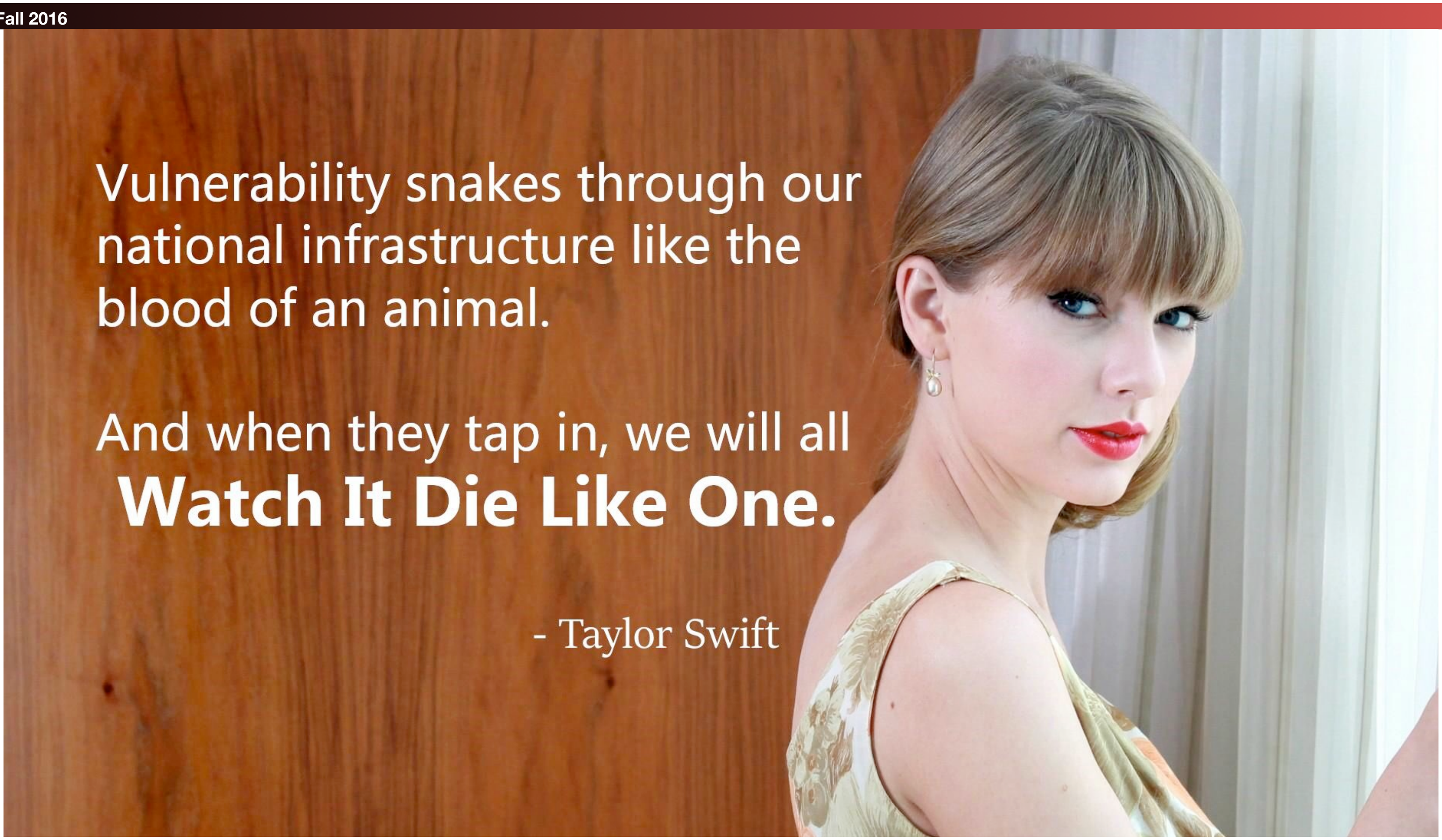
Applied Craptography: Bitcoin and Other Cryptocurrencies

Meme of the Day


Vulnerability snakes through our national infrastructure like the blood of an animal.
And when they tap in, we will all
Watch It Die Like One.
- Taylor Swift

Outline

- The Cryptography:
  - Hash Chains
  - Proof of Work
  - Putting it together: The Bitcoin Public Ledger
- The Craptography:
  - Irreversibility + Volatility -> Only good for crime
  - How to make money in Bitcoin: Theft
  - Gross inefficiencies
  - Public Data -> Clustering
  - Record History -> Prosecution Futures
  - Even more “coolness”: Ethereum
    - Aka “Lets program our dollar bills in JavaScript!” 😂 😂 😂 😂 😂 😂 😂

