The Mathematics of Bitcoin

Brian Heinold

What is Bitcoin?

- Created by Satoshi Nakamoto in 2008

What is Bitcoin?

- Created by Satoshi Nakamoto in 2008
- Digital currency (though not the first)

What is Bitcoin?

- Created by Satoshi Nakamoto in 2008
- Digital currency (though not the first)
- Decentralized

What is Bitcoin?

- Created by Satoshi Nakamoto in 2008
- Digital currency (though not the first)
- Decentralized
- Anonymous

What is Bitcoin?

- Created by Satoshi Nakamoto in 2008
- Digital currency (though not the first)
- Decentralized
- Anonymous
- Open to anyone to use (don't need a credit card issued from a bank, etc.)

What is Bitcoin?

- Created by Satoshi Nakamoto in 2008
- Digital currency (though not the first)
- Decentralized
- Anonymous
- Open to anyone to use (don't need a credit card issued from a bank, etc.)
- Electronic equivalent of cash

What is Bitcoin?

- Created by Satoshi Nakamoto in 2008
- Digital currency (though not the first)
- Decentralized
- Anonymous
- Open to anyone to use (don't need a credit card issued from a bank, etc.)
- Electronic equivalent of cash
- Price as of Saturday 3/28/15 at 7:50 pm

1 Bitcoin equals

253.48 US Dollar

Hash functions

- Bitcoin uses the SHA-256 *hash function*.

Hash functions

- Bitcoin uses the SHA-256 *hash function*.
- It takes an input of any size and returns a 256-bit output.

Hash functions

- Bitcoin uses the SHA-256 *hash function*.
- It takes an input of any size and returns a 256-bit output.
- Hash of "hello":
  `2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73043362938b9824`
- Hash of Hamlet's soliloquy:
  `fc95b400e3307f60f1ad0156c4ff1dba9b4050d11140e07f8c25342820344a0f`

Hash functions

- Bitcoin uses the SHA-256 *hash function*.
- It takes an input of any size and returns a 256-bit output.
- Hash of "hello":
  `2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73043362938b9824`
- Hash of Hamlet's soliloquy:
  `fc95b400e3307f60f1ad0156c4ff1dba9b4050d11140e07f8c25342820344a0f`
- 256 bits means $2^{256} \approx 10^{77}$ possible outputs.

Hash functions, continued

Properties of a good cryptographic hash function:

- Returns essentially random output
- Extremely difficult to reverse
- Shouldn't leak info about the input
- Extremely rare for two messages to hash to same value

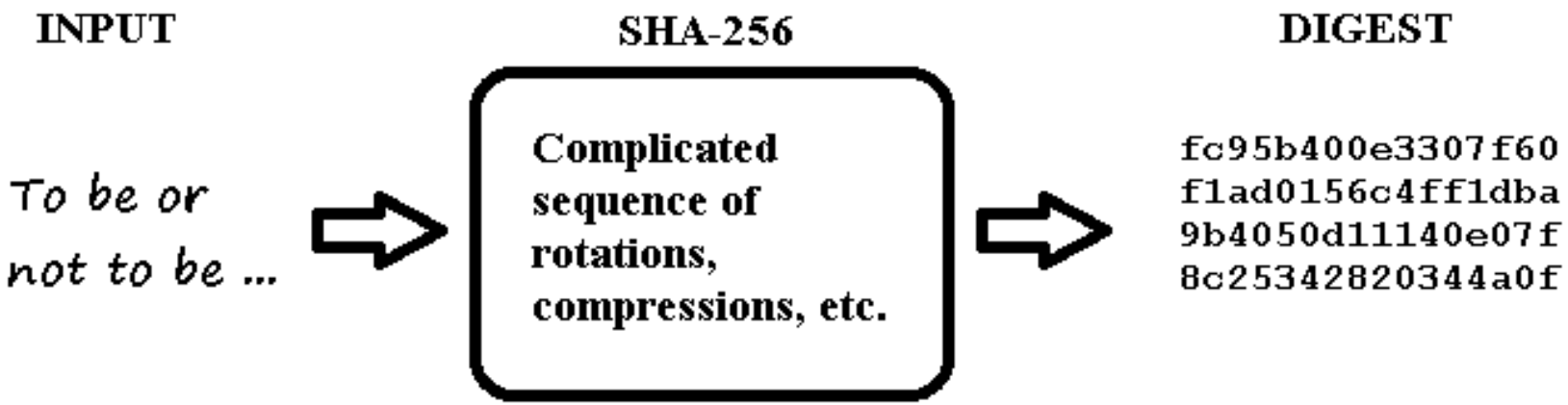
Hash functions, continued

Properties of a good cryptographic hash function:

- Returns essentially random output
- Extremely difficult to reverse
- Shouldn't leak info about the input
- Extremely rare for two messages to hash to same value

Assuming SHA-256 is secure, you would need around 10^{40} messages before two messages had same hash.

