Cryptocurrencies

CS 161 Spring 2022 Lecture 12

D J Capelis @djcapelis · Feb 19

Algorithms: I want to solve a problem
Data science: I want to understand a problem
AI: I want to solve a problem and not understand the solution
Blockchain: I want to be a problem
Why Talk About Cryptocurrencies?!?

• I am an actual **expert** in this area
  • It has been one of my research focuses for the past 9+ years!
  • Mining the space for academic papers and comedy gold since 2013

• But I want it to die in a fire!
  • There is effectively no value:
    • Private Blockchains are 20+ year old ideas
    • Public Blockchains are grossly inefficient in the name of "decentralization" without actually being decentralized!
      • And don't actually solve any problems other than those required to implement cryptocurrencies!
    • Cryptocurrencies don't work as currency unless you are a criminal!

• Yet it has refused to just go away

• And it touches on a lot of real world "security" issues that often have nothing to do with actual security!
This Lecture and Blue Slides...

• This lecture really is *mostly* a set of blue slides!
  • It really can't be tested on...

• Instead, think of this as an intellectual vaccination
  • Once you realize just how bleeding stupid that space is you will be immune from wasting your time on it

• Plus there are some concepts of relevance otherwise:
  • Hash chains, Merkle Trees, Sybils
Linked Lists Blockchains
And CryptoCurrencies

• “Blockchain Technology”
  • A fancy word for “Append-Only Data Structure”
    • That causes people’s eyes to glaze over and them to throw money at people
  • “Private/Permissioned Blockchain”:
    • A setup where only one or a limited number of systems are authorized to append to the log
    • AKA 20 year old, well known techniques
  • “Public/Permissionless Blockchain”:
    • Anybody can participate as appenders so there is supposedly no central authority:
      Difficulty comes in removing “sibyls”

• Cryptocurrencies
  • Things that don’t actually work as currencies…
Hash Chains

- If a data structure includes a hash of the previous block of data: This forms a “hash chain”
- So if you have a way of validating the ending 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
- Now just validate the head (e.g. with signatures) and voila!
  - All a “blockchain” is is a renamed hashchain!
    Linked timestamping services used this structure and were proposed back in 1990!
  - Certificate Revocation Lists are signed hash-chains
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 $\log(n)$ hashes
    Rather than having to rehash all the data
  - Patented in 1979!!

Image Stolen from Wikipedia
A Trivial Private Blockchain...

• We have a single server $s$, with keys $K_{pub}$ and $K_{priv}$...
  • And a git archive $g$... (in which we fix git to use SHA-256)

• Whenever we issue a pull request...
  • The server validates that the pull request meets the allowed criteria
  • Accepts the pull request
  • Signs the hash of the head...

• And that is it!
  • Git is an append only data structure, and by signing the new head, we have the server authenticating the entire archive!

• This is why “private” blockchain is not a revolution!!!
  • Anything that would benefit from an append-only, limited writer database already has one!
What Is A "Cryptocurrency"?

- A cryptocurrency is a tradable cryptographic token
  - The goal is to create irreversible electronic cash with no centralized trust: If Alice wants to pay Bob 200 Quatloos to pay off her losing bet on the Green thrall, there should be nobody else who can block or reverse this transfer

- Based on the notion of a public ledger (the "Blockchain")
  - A public shared document that says "Alice has 3021.1141 Quatloos, Bob has 21.13710 Quatloos, Carol has 1028.8120 Quatloos..."
  - People can only add items to the ledger ("append-only"), never remove items

- Big Idea: Alice writes and signs a check to Bob saying "I, Alice, Pay Bob 200 Quatloos"
  - This check then gets added to the public ledger so now everyone knows Alice now has 2821.1141 Quatloos and Bob has 221.13710 Quatloos
What Is A "Cryptocurrency"?
What Is A "Blockchain"  
(well, "Public" or "Permissionless" Blockchains)

