Blockchain: Magic, Mechanics and Methods

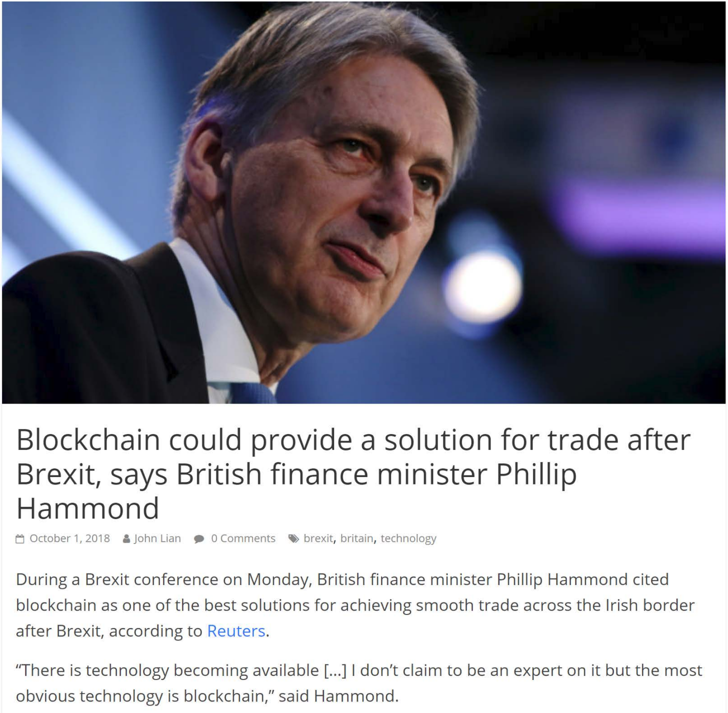
Stephen J. Mildenhall

November 2018

ST. JOHN'S UNIVERSITY

Tobin College of Business
School of Risk Management

Blockchain: Marketing and Magic



Blockchain could provide a solution for trade after Brexit, says British finance minister Phillip Hammond

October 1, 2018 John Lian 0 Comments brexit, britain, technology

During a Brexit conference on Monday, British finance minister Phillip Hammond cited blockchain as one of the best solutions for achieving smooth trade across the Irish border after Brexit, according to Reuters.

“There is technology becoming available [...] I don’t claim to be an expert on it but the most obvious technology is blockchain,” said Hammond.

Any sufficiently advanced technology is indistinguishable from magic.

Arthur C. Clarke

Define

Blockchains are **distributed** digital **ledgers** of **cryptographically signed transactions** that are grouped into **blocks**. Each block is **cryptographically linked** to the previous one after **validation** and undergoing a **consensus decision**, making it **tamper evident**. As new blocks are added, older blocks become more **difficult to modify**. New blocks are **replicated** across copies of the ledger within the network, and any **conflicts** are **resolved automatically** using established rules.

Blockchain Technology Overview, Yaga et al (2018), NIST

Define

Blockchains are **distributed** digital **ledgers** of **cryptographically signed transactions** that are grouped into **blocks.** Each block is **cryptographically linked** to the previous one after **validation** and undergoing a **consensus decision** decision, making it **tamper evident.** As new blocks are added, older blocks become more **difficult to modify.** New blocks are **replicated** across copies of the ledger within the network, and any **conflicts** are **resolved automatically** using established rules.