Hash functions, continued



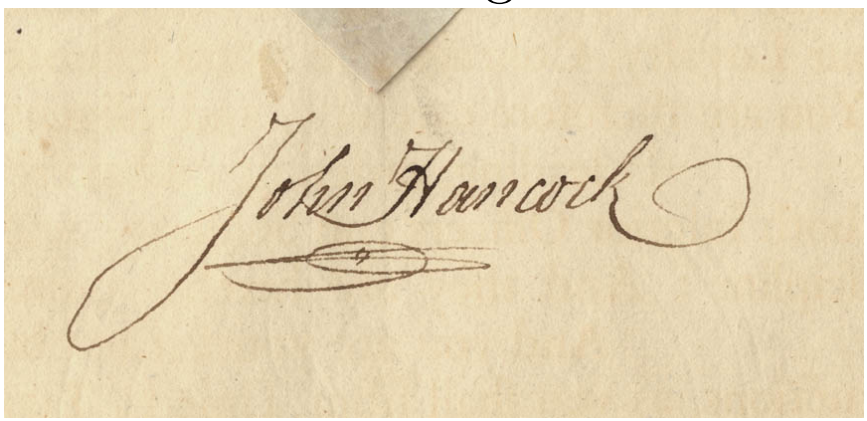
The hash's output (*digest*) serves as a short fingerprint of something.

Digital Signatures

- The problem: Alice wants to send stuff to people and people want to be sure that it really did come from Alice.

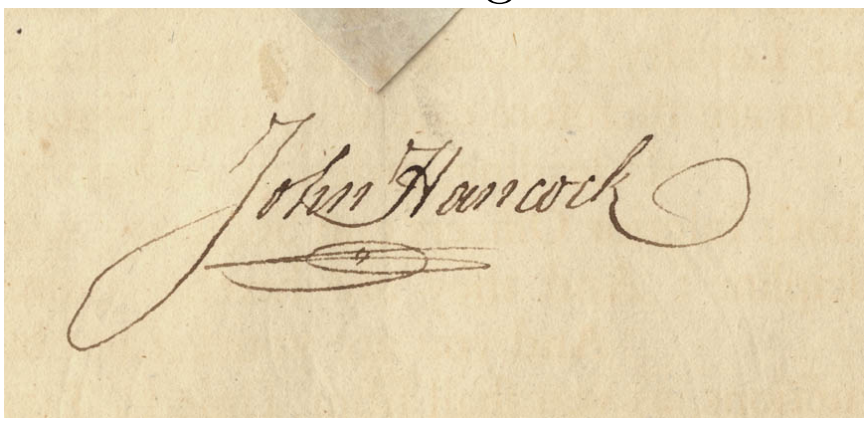
Digital Signatures

- The problem: Alice wants to send stuff to people and people want to be sure that it really did come from Alice.
- The solution: A signature



Digital Signatures

- The problem: Alice wants to send stuff to people and people want to be sure that it really did come from Alice.
- The solution: A signature



- We want a digital equivalent.
- Ordinary signatures can be forged and don't translate well to the digital world.

Digital signatures, continued

- Suppose Alice (and only Alice) has a secret pen only visible under particular wavelength of light.

Digital signatures, continued

- Suppose Alice (and only Alice) has a secret pen only visible under particular wavelength of light.
- Anyone can verify a message comes from Alice by shining a light of the right wavelength on the message.

Digital signatures, continued

- Suppose Alice (and only Alice) has a secret pen only visible under particular wavelength of light.
- Anyone can verify a message comes from Alice by shining a light of the right wavelength on the message.
- This is idea of public key cryptography.

  Public key: Fancy light
  Private key: Alice's special pen

Digital signatures, continued

- But instead of pens, use math.

Digital signatures, continued

- But instead of pens, use math.
- RSA digital signature scheme:
  - Alice generates two secret, huge primes p and q and multiplies them together to get $n = pq$.

Digital signatures, continued

- But instead of pens, use math.
- RSA digital signature scheme:
  - Alice generates two secret, huge primes p and q and multiplies them together to get $n = pq$.
  - She chooses an integer e and computes its "inverse", d.

Digital signatures, continued

- But instead of pens, use math.
- RSA digital signature scheme:
  - Alice generates two secret, huge primes p and q and multiplies them together to get $n = pq$.
  - She chooses an integer e and computes its "inverse", d.
  - public key: e, n (the fancy light)
    private key: d (the special pen)

Digital signatures, continued

- But instead of pens, use math.
- RSA digital signature scheme:
  - Alice generates two secret, huge primes p and q and multiplies them together to get $n = pq$.
  - She chooses an integer e and computes its "inverse", d.
  - public key: e, n (the fancy light)
    private key: d (the special pen)
  - Alice signs message m by by computing $m^d \bmod n$ (actually uses hash of m in place of m)

Digital signatures, continued

- But instead of pens, use math.
- RSA digital signature scheme:
  - Alice generates two secret, huge primes p and q and multiplies them together to get $n = pq$.
  - She chooses an integer e and computes its "inverse", d.
  - public key: e, n (the fancy light)
    private key: d (the special pen)
  - Alice signs message m by by computing m^d mod n (actually uses hash of m in place of m)
  - Signature:

    `21809167e3435e2256fd1005832a9219587c89abb8d2a4364103cc6e96`

Digital signatures, continued

- But instead of pens, use math.
- RSA digital signature scheme:
  - Alice generates two secret, huge primes p and q and multiplies them together to get $n = pq$.
  - She chooses an integer e and computes its "inverse", d.
  - public key: e, n (the fancy light)
    private key: d (the special pen)
  - Alice signs message m by by computing m^d mod n (actually uses hash of m in place of m)
  - Signature:
    `21809167e3435e2256fd1005832a9219587c89abb8d2a4364103cc6e96`
  - People verify signature by raising it to e and modding by n

Digital signatures, continued

- But instead of pens, use math.
- RSA digital signature scheme:
  - Alice generates two secret, huge primes p and q and multiplies them together to get $n = pq$.
  - She chooses an integer e and computes its "inverse", d.
  - public key: e, n (the fancy light)
    private key: d (the special pen)
  - Alice signs message m by by computing m^d mod n (actually uses hash of m in place of m)
  - Signature:
    `21809167e3435e2256fd1005832a9219587c89abb8d2a4364103cc6e96`
  - People verify signature by raising it to e and modding by n
  - Result should equal hash of m, which they can verify

