Computer Security

13. Blockchain & Bitcoin

Paul Krzyzanowski

Rutgers University

Spring 2018

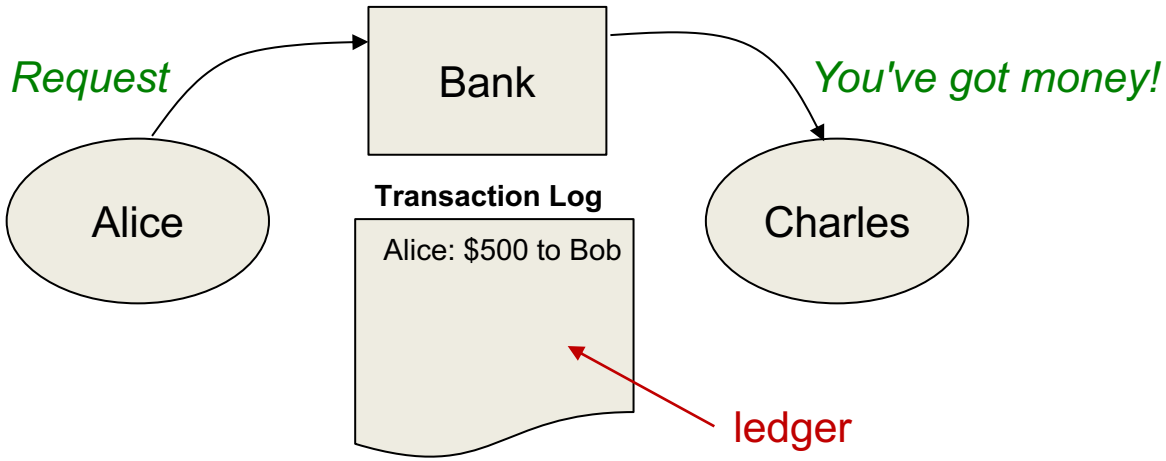
Bitcoin & Blockchain

Bitcoin cryptocurrency system

- Introduced in 2009 – anonymously by Satoshi Nakamoto
- First blockchain
- Designed to be public – anyone can participate in the system & use it

Traditional Payments

- Suppose Alice wants to pay Charles
- Send a message to the bank:
  - Transfer $500 from Alice to Charles
- Bank is a trusted third party
  - Owns register of activity
  - Only the bank can manipulate it
  - Also – controls supply of money



Centralized systems

- Transactions are simply modifications to the bank's database
- We can simply
  - Subtract $500 from Alice's account
  - Add $500 to Charles' account
  - The log is just nice for auditing but not necessary

Problems?

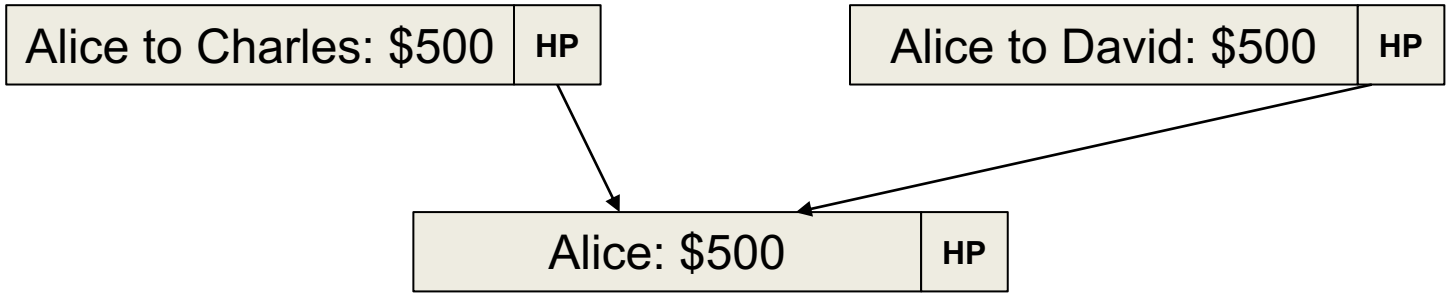
This is a centralized system

- What if the bank disappears?
- What if the banker makes a mistake?
- What if the banker is corrupt?