Bitcoin's Goal

- A decentralized, distributed digital currency
  - Decentralized: ***no point of authority or control***
  - Distributed: ***lots of independent systems, no central point of trust***
  - Digital Currency: ***Just that, a currency***
- Bitcoin is ***censorship resistant money***:
  - Nobody can say "don't spend your money on X"
- Bitcoin's Crypto: Interesting
- Bitcoin's Economics: Broken
- Bitcoin's Community: Bat-Shit Insane

Bitcoin's Public Key Signature Algorithm ECDSA

- Elliptic Curve Digital Signature Algorithm
  - So different math but conceptually similar to El Gamal and DSA
- 256b private key (32 bytes)
  - Public key is 65 bytes
- Bitcoin "address" is not the public key but the ***hash*** of the public key
  - RIPEMD-160(SHA-256(K_{pub}))
    - Why double hashing? Its a common weirdness in Bitcoin.
  - After adding a checksum and Base 58 encoding you get a "Bitcoin address" of type 1 you can send money to
  - 1FuckBTCqwBQexxs9jiuWTiZeoKfSo9Vyi is a valid address
    - I spent a lot of CPU time randomly generating private keys to find one that would match the desired prefix

Interesting Implications of Hashed Public Keys

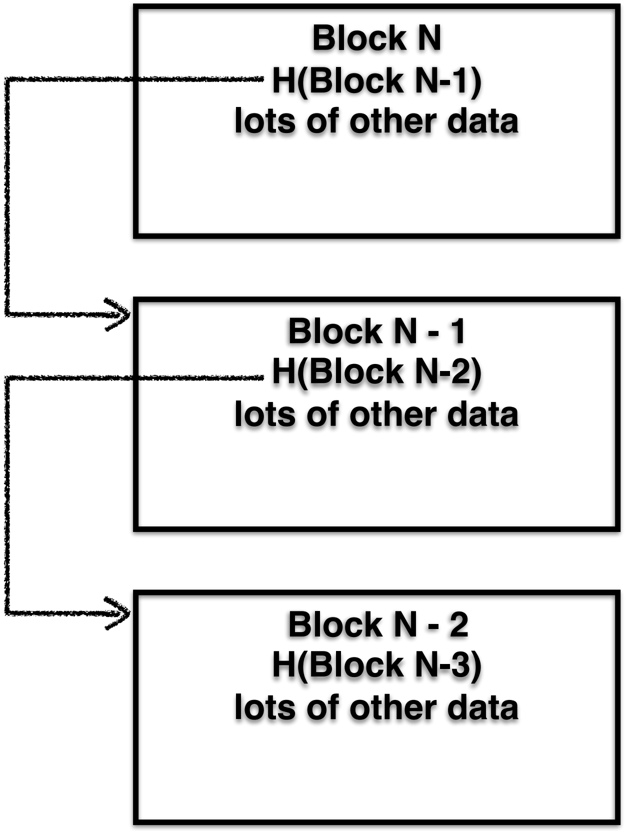
- The ECDSA public key is twice as large as the private key
  - So hashing makes the public key a lot smaller
  - But it makes the signatures themselves larger
    - Since any signature also needs to include the full public key
- Validation of a signature becomes a 2-part process
  - Validate that $H(K_{pub})$ = Address
  - Validate that the signature is valid
- But if a private key is only used ***once***, attacks which require the public key in advance can not work!

Why This Matters: Quantum Computing

- A Quantum computer rips through elliptic curve schemes as well as classic discrete log (Diffie/Hellman) and RSA type schemes
  - Given the public key it is trivial to find the private key
    - Since the private key controls money, this would be catastrophic
  - But at the same time, we don't know how to build a quantum computer big enough to factor a number much larger than 15
- If you ***never*** use a private key more than once…
  - By instead transferring all unspent money to a ***new*** random private key
  - A Quantum Computer can't steal your money!
- Many cryptographic systems need to worry today about Quantum computers which don't yet exist.

Hash Chains

- If a data structure includes a hash of the previous block of data
  - This forms a “hash chain”
- So rather than the hash of a block validating just the block
  - The inclusion of the previous block’s hash validates all the previous blocks
- This also makes it easy to add blocks to data structures
  - Only need to hash block + hash of previous block, rather than rehash everything:
    How you can efficiently hash an "append only" datastructure


Block N
H(Block N-1)
lots of other data

Block N - 1
H(Block N-2)
lots of other data

Block N - 2
H(Block N-3)
lots of other data

Merkle Trees

- Lets say you have a lot of elements
  - And you want to add or modify elements
- And you want to make the hash of the set easy to update
- Enter hash trees/merkle trees
  - Elements 0, 1, 2, 3, 4, 5...
  - H(0), H(1), H(2)...
  - H(H(0) + H(1)), H(H(2)+H(3))...
  - The final hash is the root of the top of the tree.
- And so on until you get to the root
  - Allows you to add an element and update lg(n) hashes Rather than having to rehash all the data