- Everyone involved gathers up copies of the loose checks
  - For each check, validate that there are sufficient funds
  - Bundle all the checks up into a "block" and staple them together, with a pointer to the previous pile
- Everybody now does a lot of useless "work" that may eventually get lucky
  - The one that gets lucky staples this (which is in the form of a check saying "The system pays to ME the reward for success, the hash of the total stack is X") to the block as well, publishes this, and gets the reward
- Now everybody else knows this stapled pile of checks is now verified
  - So everybody starts on a new block, pointing to the previous block and gathers up the new checks that haven’t yet been processed
- Result is an **append only** data structure
  - Rewriting history to change/remove a transaction requires as much effort as spent to create history
What Is A "Blockchain"
(well, "Public" or "Permissionless" Blockchains)
What Is Bitcoin?

• Simply the first widespread development of this idea
  • A "Bitcoin wallet" is simply a collection of cryptographic keys
    • Private key $K_{\text{priv}}$: A secret value stored in the wallet
    • Public key $K_{\text{pub}}$: A public value that anybody is allowed to see, derived from the private key
  • The "Bitcoin Blockchain" is Bitcoin's particular implementation of the shared ledger

• Spending Bitcoin is simply writing a check and broadcasting it:
  • "Pay $K_{\text{pub}}=1\text{Ross5Np5doy4ajF9iGXzgKaC2Q3Pwwxv}$ the value 0.05212115 Bitcoin...
    And whoever validates this transaction gets 0.0005 Bitcoin"
    - Signed 1FuckBTCqwBQexxs9jiuWTiZeoKfSo9Vyi:
      • This is Bitcoin transaction $d6b24ab29fa8e8f2c43bb07a3437538507776a671d9301368b1a7a32107b7139$
## What Is Bitcoin?

**Summary**
- **Size**: 763 (bytes)
- **Weight**: 3052
- **Received Time**: 2015-02-04 21:15:16
- **Included In Blocks**: 341974 (2015-02-04 21:15:16 + 2 minutes)
- **Confirmations**: 180240 Confirmations

**Inputs and Outputs**
- **Total Input**: 0.05262115 BTC
- **Total Output**: 0.05212115 BTC
- **Fees**: 0.0005 BTC
- **Fee per byte**: 65.531 sat/B
- **Fee per weight unit**: 16.383 sat/WU
- **Estimated BTC Transacted**: 0.05212115 BTC

**Transactions**:
1. **Input**: 1FuckBTCqwbQexxx9jiuWTiZeekfSo9Vyi (0.05 BTC - Output)
2. **Input**: 1FuckBTCqwbQexxx9jiuWTiZeekfSo9Vyi (0.000018 BTC - Output)
3. **Input**: 1FuckBTCqwbQexxx9jiuWTiZeekfSo9Vyi (0.00235018 BTC - Output)
4. **Input**: 1FuckBTCqwbQexxx9jiuWTiZeekfSo9Vyi (0.00025497 BTC - Output)

**Outputs**:
1. **Output**: 1Ross5Np5doy4... (Free Ross Ulbricht 🇺🇸) - (Spent) 0.05212115 BTC

**Visualize**:
- **View Tree Chart**
What Is Bitcoin Mining?

- It is the particular instance used to protect the transaction history for Bitcoin
  - Based on SHA-256
- Every miner takes all the unconfirmed transactions and puts them into a block
  - The block has fixed capacity (currently 1MB), limiting the global rate to ~3-7 transactions per second, and also includes a timestamp
  - Also attaches the "pay me the block reward and all fees" check to the front (the "coinbase")
  - Also attaches the hash of the previous block (including by reference everything in the past)
- Then performs the "Proof of work" calculation
  - Just hashes the block, changing it trivially until the hash starts with enough 0s.
    - This is the "difficulty factor", which automatically adjusts to ensure that, worldwide, a new block is discovered roughly every 10 minutes
- On success it broadcasts the new block
So Proof of Work...