Double spending problem

We can create a decentralized solution

- Use hash pointers to track the movement of money
- Problem: double spending



Decentralized system

Can we create a payment system
that does not need a trusted third party (a bank)?

- Goal of the **blockchain**
  - No trusted third party
- A group of systems: *each keeps a copy of the ledger*
  - Everyone has information about all account activity
  - This will allow anyone to detect double spending
  - The bitcoin ledger is over 100 GB
- User identities are anonymous
  - Public key = user's identity (called an "address")
  - Transactions are associated with a specific user
    - Signed by that user's private key

The Distributed Ledger: the Block

- **Block** = partial list of transactions
- Group of participating systems that accepts transactions
- Start with an empty block
- If Alice (e.g., #1111) wants to pay Charles (e.g., #2222) $500
  - She tells everyone she wants to transfer $500 to #2222
  - Everyone checks their ledger to make sure Alice has enough money
  - Then they add the transaction to the block
  - And keep listening for more transactions
- When the block is full … or some time expires (10 minutes in Bitcoin)
  - We're ready to add it to the ledger
  - To do this, we need
    - Agreement on contents
    - Assurance that the contents will not be changed later

Securing the block

Hash functions are the key to tracking the integrity of the block

- One way functions
- Output gives us no clue of what the input is
- Efficient to compute & validate

Let's make in challenging: create a puzzle

Suppose we want a hash output with a specific property?

- Example, starting with "0000"?
- No algorithmic way to do this
- Must try lots of variations of the input
- But once found – it is easy for anyone to verify that the data hashes to the result

Mining

- Solving this "puzzle" is called **mining**
  - Have a number (bit field) in the block where we can set bit patterns
  - Try to get the block to hash to a desired output
  - The resulting number is called the **Proof of Work**

We demonstrate that work has been put into figuring out what the value should be to create the desired hash

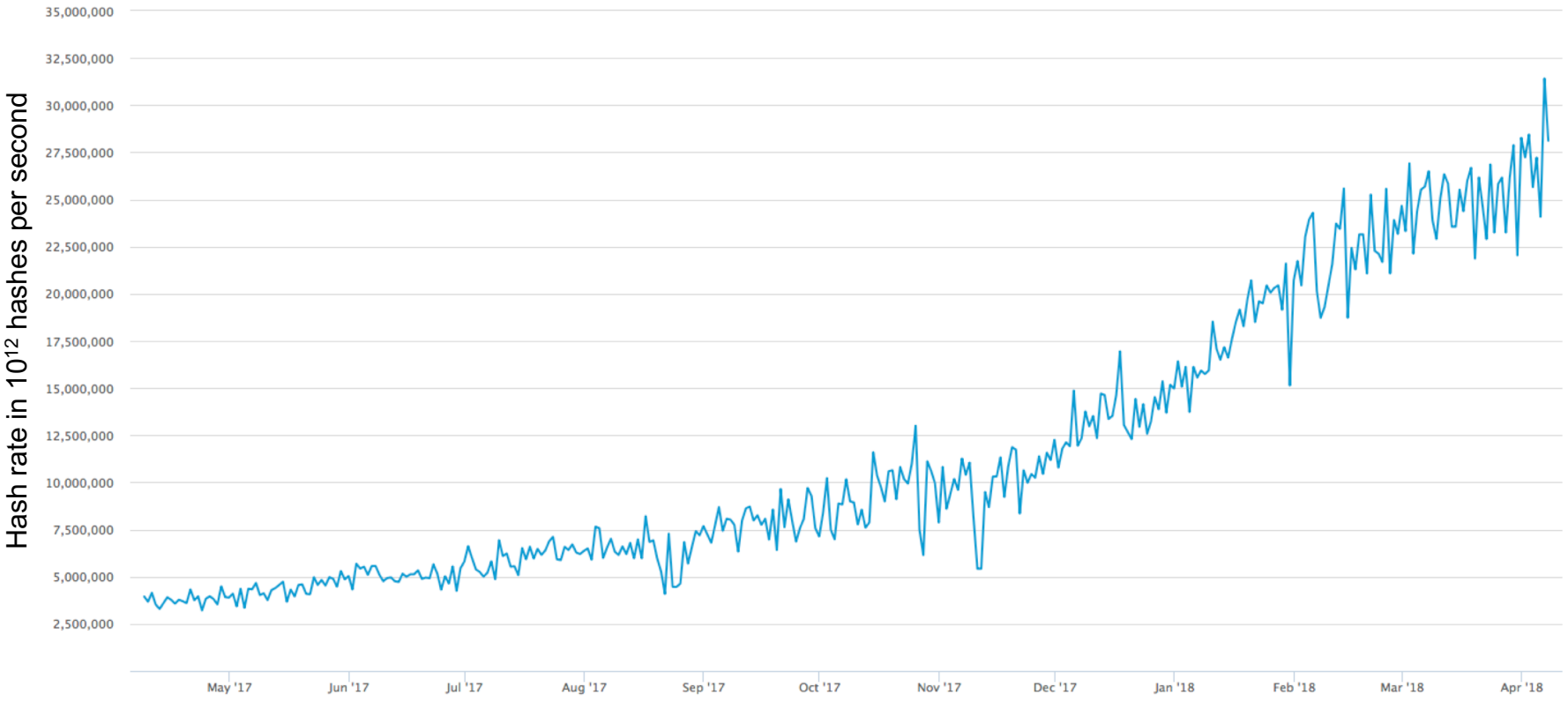
- Everyone in the network participates in this
  - The first system that finds it announces it to everyone else in the network
  - Upon receiving an announcement
    - Each system validates the Proof of Work number against the block
    - A majority of systems must grant approval
    - If they do, the block (with the Proof of Work) is made part of the **blockchain**