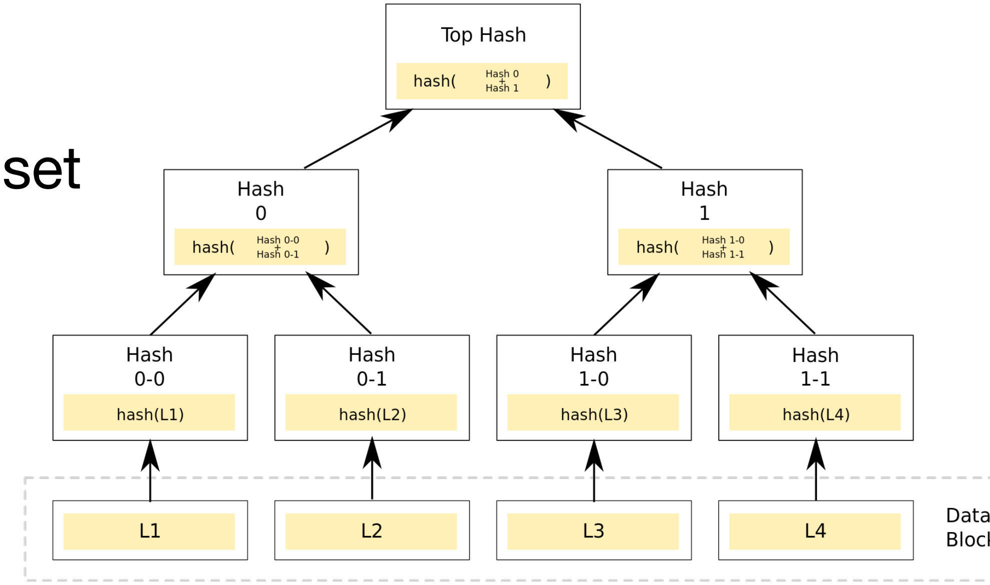


Image Stolen from Wikipedia

Proof of Work
To Establish History

- Idea: If creating a block requires so much effort
  - And it includes a pointer to all previous blocks
  - Changing history becomes expensive:
    - To rewrite the last ***k*** blocks of history requires the same amount of effort as recording those ***k*** blocks the first time around
  - But at the same time, it ***must*** be cheap to ***verify*** the work was done
- Easy proof of work: generation ***partial*** hash collisions
  - If the first ***N*** bits of a hash have to be zero…
    - You are expected to need to try ***2^N*** times to find a collision
    - But you only need to do a single hash invocation to ***check*** if someone else did the work

Taken Together this creates Bitcoin

- Every Bitcoin address (H(K_{pub})) has a corresponding balance in a public ledger (the Blockchain)
- To spend Bitcoin…
  - Sign a message saying “Pay to address A”
    - Signature includes the address it is coming from
  - Broadcast that message through the Bitcoin P2P network
- The rest of the P2P network…
  - Confirms that both the signature is valid and the balance exists
  - Then attempts to “mine” it into a new block on the Blockchain
    - This acts to ***confirm*** the transaction

Bitcoin Transactions

- A transaction consists of one or more inputs and 0 or more outputs
  - Each input refers to a single unspent transaction output:
    the input spends the ***entire*** output in the transaction
    - Each input is signed by the corresponding private key and includes the public key
  - Each output simply refers to a destination address and amount
    - If you want to make change, just send that to a new destination address or send it back to one of the input addresses
  - Sum(outputs) <= Sum(inputs)
    - Any extra is paid to whoever mines the block (the Transaction Fee)
- Validating transactions:
  - All inputs must refer to ***previously unspent outputs***
    - No double-spending, but requires knowing ALL previous Bitcoin transactions to validate!
  - All inputs must cryptographically validate

The Blockchain…
Protected by Proof of Work

- All Bitcoin miners take all unverified transactions they want and compose them into a single block
  - Block header contains a timestamp, a nonce, the hash of the previous block, and the hash of all transactions for this block
    - Transactions are hashed in a Merkle tree to make it easy to add transactions to the block in progress
- Now all the miners try to find a hash collision:
  - Modifying the block so that H(Block) < "difficulty" value
    - First by modifying the nonce value and/or timestamp and then modifying the coinbase
- Once one finds a hash collision, it broadcasts the new block to the entire Bitcoin network
  - Every other miner first verifies that block and then starts working on the next block
- Rule is always trust the longest chain
  - Now to rewrite history to depth N it takes the same amount of work as used to generate the chain you are rewriting
  - But at the same time, the current chain keeps growing!