• Remember, SHA256 looks random...
  • So just tweak one bit and the output looks totally different

• So if I present to you a string and the corresponding hash that starts with \( n \) 0-bits...
  • I probably had to do \( 2^n \) hashes

• So you can trivially verify that I did a ton of useless work with just a single hash
  • So to rewrite the last \( k \) blocks of history you have to do as many hashes as were used to record the last \( k \) blocks in the first place
The Blockchain Size Problem

• In order to verify that Alice has a balance...
  • You have to potentially check *every transaction* back to the beginning of the chain

• Results in amazingly inefficient storage
  • Every full Bitcoin node needs access to the *entire* transaction history
    • Because the entire history is needed to validate the transaction
  • A "lightweight" node still needs to keep the headers for all history
    • And still has to ask for suitable information to verify each transaction it needs to verify

• So if we have 10,000 nodes, this means 10,000 copies of the Bitcoin Blockchain!
Corollary: The Blockchain Capacity Problem...

- To limit the blockchain growth to "just" 1 MB a block...
  - An early defense against possible spam
  - The resulting design for Bitcoin can only process 3-7 transactions per second worldwide!

- Which means any "Bitcoin takes over money" requires trusted, centralized entities that maintain databases...
  - Oh, yeah, those are called banks! We have "electronic money" as a result, and have had it for decades!

- Also results in price shocks
  - When desired transactions < block capacity, transactions are cheap
  - When desired transactions > block capacity, prices spiral up because of an inelastic supply
  - Unknown attacks have cause transaction price shocks for the lulz!
The Blockchain Power Problem

• The Bitcoin system consumes roughly 23 GW of power right now (or basically Thailand!)

• This is because Proof of Work creates a Red Queen's Race
  • As long as there is potential profit to be had you get an increase in capability
  • Efficiency gains get translated into more effort, not less power consumption: 10x the hashes doesn't mean 10x the bitcoin but just 10x the difficulty factor

• There is no way to reduce Bitcoin's power consumption without reducing Bitcoin's price or the block reward
  • It is this waste of energy that protects Bitcoin!
The Bitcoin Folks \textit{lie} about the power consumption...

- Claim this rescues "stranded power"
  - But this is the point of a power \textit{grid}: We ship electricity half-way across the country (Well, not to Texas because they have a separate grid so they can ignore federal regulations)

- Claim this incentivizes "green power"
  - But bitcoin mining wants 24/7/365 power ("base load")
    Base load power is only hydroelectric, fossil-fuel, or nuclear
  - And there really are no new spots for dams

- Oh, but other things burn power too...
  - Yeah, ALL data centers together is probably 2x-3x Bitcoin...
    But Bitcoin can only do 3-7 transactions per second on a \textit{WORLDWIDE BASIS}!
  - And unlike Bitcoin, data centers try to reduce the power consumption

- Tesla's $1.5B is really a $1.5B in "Destroy the Planet Inc"
  Annual Bitcoin CO$_2$ emission of $\sim 90$ Mt of CO$_2$ is equivalent to driving an F150 Raptor for $>120$ billion miles!
The Sybil Problem...

- There is a lot of talk about "consensus" algorithms in cryptocurrencies
  - How the system agrees on a common view of history
  - Bitcoin's is simple: "Longest Chain Wins"

- But Proof of Work is **not** about consensus:
  - It is about solving the sybil (fake node) problem...
    How do you prevent someone from just spinning up a gazillion "nodes"
    - Have each node have to contribute some resource!
  - "Proof of stake" is just another solution...
    Which requires your money to be easy to steal!
    Plus enshrines "he who has the gold, rules!"

