hard!"

- Elegant solution: Nodes report "best hash" they found for block
  - I.e., they say "I didn't win, but here's the best I did"
  - Corresponds to amount of effort expended

- Distribution then based on how "hard" best hash was
  - Closer to target, more coins

Bitcoin exchange rate

- BTC-USD exchange rate very volatile
  - High over over $1,000/BTC, now ~$200/BTC (Jan 15)
  - Worries over security, feasibility as a currency
- A number of "Bitcoin millionares" exist
  - Mined a bunch of bitcoins back in 2009
  - One guy threw away machine with private key for >$500K coins

Implications of Bitcoin/Discussion

- What is hard socially/economically?
- Why does Bitcoin have value?
- How to convert bitcoins to USD?
- Who pays for the infrastructure necessary for Bitcoin?
- How does Bitcoin affect monetary policy?
- How does Bitcoin impact laws and public policy?