The Coinbase Transaction

- The first transaction in any block is special
  - It actually has 0 inputs, instead it has a small amount of arbitrary data called the "coinbase"
- The coinbase data serves two purposes:
  - It allows the miner to make a comment
    - EG, claim credit, vote on proposals, etc
  - It can be easily changed for searching for hash collisions
    - When changing the coinbase now the miner needs to update the Merkel tree
- The output of this transaction is the miner's reward
  - The miner fills it out as "pay to me"
    - Both the current block reward (now at 12.5 BTC/block) and any value not otherwise spent

Bitcoin Balances

- Each address has a balance associated with it
  - The balance is in “Satoshi”, a fixed-point value = 0.00000001 BTC
    - There have been Bitcoin systems with bugs related to fixed vs floating point issues
- This is actually the sum of all unspent outputs sent to this address
  - Calculating an address's balance requires looking at ***every*** Bitcoin transaction ever done
- This is a ***problem!***
  - Bitcoin requires knowing every transaction from the dawn of the Blockchain in order to know that things are valid
    - And currently this data grows by 1 MB every 10 minutes!

Bitcoin Difficulty

- The effort needed for the proof of work dynamically adjusts
  - Every 1024 blocks the necessary difficulty changes
  - New difficulty is based on the previous blocks difficulty and timestamps
- This ensures a constant ***rate*** of block creation
  - New blocks are expected at a rate of 1 every 10 minutes
- Also implements Bitcoin's "Monetary policy":
  - Initially +50 BTC every 10 minutes
    - Block reward halves every few years
  - Ensures a constrained supply of Bitcoin: 21,000,000 maximum
- Also acts as a global rate limit on transactions!
  - Early on, blocks were capped at 1 MB to prevent possible "spam"
    - Building huge blocks to exhaust resources
  - But now it means Bitcoin has a global limit of <3 transactions per second!

Bitcoin and Spam...

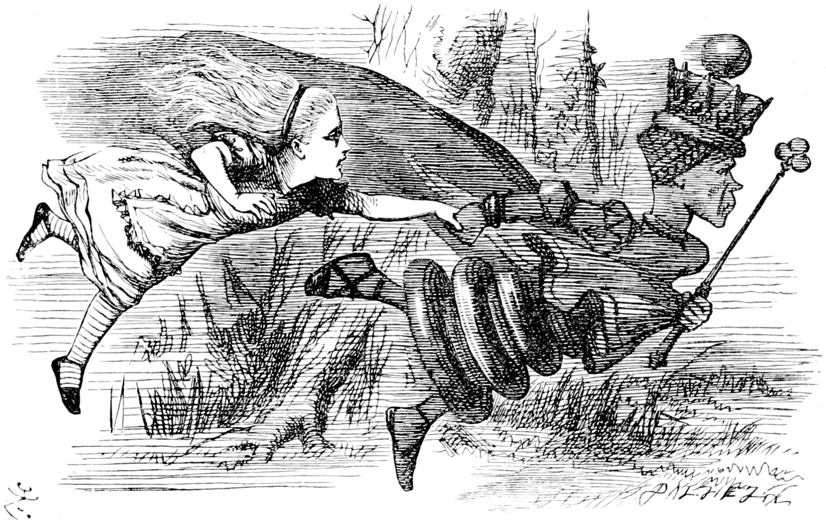
- Bitcoin has a current "block limit" set at 1MB
  - Can only add 1MB worth of transactions every 10 minutes
  - <3 transactions/second
    - This was designed to prevent a possible spam attack in the early days of Bitcoin
      - Meant to be a temporary expedient before a better solution
- Recently there is a debate about increasing this limit
  - A group calling for a larger limit have been "stress testing" Bitcoin by sending generally useless transactions
    - Effectively shuts down the network for anyone not willing to pay a higher fee than the spammers!
  - And now it just happens organically!

The Future of Bitcoin And Spam...

- With current blocksize:
  - Attackers can basically shut down the network at will with a fairly small monetary investment
    - Just charge slightly more than the transactions you want to kill
- With increased blocksize:
  - Can cause the global history to grow at TB/year by sending super cheap/free bad transactions
- With either:
  - Tune spam to avoid the inevitable spam-filters:
    will eventually cause false positives which block normal transactions!
- Reasonable government could spend a modest cost to effectively destabilize Bitcoin...