- But there is an easier one: "Articulated Trust!"
  - Like the CAs: Use human-based agreements to agree on \( M \) trusted parties
    - Only \( \frac{1}{2}M + 1 \) need to actually be trustworthy!
  - Why aren't there cryptocurrencies like this?
    - Well, there are a lot that use this under the hood but don't talk about it...
    - But if you do this you have legal obligations as a money transmitter!
The Irreversibility Problem

• A challenge: Buy $1500 worth of Bitcoin **now**, without:
  • Needing $1500 cash in hand, transferring money to an individual, or having a preexisting relationship with an exchange

• You **can't!**: Everything electronic in modern banking is by design reversible except for cryptocurrencies
  • This is designed for fraud mitigation: Ooops, something bad, undo undo undo...

• So the seller of a Bitcoin either must...
  • Take another irreversible payment ("Cash Only")
  • Have an established relationship so they can safely extend the buyer credit
  • Take a deposit from the buyer and wait a couple days
The Theft Problem...

• Irreversibility also makes things very easy to steal
  • Compromise the private key & that is all it takes!
  • See "How to make money with Bitcoin in 10 easy steps" by your's truly

• Result: You can't store cryptocurrency on an Internet Connected Computer!
  • The best host-based IDS is an unsecured Bitcoin wallet
  • So instead you have hardware devices, paper wallets, and other schemes intended to safeguard cryptocurrency
    • It is worse than money under the mattress: Stealing money under the mattress requires physical access!

• But at the same time, Not your keys, not your bitcoin!
  • Unlike a bank there is no deposit insurance should the exchange get robbed
And Even More Security Landmines...

- The primary tool these-days for self-hosted wallets is browser extensions like Metamask
  - Used to interact with NFT marketplaces and other related issues

- But horribly opaque to use!
  - Phishing email just the other day used to sign blank-checks for NFT sales
  - Experts actually can't use it right!

- And just mention that word on Twitter...
  - And you will have plenty of "helpful" support bots trying to get your cryptocurrency!
The Decentralization Dream...

- "Trust Nobody"
  - The entire system is trustworthy but each actor is not
- Requires that there never be a small group that can change things...
- It is basically an article of faith that this is a good & necessary idea
  - But about the only thing it really buys is censorship-resistance
The Decentralization Reality

• Code is inevitably developed by only one or a few groups
  • And they can *and do* change it capriciously if it affects their money: When the Ethereum "DAO" theft occurred, the developers changed things to take *their* money back from the thief

• Rewarded mining centralizes
  • Especially with ASICs and "Stealth ASICs" for proof of work mining
  • And the miners can *and do cheat*, such as enable "double spending" attacks against gambling sites, or front-running in Ethereum

• Several just aren't decentralized at all
  • Trusted coordinator or seed nodes
  • Ability to override/freeze assets
The True Value of Cryptocurrencies: Censorship Resistance...

- There is (purportedly) no central authority to say "thou shalt not" or "thou shouldn't have"
  - Well, they exist but they don't care about your drug deals...
- If you believe there should be no central authorities...
  - Cryptocurrencies are the only solution for electronic payments
- But know this enables
  - Drug dealing, money laundering, crim2crim payments, gambling, attempts to hire hitmen etc...
  - Ease of theft of the cryptocurrencies themselves
  - Ransomware and extortion: estimates of several \textit{billion dollars a year}!
- And some minor "good" uses
  - Payments to Wikileaks and Backpage when they were under financial restrictions
Cryptocurrencies don't work unless you *need* censorship resistance

- *Any* volatile cryptocurrency transaction for real-world payments requires two currency conversion steps
  - It is the only way to remove the volatility risk
    - Which is why companies selling stuff aren’t actually using Bitcoin, but a service that turns BTC into Actual Money™
    - And thanks to the irreversibility problem, buying is expensive
  - But if you believe in the cryptocurrency, you must hodl!