Digital signatures, continued

- But instead of pens, use math.
- RSA digital signature scheme:
  - Alice generates two secret, huge primes p and q and multiplies them together to get $n = pq$.
  - She chooses an integer e and computes its "inverse", d.
  - public key: e, n (the fancy light)
    private key: d (the special pen)
  - Alice signs message m by by computing m^d mod n (actually uses hash of m in place of m)
  - Signature:
    `21809167e3435e2256fd1005832a9219587c89abb8d2a4364103cc6e96`
  - People verify signature by raising it to e and modding by n
  - Result should equal hash of m, which they can verify

Digital Signatures and Bitcoin

- Bitcoin uses related digital signature scheme, ECDSA.

Digital Signatures and Bitcoin

- Bitcoin uses related digital signature scheme, ECDSA.
- Relies on similar principles to RSA, but uses elliptic curves instead of primes and modular arithmetic

Digital Signatures and Bitcoin

- Bitcoin uses related digital signature scheme, ECDSA.
- Relies on similar principles to RSA, but uses elliptic curves instead of primes and modular arithmetic
- In Bitcoin, every user has both a public key and a private key. These are large numbers (given in base-58).

Digital Signatures and Bitcoin

- Bitcoin uses related digital signature scheme, ECDSA.
- Relies on similar principles to RSA, but uses elliptic curves instead of primes and modular arithmetic
- In Bitcoin, every user has both a public key and a private key. These are large numbers (given in base-58).
- Example public key: `1JHhJuduB86iVzSuubRhwTnV5stimPGQoD`
- Example private key:

  `5JNfwXtrJisnysV84QgkqBr4PwprEX7QPFg6f5jcdDkzT6erXGc`

Digital Signatures and Bitcoin

- Bitcoin uses related digital signature scheme, ECDSA.
- Relies on similar principles to RSA, but uses elliptic curves instead of primes and modular arithmetic
- In Bitcoin, every user has both a public key and a private key. These are large numbers (given in base-58).
- Example public key: `1JHhJuduB86iVzSuubRhwTnV5stimPGQoD`
- Example private key:

  `5JNfwXtrJisnysV84QgkqBr4PwprEX7QPFg6f5jcdDkzT6erXGc`
- Alice signs message by encrypting a hash of it with her private key.
- Other people decrypt it with the public key, and run message through hash function.
- Digest will match decrypted result only if it was sent by Alice.

Bitcoin transactions

- Say Alice wants to send some bitcoins to Bob.

Bitcoin transactions

- Say Alice wants to send some bitcoins to Bob.
- If this was a centralized currency, Alice would sign a transaction request and central authority would do all the work.

Bitcoin transactions

- Say Alice wants to send some bitcoins to Bob.
- If this was a centralized currency, Alice would sign a transaction request and central authority would do all the work.
- Central authority – single point of failure, requires users to trust it.

Bitcoin transactions

- Say Alice wants to send some bitcoins to Bob.
- If this was a centralized currency, Alice would sign a transaction request and central authority would do all the work.
- Central authority – single point of failure, requires users to trust it.
- Bitcoin is decentralized, relying on a peer-to-peer network.

Bitcoin transactions

- Say Alice wants to send some bitcoins to Bob.
- If this was a centralized currency, Alice would sign a transaction request and central authority would do all the work.
- Central authority – single point of failure, requires users to trust it.
- Bitcoin is decentralized, relying on a peer-to-peer network.
- Without central authority, When Alice wants to do a transaction, how does it get recorded?

Bitcoin transactions

- Say Alice wants to send some bitcoins to Bob.
- If this was a centralized currency, Alice would sign a transaction request and central authority would do all the work.
- Central authority – single point of failure, requires users to trust it.
- Bitcoin is decentralized, relying on a peer-to-peer network.
- Without central authority, When Alice wants to do a transaction, how does it get recorded?
- And what is to stop Alice from double-spending?

Bitcoin transactions, continued

- Alice's bitcoins are just entries in a decentralized ledger that everyone on bitcoin network has access to.

Bitcoin transactions, continued

- Alice's bitcoins are just entries in a decentralized ledger that everyone on bitcoin network has access to.
- When connecting to bitcoin network, download copy of ledger from peers.

Bitcoin transactions, continued

- Alice's bitcoins are just entries in a decentralized ledger that everyone on bitcoin network has access to.
- When connecting to bitcoin network, download copy of ledger from peers.
- There is no physical coin or even virtual coin. It is just an entry in that ledger.

Bitcoin transactions, continued

- Alice's bitcoins are just entries in a decentralized ledger that everyone on bitcoin network has access to.
- When connecting to bitcoin network, download copy of ledger from peers.
- There is no physical coin or even virtual coin. It is just an entry in that ledger.
- All transactions are publicly visible to everyone else (not what was bought, just how many coins were transferred).

Bitcoin transactions, continued

- Alice's bitcoins are just entries in a decentralized ledger that everyone on bitcoin network has access to.
- When connecting to bitcoin network, download copy of ledger from peers.
- There is no physical coin or even virtual coin. It is just an entry in that ledger.
- All transactions are publicly visible to everyone else (not what was bought, just how many coins were transferred).
- Alice is only known by her public key, not by her name.