What's the puzzle?

- Bitcoin uses hashcash (created in 1997)
  - Hashcash searched for a *hash(message, random #, N)* where the leading k bits are 0
    - Random # - 128-bit starting value to make it unlikely that two systems start tat the same point
    - N – the number we vary until we get the hash we need
    - Choice of k sets the difficulty of the problem
- Ensure that one node doesn't take credit for another's work
  - 256-bit SHA-1 hash of
    - ***B***, transaction block, which includes hash pointer to previous block
    - ***A***, recipient's reward address (public key of who gets credit)
    - ***N*** – the number we vary until we get the hash we need
- Bitcoin uses a floating-point k to scale the work more precisely
  $hash(B, A, N) < 2^{n-k}$

How much work is going on?

Currently (April 2018), around 28-31x10^{18} hashes per second



See blochain.info/charts/hash-rate

The blockchain

Blockchain = sequence of blocks linked with hash pointers

- Hash pointer = { block ID, PoW }
- PoW = Proof of Work for that block = hash(block)
  - Hard to compute
  - Easy to verify

- There is **no authoritative copy** of the blockchain
  - Every participating node keeps a copy

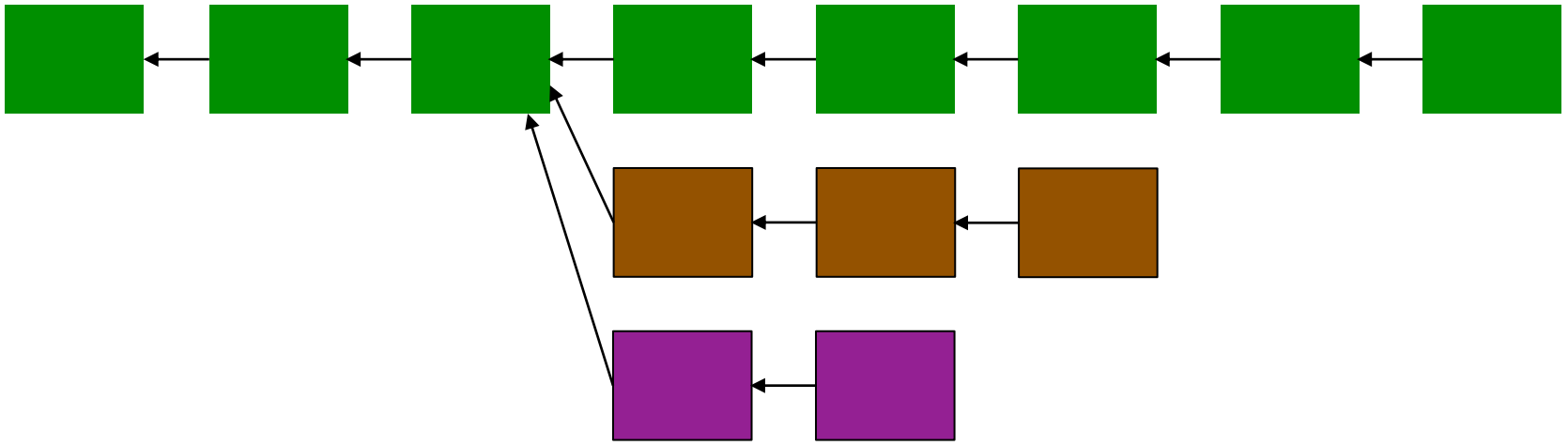
- Some participants may be faulty
  - A participant may have missed some transactions – data can get lost
  - There might have been errors on the system
  - A participant might be dishonest