- Result is that the promised financial applications (cheap remittances etc) can *never apply* in volatile currencies like Bitcoin
  - Really Bitcoin et al are *only* appropriate for buying drugs, paying ransoms, hiring fake hitmen, money laundering...
  - Otherwise, use PayPal, Venmo, Zelle, MPasa, Square, etc etc etc...
Worse:
Censorship Resistance Enables Crime

• Before the cybercrooks had Liberty Reserve and still have Webmoney...
  • But Liberty Reserve got shut down by the feds (a shutdown that really screwed up the black market hackers), and WebMoney is Russia-only

• So the only censorship alternative is cash
  • Which requires mass ($1M ≈ 10 kg) and physical proximity

• So the cryptocurrencies are the only game in town!
  • The drug dealers hated Bitcoin in 2013, and hate them all still, but it is the only thing that works
  • Ransomware used to be Green Dot & Bitcoin, but Green Dot was forced to clean up its act
  • Modern ransomware is a multi-billion-dollar industry enabled by Bitcoin payments
And "Stablecoins" are no better...

- Removing the two currency conversion steps requires eliminating volatility
- Building a stable cryptocurrency requires an entity to convert dollars to tokens and vice versa at par.
  AKA a "Bank" and "Banknotes"
  - Thus a centralized entity, so why bother with a "decentralized" blockchain? 😐
  - All other "algorithmic stablecoins" are snake oil that implode spectacularly
- There is now a choice for the bank
  - Either you become as regulated as PayPal & Visa
  - Or you have a "wildcat bank": This is banking in the 1800s
  - Or you have "Liberty Reserve" and the principals end up in jail
And The Big Stable-Coin, Tether, IS A FRAUD!!!

- Bitcoin's value is purely a speculative bubble
  - Somebody in the future will pay more than you paid today
- Bitcoin has a price equation based on supply/demand
  - New Bitcoin = (New $ + New Fake $s)
- Bubbles have been drive by fake $
  - 2013: Willy-Bot on MtGox:
    Created fake $ in deposit in the
    Magic The Gathering Online Exchange Bitcoin exchange, bought Bitcoin
  - 2017: Tether:
    A stablecoin which unbanked Bitcoin exchanges use since they can't access the banking system.
    Roughly 1/3rd of the price runup then
  - 2020-22: Tether AGAIN:
    The Tether Printer go BRRRRR. Now in a situation where real new $ is deeply negative as they are
    adding billions of "dollars" a week in Tether (and now Circle) to buy Bitcoin to back the Tether...
Practically Every Cryptocurrency is "Me Too" with some riff...

- There are lots of cryptocurrencies...
  - But in many ways they act the same: A public ledger structure and (perhaps) a purported decentralized nature

- Litecoin:
  - Bitcoin with a catchy slogan

- Dogecoin:
  - Bitcoin with a cool joke

- Ripple:
  - (Centralized) Bitcoin with unrelated settlement structures

- IOTA:
  - (Centralized) Bitcoin but with trinary math 🧙‍♂️ and roll-thy-own cryptography 🧙‍♀️ ?!?!?

- Monero:
  - Bitcoin with some better pseudonymity

- Zcash:
  - Bitcoin with real anonymity, err, "money laundering built in!"

- Ethereum:
  - Bitcoin with "smart contracts", unlicensed securities and million dollar bug bounties
Public Blockchain's Weak Security Guarantees

- "Public blockchains" protected by proof-of-whatever promise a "no central authorities" & "fully distributed trust" append-only data structure.
  - But this isn't the case!
- Any lottery-based reward creates mining pools
  - Which means a few entities can and do control things: 5 entities effectively control Bitcoin with >50% of the hashrate
- The code developers also can and do act as central authorities
  - When ~10% of Ethereum was stolen from the "DAO", the developers rolled out a fork to undo the theft
- And worse...
Proof of Work's Economic Unsoundness