Bitcoin transactions, continued

- Alice's bitcoins are just entries in a decentralized ledger that everyone on bitcoin network has access to.
- When connecting to bitcoin network, download copy of ledger from peers.
- There is no physical coin or even virtual coin. It is just an entry in that ledger.
- All transactions are publicly visible to everyone else (not what was bought, just how many coins were transferred).
- Alice is only known by her public key, not by her name.
- Transfer bitcoins by signing transactions with private key.

Bitcoin transactions, continued

- Alice's bitcoins are just entries in a decentralized ledger that everyone on bitcoin network has access to.
- When connecting to bitcoin network, download copy of ledger from peers.
- There is no physical coin or even virtual coin. It is just an entry in that ledger.
- All transactions are publicly visible to everyone else (not what was bought, just how many coins were transferred).
- Alice is only known by her public key, not by her name.
- Transfer bitcoins by signing transactions with private key.
- Lose private key, lose bitcoins.

Bitcoin transactions, continued

- Alice's bitcoins are just entries in a decentralized ledger that everyone on bitcoin network has access to.
- When connecting to bitcoin network, download copy of ledger from peers.
- There is no physical coin or even virtual coin. It is just an entry in that ledger.
- All transactions are publicly visible to everyone else (not what was bought, just how many coins were transferred).
- Alice is only known by her public key, not by her name.
- Transfer bitcoins by signing transactions with private key.
- Lose private key, lose bitcoins.
- Can store private key in bitcoin wallet software or in "cold storage" (offline)

Transaction details

- Say Alice wants to transfer 15 BTC to Bob.

Transaction details

- Say Alice wants to transfer 15 BTC to Bob.
- Maybe she has 20 bitcoins:

Transaction details

- Say Alice wants to transfer 15 BTC to Bob.
- Maybe she has 20 bitcoins:
  - 3 obtained in a transaction from Carol
  - 5 obtained in a transaction from Dave
  - 12 obtained in a transaction from Ellen

(Note: can't break apart transactions.)

Transaction details

- Say Alice wants to transfer 15 BTC to Bob.
- Maybe she has 20 bitcoins:
  - 3 obtained in a transaction from Carol
  - 5 obtained in a transaction from Dave
  - 12 obtained in a transaction from Ellen

  (Note: can't break apart transactions.)
- Each of those is a record in the public ledger.

Transaction details

- Say Alice wants to transfer 15 BTC to Bob.
- Maybe she has 20 bitcoins:
  - 3 obtained in a transaction from Carol
  - 5 obtained in a transaction from Dave
  - 12 obtained in a transaction from Ellen

  (Note: can't break apart transactions.)
- Each of those is a record in the public ledger.
- Alice hashes the transaction records of those.

Transaction details

- Say Alice wants to transfer 15 BTC to Bob.
- Maybe she has 20 bitcoins:
  - 3 obtained in a transaction from Carol
  - 5 obtained in a transaction from Dave
  - 12 obtained in a transaction from Ellen

  (Note: can't break apart transactions.)
- Each of those is a record in the public ledger.
- Alice hashes the transaction records of those.
- She also specifies Bob (i.e. Bob's public key), how much Bob gets, how much change she needs, and the transaction fee.

Transaction details

- Say Alice wants to transfer 15 BTC to Bob.
- Maybe she has 20 bitcoins:
  - 3 obtained in a transaction from Carol
  - 5 obtained in a transaction from Dave
  - 12 obtained in a transaction from Ellen

  (Note: can't break apart transactions.)
- Each of those is a record in the public ledger.
- Alice hashes the transaction records of those.
- She also specifies Bob (i.e. Bob's public key), how much Bob gets, how much change she needs, and the transaction fee.
- Signs it all with her private key.

Transaction details

- Say Alice wants to transfer 15 BTC to Bob.
- Maybe she has 20 bitcoins:
  - 3 obtained in a transaction from Carol
  - 5 obtained in a transaction from Dave
  - 12 obtained in a transaction from Ellen

  (Note: can't break apart transactions.)
- Each of those is a record in the public ledger.
- Alice hashes the transaction records of those.
- She also specifies Bob (i.e. Bob's public key), how much Bob gets, how much change she needs, and the transaction fee.
- Signs it all with her private key.
- Then she broadcasts it to the whole network.

Mining

- Broadcast is picked up by other nodes in the network, specifically *miners*.

Mining

- Broadcast is picked up by other nodes in the network, specifically *miners*.
- They verify transaction is valid by checking signature to make sure it's Alice, they check public ledger to make sure Alice really does have the bitcoins and hasn't already spent them.

Mining

- Broadcast is picked up by other nodes in the network, specifically *miners*.
- They verify transaction is valid by checking signature to make sure it's Alice, they check public ledger to make sure Alice really does have the bitcoins and hasn't already spent them.
- Questions:
  - How to record the transaction?
  - How will all the millions of users agree to record this?
  - How to make sure people don't cheat the system?

Mining

- Broadcast is picked up by other nodes in the network, specifically *miners*.
- They verify transaction is valid by checking signature to make sure it's Alice, they check public ledger to make sure Alice really does have the bitcoins and hasn't already spent them.
- Questions:
  - How to record the transaction?
  - How will all the millions of users agree to record this?
  - How to make sure people don't cheat the system?
- Solution: *Mining*

Mining, continued

- Alice's broadcast and others are picked up by some miners. They compile those transactions into *blocks*.

Mining, continued

- Alice's broadcast and others are picked up by some miners. They compile those transactions into *blocks*.
- They hash all the block's tansactions together in pairs, then hashing together those pairs, etc. in a tree structure, called a Merkle tree.