- To remain a participant
  - You need to discard bad blocks and retrieve them from someone else

Competing chains

What if a malicious participants wants to modify an old transaction?

- Need to modify an old block
- Recompute the Proof of Work (which takes a lot of effort)
  for the block and each successive block (tons of work)
- This participant will be creating another chain in the blockchain



Competing chains

- BUT
  - One malicious participant will not be able to catch up with the cumulative work of all the others
  - It is expected that some nodes will occasionally have different versions
  - Length of chain = **score**
- If we observe two states of the blockchain, we select the one that was the hardest to generate (= longest chain)
  - Blockchain rules state that

    The longest chain in the network is the correct one

    Keep the highest-scoring (longest) version of the database
  - If a participant receives a higher-scoring version

    It overwrites its blockchain with the better data & transmits updates to peers

Producing a longer ledger than the current one requires computing power that competes with the rest of the entire network

Confirming transactions

- A transaction is ***confirmed*** after *N* number of additional blocks are added to the blockchain
  - Large values of *N* are recommended for high-value transactions

- *The more blocks are added after a transaction, the more difficult it is to modify it*

- Higher values of *N* mean that an attacker will need to recompute *N+1* Proof of Work values to modify the blockchain
  - Computationally not feasible

Bitcoin Confirmation Recommendations

1: Small payments <$1,000
3: Deposits and payments of $1,000-$10,000
6: Large payments $10k-$1M
60: Payments >$1M

https://www.buybitcoinworldwide.com/confirmations/

51% Attack

If the majority of participants decide to cheat, the protocol will fail

- Blockchain works only because of the assumption that the majority of participants are honest.

- To double-spend a bitcoin
  - You would need to rewrite the blockchain (change past transactions)
  - An attacker would need to control more than 50% of computing capacity
    - **This is a lot**: as of 12/17, The Economist estimates "*bitcoin miners now have 13,000 times more combined number- crunching power than the world's 500 biggest supercomputers*"
    - Even if someone tried to do this attack, they'd likely only modify transactions in the past few blocks
  - Keeping history of all transactions among all participants allows anyone to check for double spending

Incentives

Computing the Proof of Work takes a lot of work – *why do it?*

- For bitcoin:
  - First participant to compute the Proof of Work gets rewarded with bitcoin
  - BUT … only after another 99 blocks have been added to the ledger
  - This gives miners an incentive to participate & validate transactions
- Reward is decreasing (*assumption: bitcoins will be more valuable*)
  - 50 bitcoins for the first 4 years since 2008
  - 25 bitcoins from 2012-2015
  - 12.5 bitcoins from 2016-2019
- Eventually there will be a maximum of ~21 million bitcoins
- There are also transaction fees

Centralization

- Anyone can run a bitcoin node
  - Requires a good chunk of disk space but is accessible
  - Highly decentralized
- Mining
  - Anyone can mine but requires a lot of computing power
  - Not as decentralized as we'd like
- Software development/support
  - Open but there's a core set of trusted developers – not really decentralized
- In theory
  - Teams of sneaky developers may be able to mount an attack
  - Mining pools may try to mount a 51% attack
  - Both scenarios highly unlikely today

The end