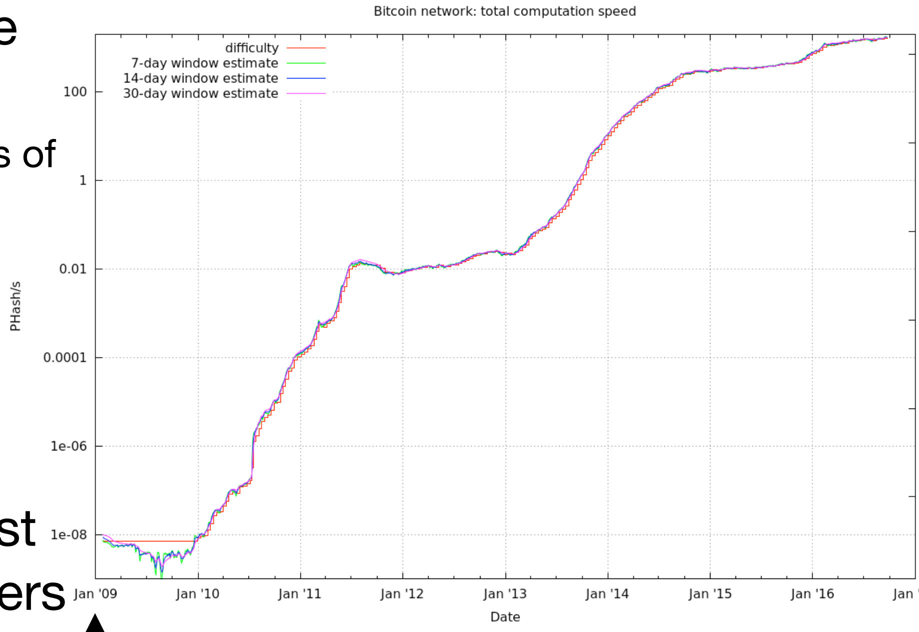
The Red Queen's Race

- Lets say you develop a Bitcoin "miner" than can try twice as many hashes/second
  - Initially you get more block rewards
- But now everybody else follows your lead...
  - And you are right back at the same spot you were before
- This cycle continues every time there is an upgraded mining technique



Economic Implications of The Red Queen's Race

- Any profitable mining strategies will attract more miners
  - Switch from CPUs to GPUs to FPGAs to multiple generations of ASICs
  - Current rate is astonishing 1500 PHash/s!
- Ends up having most reward ending up being spent on the cost of mining
  - Since as long as reward > cost, you get more miners!
- Now the "decentralized" mining system is almost entirely controlled by a few Chinese bitcoin miners
  - Latest generation ASICs with inexpensive design costs
  - Effectively no safety requirements for machine rooms
  - Cheap power



http://bitcoin.sipa.be/speed-lin-ever.png

Yes, that is 11 orders of magnitude!

This Means Bitcoin Transactions Are Incredibly Expensive...

- Each transaction may have just a small fee
  - Say $.10 or so to reliably be processed
- But there is the additional inflation "tax" in the block reward
  - Lowers the value of all existing Bitcoin
- So at $600/BTC and a 12.5 BTC block reward
  - The ***real*** transaction cost = (600 $/BTC * 12.5 BTC/block) / (1600 TX/block)
    - >$4.50 per transaction!?!?!?
- This is what ***really*** protects the Blockchain:
  - "Proof of burning Chinese coal!"

Hyper-***deflationary*** Currency

- New bitcoins are added at a fixed rate
  - Currently 1800 BTC/day
    - Exponential die off for a limit of 21M BTC
    - Possible to lose/destroy bitcoins
  - The ultimate Goldbug Monetary Policy
- If BitCoin has future value, why spend it today?
  - The value can only go up due to the fixed supply
  - The only rational thing for BitCoin believers to do is to hoard their BitCoins!
    - Buy, steal, mine, whatever. ***Just never spend them***!
- How can you have a currency that should never circulate?

Irreversibility

- Until a transaction is confirmed in a block you can't trust it
  - Since the sender could send a different transaction which, if confirmed first, would spend the money someplace else
    - And this happens: Many who accepts "0-confirmation" transactions has experienced this to some degree
  - Before confirmation, a Bitcoin transaction may as well be written in water
- But once it is confirmed it is effectively irreversible
  - If it is at depth N in the chain, the attacker would need to do an equal amount of work to change history!
  - After a few confirmations, a Bitcoin transaction is written in stone
- So once a transaction is accepted, there ***is no undo***

Irreversibility Implication: Cost

- You can't just transfer money from your bank account to buy Bitcoin
  - Because otherwise you could transfer money, take the Bitcoin, go "whoops", take it back
- Which means ***anytime*** you want to buy Bitcoin you either
  - Have to wait for a few days so that things can't be undone
  - Effectively purchasing on credit
  - Go through a "cash step":
    - Withdraw cash ***in person***
      - Its how banks handle necessary irreversibility, force it to be in person
    - Transfer the cash to the seller
      - Deposit into their account
      - Western Union
      - Face to face meeting in a Vegas casino...

The First Incarnation of Tradehill

- Tradehill was a BitCoin exchange based in the US
  - They accepted transfers using Dwolla
  - Dwolla is a “me-too” PayPal knockoff
    - Bank accounts only
    - Initially a no chargebacks allowed policy
      - Thus their play was to be more merchant friendly than the notorious chargeback-happy PayPal
- Dwolla changed their chargeback policy in June 2011 to add chargeback if they were charged-back
  - $90K clawed back from TradeHill for fraud
    $70K frozen against future chargebacks
- Tradehill goes bankrupt, sues for $2M in damages
  - Too bad there was a binding arbitration clause in the Dwolla contract....