Mining, continued

- Alice's broadcast and others are picked up by some miners. They compile those transactions into *blocks*.
- They hash all the block's tansactions together in pairs, then hashing together those pairs, etc. in a tree structure, called a Merkle tree.
- Then each miner hashes in a "nonce," different for each miner

Mining, continued

- Alice's broadcast and others are picked up by some miners. They compile those transactions into *blocks*.
- They hash all the block's tansactions together in pairs, then hashing together those pairs, etc. in a tree structure, called a Merkle tree.
- Then each miner hashes in a "nonce," different for each miner
- After that, they hash in the previous block, thus tying this block to prior transactions.

Mining, continued

- Alice's broadcast and others are picked up by some miners. They compile those transactions into *blocks*.
- They hash all the block's tansactions together in pairs, then hashing together those pairs, etc. in a tree structure, called a Merkle tree.
- Then each miner hashes in a "nonce," different for each miner
- After that, they hash in the previous block, thus tying this block to prior transactions.
- All blocks ever since the beginning of bitcoin are concatenated together into one long chain, called the *blockchain* (this is the public ledger).

Mining, continued

- Alice's broadcast and others are picked up by some miners. They compile those transactions into *blocks*.
- They hash all the block's tansactions together in pairs, then hashing together those pairs, etc. in a tree structure, called a Merkle tree.
- Then each miner hashes in a "nonce," different for each miner
- After that, they hash in the previous block, thus tying this block to prior transactions.
- All blocks ever since the beginning of bitcoin are concatenated together into one long chain, called the *blockchain* (this is the public ledger).
- Then they *mine* the current block.

Mining, continued

- Alice's broadcast and others are picked up by some miners. They compile those transactions into *blocks*.
- They hash all the block's tansactions together in pairs, then hashing together those pairs, etc. in a tree structure, called a Merkle tree.
- Then each miner hashes in a "nonce," different for each miner
- After that, they hash in the previous block, thus tying this block to prior transactions.
- All blocks ever since the beginning of bitcoin are concatenated together into one long chain, called the *blockchain* (this is the public ledger).
- Then they *mine* the current block.
- This establishes the current block's validity (and with it, all the transactions it contains).

Mining, continued

- Alice's broadcast and others are picked up by some miners. They compile those transactions into *blocks*.
- They hash all the block's tansactions together in pairs, then hashing together those pairs, etc. in a tree structure, called a Merkle tree.
- Then each miner hashes in a "nonce," different for each miner
- After that, they hash in the previous block, thus tying this block to prior transactions.
- All blocks ever since the beginning of bitcoin are concatenated together into one long chain, called the *blockchain* (this is the public ledger).
- Then they *mine* the current block.
- This establishes the current block's validity (and with it, all the transactions it contains).
- How?

Proof of work protocol

- The result of all the hashing is a single value, v.

Proof of work protocol

- The result of all the hashing is a single value, v.
- Miners must find a p such that the digest of $h(v + p)$ is a 256-bit value starting with a certain number of zeroes, say 40 zeroes.

Proof of work protocol

- The result of all the hashing is a single value, v.
- Miners must find a p such that the digest of $h(v + p)$ is a 256-bit value starting with a certain number of zeroes, say 40 zeroes.
- The hash function h is SHA-256, which is thought to be secure, so a brute-force technique is the only way known to find such a p.

Proof of work protocol

- The result of all the hashing is a single value, v.
- Miners must find a p such that the digest of $h(v + p)$ is a 256-bit value starting with a certain number of zeroes, say 40 zeroes.
- The hash function h is SHA-256, which is thought to be secure, so a brute-force technique is the only way known to find such a p.
- They do this by starting with $p = 0$ and incrementing until they solve it.

Proof of work protocol

- The result of all the hashing is a single value, v.
- Miners must find a p such that the digest of $h(v + p)$ is a 256-bit value starting with a certain number of zeroes, say 40 zeroes.
- The hash function h is SHA-256, which is thought to be secure, so a brute-force technique is the only way known to find such a p.
- They do this by starting with $p = 0$ and incrementing until they solve it.
- To get a result staring with 40 zeroes takes a while, basically $2^{40} \approx 1$ trillion tries.

Proof of work protocol

- The result of all the hashing is a single value, v.
- Miners must find a p such that the digest of $h(v + p)$ is a 256-bit value starting with a certain number of zeroes, say 40 zeroes.
- The hash function h is SHA-256, which is thought to be secure, so a brute-force technique is the only way known to find such a p.
- They do this by starting with $p = 0$ and incrementing until they solve it.
- To get a result staring with 40 zeroes takes a while, basically $2^{40} \approx 1$ trillion tries.
- Many miners are working on this simultaneously, so eventually one will solve it.

Proof of work protocol

- The result of all the hashing is a single value, v.
- Miners must find a p such that the digest of $h(v + p)$ is a 256-bit value starting with a certain number of zeroes, say 40 zeroes.
- The hash function h is SHA-256, which is thought to be secure, so a brute-force technique is the only way known to find such a p.
- They do this by starting with $p = 0$ and incrementing until they solve it.
- To get a result staring with 40 zeroes takes a while, basically $2^{40} \approx 1$ trillion tries.
- Many miners are working on this simultaneously, so eventually one will solve it.
- The value 40 is specially chosen so that it takes on average 10 minutes for some node in the network to solve.