- Idea: The system wastes $x$ per hour to defend against potential attackers
- If an attacker needs to change the last $n$ hours of history...
  - They will need to spend at least $nx$, which acts as a floor
- This puts a ceiling on security as well: an attacker doesn't need to spend much more than $nx$
  - If an attacker can make more than $nx$ for an attack, they will!
- And its grossly inefficient:
  - The system is wasting $x$ per hour whether or not it is under attack
- Oh, and there are services!
So The Security Must Be Either Weak or Inefficient

- Proof of work is provably wasteful
  - It *may* be possible to make "proof of stake" work, but that has different problems
- And there is no way to make proof of work cheap!
  - Proof of "whatever" protects up to the amount that "whatever" costs, *but not more!*
- So "articulated trust" is vastly cheaper
  - Take 10 trustworthy entities, each one has a Raspberry Pi that validates and signs transaction independently
    - In the end, 6 need to prove to be honest, but could easily process every Bitcoin transaction
    - This requires 100W of power and $500 worth of computers!, or 9 *orders of magnitude less power*
The Worm Problem....

• These cryptocurrencies form a closely connected peer-to-peer network
  • If you have an exploit that can compromise other nodes...
    You can make a self propagating attack (a "worm"), but do NOT DO SO

• Would be able to compromise **every node** in the P2P network in **seconds**
  • And you know that thing about "don't keep your cryptocurrency on an internet connected system"? Yeah, how many actually do that!

• Target a secondary cryptocurrency...
  • EG, Dogecoin is a fork of Luckycoin is a fork of Litecoin is a fork of Bitcoin....
  • With half a decade of **NO UPDATES!**
  • So search the post-fork Bitcoin code for indications of memory vulnerabilities
  • And write a worm that steals all the OTHER cryptocurrencies!
But wait, what about all the Venture Money!!!

- **Old VC model**
  - Invest in several companies
  - One or two end up thriving
  - Sell stock to the public in an IPO or sell to a larger company

- **New A16Z: Securities Fraud as a Business**
  - Invest in several "blockchain" startups
  - Startups issue new tokens promising something, eventually
    - These are unlicensed securities and this is blatantly illegal in the US, just not enforced by the SEC!
  - A16Z gets a ton of these tokens, sells to retail suckers
    - Ideally gets listed on Coinbase, but sketchier exchanges will do
  - If SEC ever wakes up...
    - It is the startups that committed the securities fraud, not A16Z! So they are safe with their money!
What About Non-Currency Blockchain Applications?

- Put A Bird Blockchain On It!
- "Private" or "Permissioned" Blockchain
  - Simply a cryptographically signed hashchain: Techniques known for **20+ years**!
  - The only value gained is you say "Blockchain" and idiots respond with "Take My Money!"
- "Public" Blockchains are grossly inefficient and can't actually deliver on what they promise
- And those proposing "blockchain" don't actually understand the problem space!
A couple years ago there was a "Blockchain" class here at Berkeley
  • Yes, I screamed inside
  • I attended one session to give a short rebuttal...
    • But the two outside "experts" also present were delusional

Concrete example: Vaccine supply chains...
  • You need to keep a vaccine supply chain suitably cold, if it gets too hot that is a problem...
  • One expert: "You can solve this in India with Blockchain!"

BULLSHIT! You solve this with temperature-sensitive labels! At $1.50 each

Proof of Nick's Iron Law of Blockchain:
Blockchain solves exactly one problem: When someone says "you can solve X with Blockchain", they clearly don't know anything about X and should be ignored
But There Is One Innovative New Stupidity: "Smart Contracts"

• Idea! "Contracts are expensive!" 😳
  • So lets take standard things written in a formal language ("Legaleze")
  • And replace them with things written in a horrid language (that looks vaguely like JavaScript)
    • By default these "smart contracts" are fixed once released!
    • And this makes things cheaper *how*?

• And ditch the exception handling mechanism
  • If you can steal from a Smart Contract, are you actually violating the contract?