Bitcoin will always be too-high friction for usability

- BitCoin transactions themselves are low-friction:
  - ~ $.15 or less
  - But “wait 10 minutes” (or often, considerably longer!) has a cost for real-world transactions
- But moving dollars into Bitcoin will always be high friction
  - At least 5% should be a reasonable assumption
    - A colleague’s experience suggests its even higher
  - Compare with Square’s 2.75% for accepting any credit card (including Amex)
- Yet volatility means receiver must convert back to dollars quickly
  - Most legitimate "buy with Bitcoin" sites are actually using a payment processor that ***immediately*** converts the Bitcoin back to Actual Money™ at a cost of 1%
- So the USD->BTC friction is the friction for BTC transactions

So the only real use is Censorship Resistant Money...

- Since Bitcoin is more expensive in practice than Actual Money™, why use it?
  - Apart from a political statement, that is
- Censorship resistant: ***no*** central authority that can say "Tho Shalt Not" spend money on
  - Drugs
  - Fake hitmen
  - Extortion schemes
- Bitcoin ***is*** the money of cybercrime:
  - $500k/day drug sales
  - ?? how much extortion/ransomware
  - <$2M day "real" sales

Drugs: Silk Road

- Silk Road was a TOR hidden service marketplace
  - Selling almost exclusively drugs
  - Mostly US centric
  - Only currency accepted is BitCoin
- Three main innovations:
  - TOR hidden service to prevent tracking & takedown
  - Mandatory feedback and escrow system
    - Silk Road sold ***trust***: You had to trust the market not the individuals
  - Optional currency hedge for sellers
    - Price can be tied to USD/BTC exchange rate
    - Payout on close-of-escrow is in constant USD, not BTC
    - Eliminate's seller volatility risk
    - Currency hedge used Mt Gox

Silk Road's Failure-Spinoff: The Armory

- The Silk Road operators were a bit more leery of guns
  - So they spun off guns/ammo/weapons into a separate site: ***The Armory***
- ?Unfortunately? the Armory failed in relatively short order. Why?
- Buying guns on The Armory is amazingly illegal for everyone
  - Bureau of Alcohol, Tobacco, Firearms, Explosives and Other Fun Things is quite strict about in-the-mail sales which bypass federal laws
  - Yet black market means black market prices!
- For 95% of US citizens, buying guns and ammo online amazingly legal
  - Ammo shipped to your door
  - Guns to your down-the-street Federal Firearm Licensed gun dealer

Extortion: Ransomware

- Probably the most, umm, ***exciting*** use for cryptography developed by the cybercrime underground
- After you infect a system...
  - Encrypt all the files...
    - Keep the mast key in memory for a while, so you can keep encrypting files, attached backup disks, etc
  - Then encrypt the master key with the extortionist's public key
  - "Pay $X or you'll never see your data again"
- Now there are crimeware kits for this
  - Pay $X and start your own cyber-crime business ransoming people's data...
- The problem is not infecting systems but getting ***paid***

Ransomware Payments

- It used to be ransomware offered both Green Dot and Bitcoin payment
  - Green Dot MoneyPak is a service which allows you to transfer money to reload prepaid credit cards for $6
    - You buy a card at 7-11 or the like
    - Cashout network developed in Europe for supporting criminal transfers
    - But a couple years back the US Treasury got Green Dot to clean up their act a lot
      - Now MoneyPak can ***only*** reload cards bound to real identities
  - Customers much preferred Green Dot
- Now it is Bitcoin only
  - And the customers hate it:
    Rumors of financial institutions buying Bitcoin in advance to deal with ransomware attacks!
- Unofficial homework assignment: Evaluate your and your parent's backup policies

Irreversibility Implication: Theft

- Electronic theft is a lot more pernicious than physical theft
  - To steal my wallet you have to get close to me
  - To steal from my computer you can be anywhere in the world
- Modern finance tries to prevent this with ***reversibility***
  - Until a small time passes, anything electronic ***must*** have an undo button and dispute resolution
  - This enables ***detection and mitigation***
- Bitcoin relies solely on ***theft prevention***
  - So Bitcoin is a lot easier to steal than Actual Money

How To Make Money in Bitcoin
In 10 Easy Steps

- Step 0: Move to Sochi
- Step 1: Break into blockchain.info and other web-wallet services
- Step 2: Download the saved web wallets for offline cracking
- Step 3: Modify the wallet service javascript to leak passwords
- Step 4: Be patient and wait
- Step 5: When discovered, steal all the Bitcoin

http://www.buttcoinfoundation.org/how-to-make-money-with-bitcoin-in-10-easy-steps/

How To Make Money in Bitcoin In 10 Easy Steps

- Step 6: Blame the victims
- Step 7: Write malcode to look for Bitcoin wallets
- Step 8: Crack away but wait before robbing them blind
- Step 9: Blame the victims
- Step 10: Enjoy life
- And if you get bored of retirement:
  - Step 11: tamper with the random number generators for a paper wallet service...