Proof of work protocol

- The result of all the hashing is a single value, v.
- Miners must find a p such that the digest of $h(v + p)$ is a 256-bit value starting with a certain number of zeroes, say 40 zeroes.
- The hash function h is SHA-256, which is thought to be secure, so a brute-force technique is the only way known to find such a p.
- They do this by starting with $p = 0$ and incrementing until they solve it.
- To get a result staring with 40 zeroes takes a while, basically $2^{40} \approx 1$ trillion tries.
- Many miners are working on this simultaneously, so eventually one will solve it.
- The value 40 is specially chosen so that it takes on average 10 minutes for some node in the network to solve.
- It is adjusted every couple of weeks to maintain this average.

Mining simulation

```
 >>> f(1)
40 0b918943df0962bc7a1824c0555a389347b4febdc7cf9d1254406d80ce44e3f9
>>> f(2)
287 00328ce57bbc14b33bd6695bc8eb32cdf2fb5f3a7d89ec14a42825e15d39df60
>>> f(3)
887 000f21ac06aceb9cdd0575e82d0d85fc39bed0a7a1d71970ba1641666a44f530
>>> f(4)
88485 0000a456e7b5a5eb059e721fb431436883143101275c4077f83fe70298f5623d
>>> f(5)
596139 00000691457f4f0ce13e187b9ab4fda6d42c8647752909b8f71f9dbd8f6bd4ab
>>> f(6)
665783 0000000399c6aea5ad0c709a9bc331a3ed6494702bd1d129d8c817a0257a1462
>>> f(7)
665783 0000000399c6aea5ad0c709a9bc331a3ed6494702bd1d129d8c817a0257a1462
>>> f(8)
426479725
00000000690ed426ccf17803ebe2bd0884bcd58a1bb5e7477ead3645f356e7a9
```

More on mining

- Each miner has a slightly different v, so all miners are solving different puzzles. Eventually one of them solves it.

More on mining

- Each miner has a slightly different v, so all miners are solving different puzzles. Eventually one of them solves it.
- Once a miner solves the challenge, they broadcast it to rest of network.

More on mining

- Each miner has a slightly different v, so all miners are solving different puzzles. Eventually one of them solves it.
- Once a miner solves the challenge, they broadcast it to rest of network.
- The other miners stop working on whatever they are doing.

More on mining

- Each miner has a slightly different v, so all miners are solving different puzzles. Eventually one of them solves it.
- Once a miner solves the challenge, they broadcast it to rest of network.
- The other miners stop working on whatever they are doing.
- They all start on a new block, building on the just-mined block.

More on mining

- Each miner has a slightly different v, so all miners are solving different puzzles. Eventually one of them solves it.
- Once a miner solves the challenge, they broadcast it to rest of network.
- The other miners stop working on whatever they are doing.
- They all start on a new block, building on the just-mined block.
- **It is this massive amount of work that backs all those transactions.**

More on mining

- Each miner has a slightly different v, so all miners are solving different puzzles. Eventually one of them solves it.
- Once a miner solves the challenge, they broadcast it to rest of network.
- The other miners stop working on whatever they are doing.
- They all start on a new block, building on the just-mined block.
- **It is this massive amount of work that backs all those transactions.**
- If someone wants to add in a fraudulent transaction, they have to mine faster than the whole rest of network.

More on mining

- Each miner has a slightly different v, so all miners are solving different puzzles. Eventually one of them solves it.
- Once a miner solves the challenge, they broadcast it to rest of network.
- The other miners stop working on whatever they are doing.
- They all start on a new block, building on the just-mined block.
- **It is this massive amount of work that backs all those transactions.**
- If someone wants to add in a fraudulent transaction, they have to mine faster than the whole rest of network.
- Only feasible if you have roughly as much computing power as the rest of the network combined.

Why mine?

- What benefit is there to miners?

Why mine?

- What benefit is there to miners?
- The miner who successfully solves the challenge gets the transaction fees for all the transactions that make up the block.

Why mine?

- What benefit is there to miners?
- The miner who successfully solves the challenge gets the transaction fees for all the transactions that make up the block.
- They also get a reward, currently 25 bitcoins.

Why mine?

- What benefit is there to miners?
- The miner who successfully solves the challenge gets the transaction fees for all the transactions that make up the block.
- They also get a reward, currently 25 bitcoins.
- Rewards are enough that it's more profitable to put your resources toward mining than towards trying to create fraudulent transactions.

The bitcoin money supply

- About 14 million bitcoins have been created to date.

The bitcoin money supply

- About 14 million bitcoins have been created to date.
- There will never be more than 21 million.

The bitcoin money supply

- About 14 million bitcoins have been created to date.
- There will never be more than 21 million.
- Mining reward was initially 50.

The bitcoin money supply

- About 14 million bitcoins have been created to date.
- There will never be more than 21 million.
- Mining reward was initially 50.
- It is currently 25.

The bitcoin money supply

- About 14 million bitcoins have been created to date.
- There will never be more than 21 million.
- Mining reward was initially 50.
- It is currently 25.
- It halves after every 210,000 blocks (about 4 years since 10 minutes per block).

The bitcoin money supply

- About 14 million bitcoins have been created to date.
- There will never be more than 21 million.
- Mining reward was initially 50.
- It is currently 25.
- It halves after every 210,000 blocks (about 4 years since 10 minutes per block).
- Last one will be created around year 2140 (though by exponential decay most will be generated well before).
- Bitcoins can be broken down. Minimum unit is 1/100,000,000 of a bitcoin, a *satoshi*.

Mining

- Bitcoin mining has become so popular that it's not feasible for individual to mine without special hardware and/or joining a mining pool.
- Total processing consumed by mining exceeds that of the world's top 500+ supercomputers combined.
- Some people are worried about all the power consumed by bitcoin mining.