• Backstory:
  Idea created by Vittalik Butterin who was upset that World of Warcraft nerfed his spellcaster!
"Smart Contract" Reality: Public Finance-Bots

- They are really Public Finance-Bots
  - Small programs that perform money transfers
    - Finance bots are not new:
      The novelty is these finance bots are public and publicly accessible
    - Oh, and these aren't "distributed apps"

- Predictable Result: Million Dollar Bug Bounties!
  - The "DAO", a "voted distributed mutual fund as smart contract":
    Got ~10% of Ethereum before someone stole all the money!
  - The "Parity Multi-Signature Wallet" (an arrangement to add multiple-signature control to reduce theft probability)
  - The "Proof of Weak Hands 1.0" explicit Ponzi Scheme
And "Decentralized Autonomous Organizations"

- Hey, let's get together and create an organization where we all invest and get a vote...
  - Yeah, this was invented centuries ago: It is called a "Joint Stock Corporation"
- But instead do it on a Blockchain...
  - So if something screws up, eh, ah well
- And not do the paperwork needed to actually **be** a corporation
  - Corporations have liability protections, investors aren't on the hook when a corporation commits crimes
- A better term is "Conspiracy"
The Rest Is Speedrunning
500 years of bad economics...

- Almost every cryptocurrency exchange is full of frauds banned in the 1930s
- Ponzi schemes without postal reply coupons, including explicit ponzies as "Smart Contracts"
- Tether, a "stablecoin" is almost certainly a wildcat bank from the 1800s
- Every tradable ICO is really an unregulated security just like the plagues in the South Sea Bubble of 1720
- Replicated rare tulips with rare cats on the Ethereum Blockchain as a "Smart Contract"! Time to party like it is 1637!
- And don't forget the goldbug-ism...
Smart Contracts and "Decentralized Finance": Speed Running the Speed Run

• "Hey, only Wall Street has previously benefitted from super-whiz-bangie techno innovations"
  • So lets instead build them as "Smart Contracts"?

• ONLY applications end up being:
  • Fraudulent stocks (ERC20 tokens)
  • Tulip Manias (Non-Fungible Tokens: A receipt for a URL saying 'I ownz this')
  • Implicit ponzi schemes ("Yield Farming")
  • Explicit ponzi schemes
  • Front-running bots and fraudulent miners
  • And million dollar thefts seemingly on a near-daily basis
    • Not sure which is more, the thefts or the frauds ("Rugpulls")?
And Now Rebranding: "Web 3"

• Hey, let's bring the **UNSTOPPABLE CENSORSHIP RESISTANT BLOCKCHAIN POWAH TO THE WEB**

• The current web: *distributed*
  • You need to contract with a DNS provider and a web hosting provider for a few bucks
    • If either dislike you they can censor you
    • But you can chose a friendly provider: Actual nazis can web host just fine, just not in Germany

• The computation in the current web:
  • A distributed computation split between the server and the user's browser

https://www.preethikasireddy.com/post/the-architecture-of-a-web-3-0-application
The Web 3 Vision: ADD On Additional Crap...

- You still have the centralized hosting!!!
  - So no gatekeepers were removed
- You end up depending on additional **centralized** providers!
- But now some of the computation is paid for in cryptocurrency and performed on the "blockchain"
  - Signed for by the customer's cryptocurrency wallet bolted onto the browser
So How Good Is The Ethereum Blockchain As A Computer

• Global Limit: 2 million "gas" per second
  • Any computation takes some "gas" as measured in the Ethereum Virtual Machine

• Simplest computation: 256b addition = 3 gas

• Ethereum Blockchain:
  • 600,000 additions per second
  • Cost to use? $250 a second!