Upshot: You Can't Store Bitcoin on an Internet Connected Device!

- Yes, the "Internet of money" is not safe to use on the Internet
- Anyone who's serious about holding Bitcoin needs to hold in "cold storage"
  - The private keys stored in offline media, such as in USB keys or printed out on paper
- Anyone who fails this lesson gets robbed
  - The latest was BitFinex: $60M stolen from an exchange
    The exchange's response was to steal all the money from the customers...
    NO INSURANCE!

Yes, This Happened To Us!

- We set up a small Bitcoin wallet to install on our honeypots
  - Hoping to see if in running random malicious programs, one would try to steal our money...
- We also set up a small monitoring script
  - Using a Bitcoin service to monitor for any change in value
- A couple months later... All our money was stolen!
  - Yay, we detected an attacker, but...
  - It wasn't stolen from our honeypots!
- The grad student who set this up had a copy in his Dropbox account
  - Attacker managed to compromise it through a chain of attacks
  - Also stole $2k of general research funds in Bitcoin for other purposes

Know Your Threat Model:
Da Gubment Is Gonna Take Yer Money!!!

- One strain of Bitcoin advocacy is that because there is no control...
  - It is immune from government bulk seizure or similar black-helicopter scenarios
- Unfortunately one problem...
  - If this is your threat, you don't just need a store of value that resists the catastrophe...
  - You need a store of value that others will accept in that catastrophe...
- So if this is your threat:
  Don't invest in Bitcoin
- Invest in gold and .223

ECDSA stumbling block: Reusing the nonce (k-value)

- The ECDSA signature scheme has a little detail that is easy to screw up:
  - You don't just sign the message, you also have a ***nonce*** called *k* used in the signature
    - DSA is a variant on the El Gamal signature scheme, its roughly equivalent to the *r* used in El Gamal encryption
- If you ***ever*** sign two different messages with the same nonce…
  - It becomes trivial to recover the private key!
- And there was a bug in Android bitcoin code in 2013…
  - Well, actually the bug was in the random number generator library where it would occasionally return the ***same random number twice!?!?***
  - So if you did two back-to-back transactions…
- Somebody noticed this
  - And set up a bot to look for this automatically:
    When this happens the money is stolen!

Bitcoin's Ecological Damage...

- Since the Bitcoin network "earns" $45,000/hr in new Bitcoin...
  - And since the Red Queen's Race ensures that most of this earning goes to cover the cost of mining
  - Probably ~1/2 of this "earning" goes to power the mining farms!
    - Which are now centralized in China
- So Back of the Envelope: (45,000 $/hr * .5) / (.1 $/kwh)...
- Bitcoin consumes ~200 MW of electricity!
  - May be ~100MW, the price has spiked recently so the mining hasn't necessarily increased in lock step
- Still, that's significant:
  - UC Berkeley average power consumption ~60 MW averaged over the whole year
- But fortunately its not likely to grow worse

Bitcoin's Psychological Problem
“Bitcoin Savings and Trust”

- Super duper secret high yielding investment in BitCoin
  - Claiming an insane (~7% weekly) rate of return through some BitCoin-based super-duper-top-secret-codeword-specific investment
- A huge number of the active BitCoin community bought into it
  - Even while others were screaming Ponzi! PONZI!
- Developed side bets:
  - The director of BitCoin Magazine, Matthew Wright, bet a huge amount of BTC (10K BTC, $100K USD at the time, that he did not have!) that it was not a Ponzi scheme

Of Course it was a Ponzi Scheme

- And a big one: Notional value perhaps 500K BTC
  - Or 5% of all BitCoin at the time!?!?!??!!!
  - And of course Matt couldn't pay his bets either...
- The BitCoin community unmasked the anonymous account behind BS&T...
  - Trendon Shavers, of Texas
- But guess what the account name was...
- pirateat40!?!?!?!?!?!
  - There were even PPT: Pirate Pass-Through operations:
    Since pirateat40 would only allow select, large investors...

Bitcoin's Delusion of Anonymity

- Many people mistakenly call Bitcoin "anonymous" money...
  - But it is really ***pseudonymous***: Every wallet is a distinct pseudonym
  - And every transaction is public
- If someone always uses the same wallet
  - It is easy to identify them...
- But there are two heuristics that work well
  - Same Inputs -> Same Controller
    - With a minor exception, multiple inputs to the same transaction are controlled by the same person
  - Trace the change
    - Since a transaction must spend complete inputs, any change is also the same controller
    - Use a heuristic to detect

Clustering:
Now Available at a Police Department Near You

- A company ***Chainalysis*** sells Bitcoin clustering as a service
  - With additional tagging by doing test purchases/transfers
- Also a previous version is available free:
  - https://www.walletexplorer.com/
- So with a little bit of "ground truth"
  - E.g. a couple of test purchases...
  - It becomes quite obvious

Clustering in Practice:
The Dread Pirate Ulbricht

- When the FBI arrested Ross Ulbricht for running Silk Road...
  - They tackled him with his computer open in the library
  - Not only did he take notes on a criminal conspiracy...
  - But it also included all his own Bitcoins
    - A rather large fortune!
- The FBI seized those Bitcoins...
  - Transferring the ones from the Silk Road server first to one address
  - And later transferring the ones from Ulbricht's laptop to a second address

Ross Ulbrich's Lawyer Is A Drooling Idiot...