From https://www.coinbase.com/network

Bitcoin Node

Our connection to the bitcoin network

349,950 Blocks

Last block created 3 minutes ago

67,008,441 Transactions

Last transaction created less than a minute ago

1,608 unconfirmed transactions

35 Peers Connected

8360 available

From https://www.coinbase.com/network

Network › Transactions

Transaction Hash	Pool	Net Amount	Created
df321b40f89ffa9a1b818dc48229e1dc3caee5c2d52e08d5855e1d3bbedba4fb	Transaction Pool	0.41453829	less than a minute ago
2433246261342d91641ee7a8235f704b558e75a9ee0f7faf256f13d594860879	Transaction Pool	1.90085498	less than a minute ago
f3a5e73b5cb2717dbd6098d34dcfaff650fb2e92a566c430ea6fcbc28b0b32eb	Transaction Pool	0.00100000	less than a minute ago
7847e3b08456d28a2587167f686a4bd08407b257661abe5efa9b165a9e97e21e	Transaction Pool	0.49999000	less than a minute ago
b5a31252d83ead1ed3b2de2aaaa5e53ad979b04b741b8384195cdf3a61aa1bf6	Transaction Pool	0.14321780	less than a minute ago
8c4fa57b87e23d116bdcb0bd868211493ec14a6ba1d1baf113ecc42f4a3007d2	Transaction Pool	0.01926981	less than a minute ago
79613d9c3389625e35327ec112922fbcfd331fac9ecbc5dd272afed7d66d7605	Transaction Pool	0.61214489	less than a minute ago
7b8139e4a738ff5e64e0eb3d8462e1b6d0c9722796375b0413993e21a24ef3e6	Transaction Pool	5.01930000	less than a minute ago