• Raspberry Pi 4:
  • 3,000,000,000 additions per second
  • Cost to use? $45 to buy forever!
So The Space is Dismal

- The value is nonexistent
- The harms are great
- So avoid it...
- Or work on making it die in a fire
Case Study: Facebook Messenger Abuse Protocol
Content Warning

- The following is discussing Facebook's Messenger's abuse protocol
- Abusive behavior in messengers is dominated by gendered abuse
- This is a very useful protocol to understand:
  - It tackles real world issues and is a very nice example of applied cryptography
Facebook Messenger: Background

- Facebook Messenger now has an encrypted chat option
  - Limited to their phone app
  - The cryptography in general is very good
  - Uses a well-regarded asynchronous messenger library (from Signal) with good security properties
Facebook's Unique Messenger
Problem: Abuse

• Much of Facebook's biggest problem is dealing with abuse...
  • What if either Alice or Bob is a stalker, an a-hole, or otherwise problematic?
    • Aside: A huge amount of abuse is explicitly gender based, so I'm going to use "Alex" as the abuser and "Bailey" as the victim through the rest of this example

• Facebook would expect the other side to complain
  • And then perhaps Facebook would kick off the perpetrator for violating Facebook's Terms of Service

• But fake abuse complaints are also a problem
  • So can't just take them on face value

• And abusers might also want to release info publicly
  • Want sender to be able to denote to the public but not to Facebook
  • Deniability is in many ways anti-authentication: Want to make it so you don't have public key signatures
Facebook's Problem Quantified

- Unless Bailey forwards the unencrypted message to Facebook
  - Facebook must not be able to see the contents of the message
- If Bailey does forward the unencrypted message to Facebook
  - Facebook must ensure that the message is what Alex sent to Bailey
- Nobody but Facebook should be able to verify this: No public signatures!
  - Critical to prevent abusive release of messages to the public being verifiable: Messages are deniable for everybody but Facebook
What Is Bailey's Public Key?
The Protocol
In Action

{message=E(K_{pub_b},
M={"Hey Bailey: Abusive Message",
 k_{rand}}),
mac=HMAC(k_{rand}, M),
to=Bailey,
from=Alex,
time=now,
fbmac=HMAC(K_{fb}, {mac, from, to, time})}
Some Notes

- Facebook **can not** read the message or **even verify Alex's HMAC**
  - As the key for the HMAC is in the message itself

- Only Facebook knows their HMAC key
  - And its the only information Facebook **needs** to retain in this protocol: Everything else can be discarded

- Bailey upon receipt checks that Alex's HMAC is correct
  - Otherwise Bailey's messenger silently rejects the message
    - Forces Alex's messenger to be honest about the HMAC, **even thought Facebook never verified it**

- Bailey trusts Facebook when Facebook says the message is from Alex
  - Bailey does **not verify** a signature, because there is no signature to verify… But the Signal protocol uses an ephemeral key agreement so that implicitly verifies Alex as well
Now To Report Abuse

{Abuse{
  M={"Hey Bailey: Abusive Message",
     \text{k}_{\text{rand}}},
  \text{mac}=\text{HMAC}(\text{k}_{\text{rand}}, \text{M}),
  \text{to}=\text{Bailey},
  \text{from}=\text{Alex},
  \text{time}=\text{now},
  \text{fbmac}=\text{HMAC}(\text{K}_{\text{fb}},\{\text{mac}, \text{from},
     \text{to}, \text{time}\})}
Facebook's Verification

- First verify that Bailey correctly reported the message sent
  - Verify $\text{fbmac} = \text{HMAC}(K_{fb}, \{\text{mac}, \text{from}, \text{to}, \text{time}\})$
    - Only Facebook can do this verification since they keep $K_{fb}$ secret
  - This enables Facebook to confirm that this is the message that it relayed from Alex to Bailey
- Then verify that Bailey didn't tamper with the message
  - Verify $\text{mac} = \text{HMAC}(k_{\text{rand}}, \{M, k_{\text{rand}}\})$
  - Now Facebook knows this was sent from Alex to Bailey and can act accordingly
    - But Bailey can't prove that Alex sent this message to anyone other than Facebook
    - And Bailey can't tamper with the message because the HMAC is also a hash