- In addition to throwing away the case elsewhere...
- He let lose with a fantastically bad opening statement
  - Basically: "You know that huge pile of Bitcoin on my client's computer? Yeah, that was legitimate Bitcoin trading..."
- My reaction: *BULLSHIT*
- So I created two clusters of Bitcoin
  - Silk Road and Ulbricht
    - All addresses which sent to the FBI seizure addresses
- Strong links:
  - 20% ***directly*** transferred from Silk Road
  - +40% ***strongly*** linked

But the Prosecution Did One Better: wallet.dat file

- The Bitcoin wallet.dat file holds all the private keys
  - And it ***never*** willingly deletes private keys:
    - After all, even if you have spent all the money in an address, it might still get more money later
  - So you don't need fancy clustering...
  - Just dump the addresses corresponding to the private keys!
- So the Feds did that...
  - Dumped all Silk Road and Ulbricht wallets
  - Showed that almost all Ulbricht's money came from Silk Road

The Latest Hotness: Ethereum

- Bitcoin has a limited amount of programmability
  - Inputs are actually small scripts, not just addresses
- Simple stack logic also allows some slightly more complicated versions:
  - "Pay to script hash" (the 3xxx addresses)
  - Rather than checking an address, you have to check that the script processes correctly
    - Including any signatures
  - Enables M of N multi signature escrow or similar options
- But that wasn't good enough for some...
  - Anyway, create a new crypto-currency, sell it to suckers, take the money and run is a common pastime, so this is a good excuse as any...
- Enter Ethereum

Ethereum: Lets Program "Smart" Contracts in a JavaScript (like) Language

- Ethereum executes a small virtual machine
  - And payment to a destination can invoke that ***destination*'s** program
    - Limited only by "gas": how much payment is desired
- The language itself is JavaScript like
  - And has a nasty property:
    In paying someone else, it invokes code outside itself
    - And this code can then recall whatever function called it!
  - At the same time, the cryptocurrency community tends to believe "code is law"
- The basis of some very interesting attacks

Attack #1:
DOS

- Idea: Find a bit of code that is cheap in terms of "gas" but expensive in practice
  - Something that nails disk I/O is a great choice, disk is expensive
- Now just "spend" a bunch of money to execute these transactions
  - And then grind the network to a halt!
- In this case, the EXTCODESIZE opcode which causes miners to search over disk

Attack #2:
The DAO

- Being the "code is law" types, many in the Ethereum community were happy to play with the DAO, a "Distributed Autonomous Organization"
  - Imagine a mutual fund who's investments were determined by consensus of the participants
    - Including the ability to split off and perform other actions
- It tended to be a bit of a "natural ponzi" scheme right from the start
  - Nearly 10% of all Ethereum was "invested" in "The DAO"

But of course there was a bug!

- An attacker could propose a split...
  - Which would split off just the attacker's portion, but...
  - The split process would ***first*** transfer the money to the attacker and only ***then*** reduce the balance
  - But in transferring the destination is ***simply calling another function, one written by the attacker!***
- So what the attacker did was simply have the "pay me" function request another spilt
  - Resulting in the attacker quickly draining almost the entire DAO funds into the attacker's account!
  - Time of Check to Time of Use
- More details here: http://hackingdistributed.com/2016/06/18/analysis-of-the-dao-exploit/

There are no Libertarians when their money is stolen

- On one hand, this ***abided by the rules of the DAO!***
  - After all, this is the point of a "smart" contract, if its in the contract it is allowed
- OTOH, this is why smart contracts are a dumb idea
  - Real world contracts have an exception mechanism: the judge
- So Ethereum split in two!
  - A large group decided to "revise history" and simply have the miners ignore the DAO theft
    - Since, well, they had a lot of money "invested" in the DAO
  - A smaller group kept history alive
    - Who, of course, did not
    - And the difficulty adjusted so that both chains grew at the same pace
- Now you can play interesting games
  - Someone pays you on one chain, but if its still valid on the other...
  - You can broadcast on the other chain as well

And now you know...

- Why I hate cryptocurrencies.
- It's the ultimate dotcom stock, minus the sock puppet.
    - Matthew O'Brien