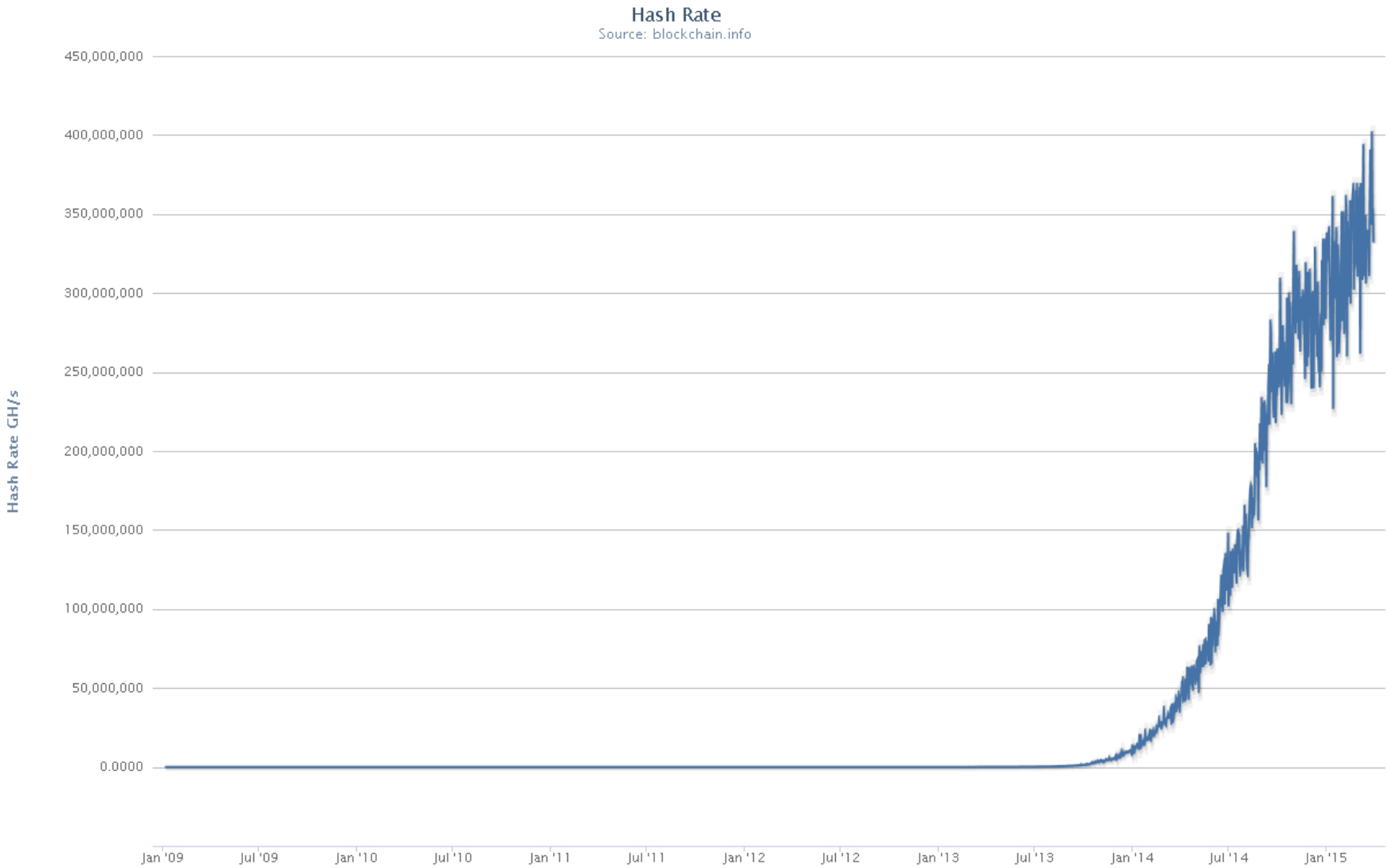
From https://www.coinbase.com/network

Network › Blocks

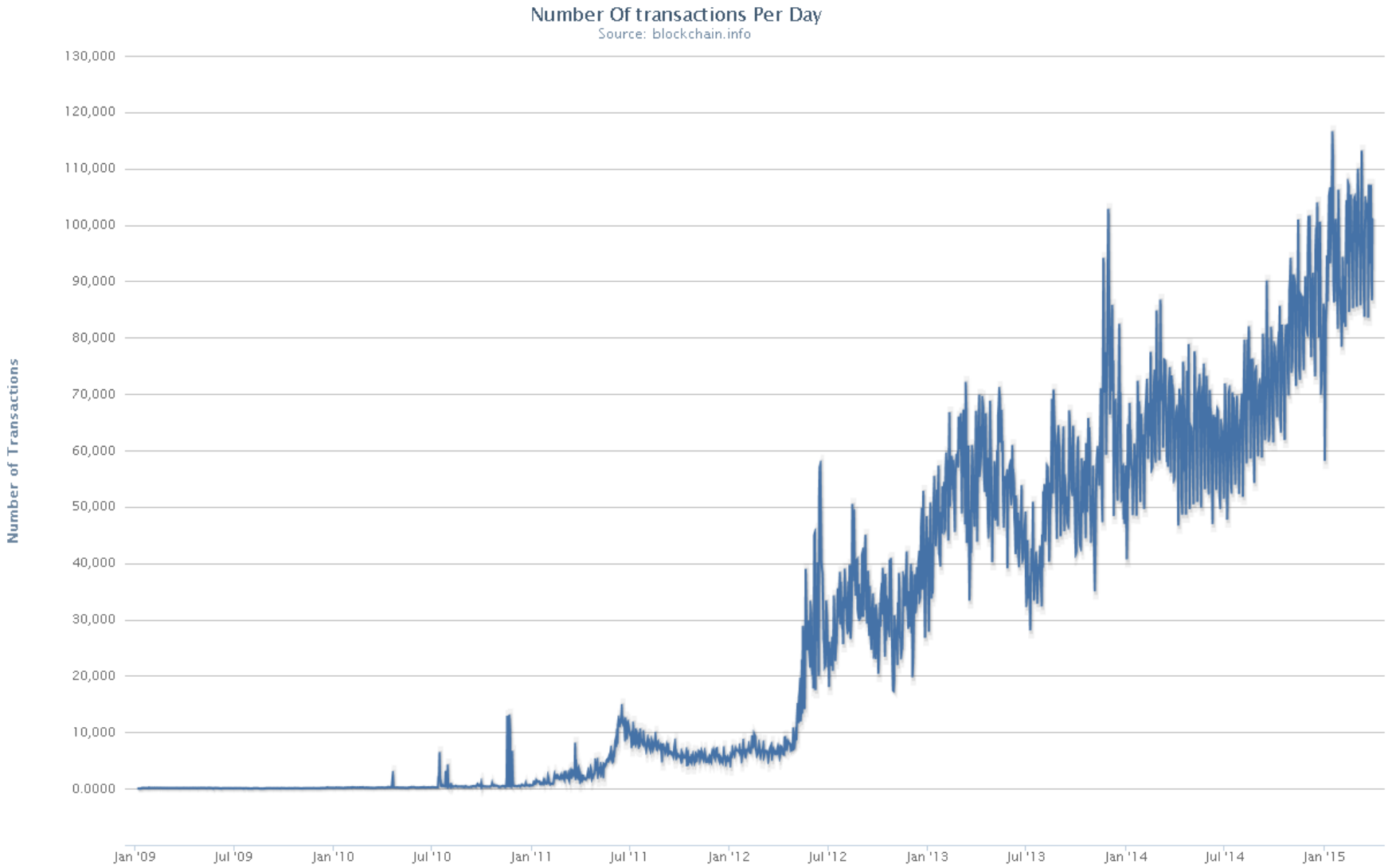
Block Hash	Branch	Height	Transactions	Timestamp
00000000000000000116fc83685867b561f6b0566999ff502cb5d296844847fc8	main	349951	356	2 minutes ago
000000000000000001055564896d474580b19b18bab2448a7978e6972b59b4111b	main	349950	1050	6 minutes ago
00000000000000000001a7f5b11d35183c7a0d9d9c5e4803d5705865ad8a0cd89d	main	349949	1723	15 minutes ago
000000000000000000001453922aff3e7ca334f070920f081ae9de129870b69fd47	main	349948	85	40 minutes ago
00000000000000000001d9c1e745d1c091d8012ab87942a8335583b7357d6d7be3	main	349947	1039	39 minutes ago
0000000000000000064f4270c6878fdf555c1c3cdf6223f8810fafed500616bf	main	349946	706	about 1 hour ago
00000000000000000006cbec774c2acf7c595bc3a3f3010124592a147f8a670ec7	main	349945	1410	about 1 hour ago

From https://blockchain.info/charts

From https://blockchain.info/charts



From https://blockchain.info/charts



Sources

- www.khanacademy.org/economics-finance-domain/core-finance/money-and-banking/bitcoin/
- chrispacia.wordpress.com/?s=Explained+Like+You%27re+Five&submit=Search
- bitcoin.org/bitcoin.pdf (Nakomoto's original whitepaper.)
- bitcoin.org/en/faq
- www.coinbase.com/network (Current info about blocks/transactions)
- blockchain.info/charts (Historical and recent graphs)