Blockchain Technology Overview, Yaga et al (2018), NIST

Describe

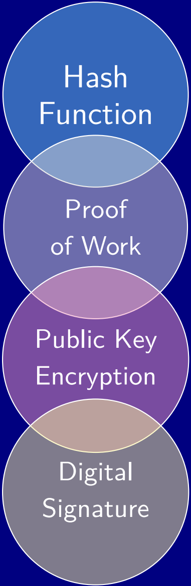
Components

- Distributed database
- Ledger
- Cryptographically...
- ...Signed transactions
- ...Linked (chained)
- Consensus Validation

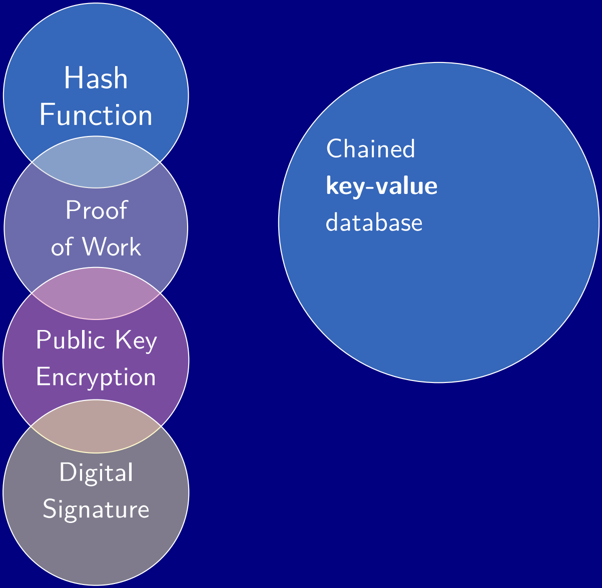
Characteristics

- No authority
- High availability
- Replicated, robust
- Tamper evident
- Difficult to modify
- Conflicts resolved

Dissect: Magical Ingredients & Recipe



Dissect: Magical Ingredients & Recipe



Hash Function

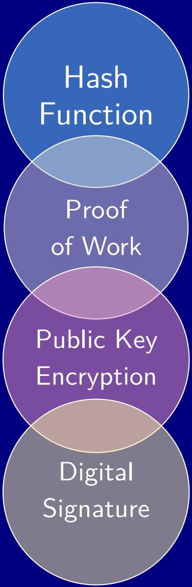
Proof of Work

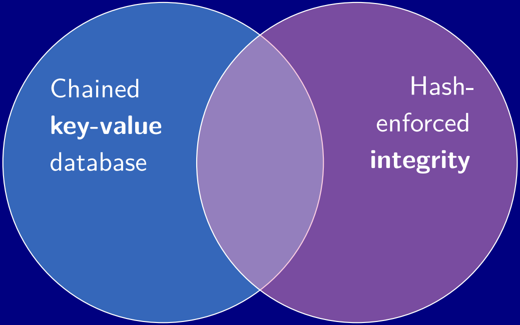
Public Key Encryption

Digital Signature

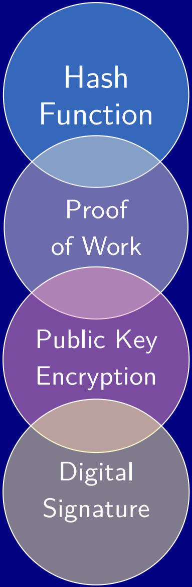
Chained **key-value** database

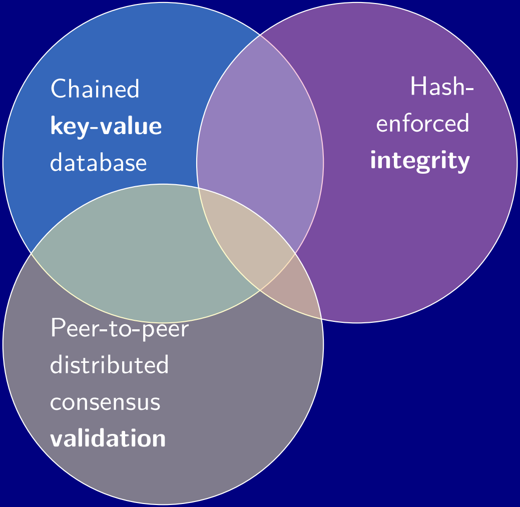
Dissect: Magical Ingredients & Recipe



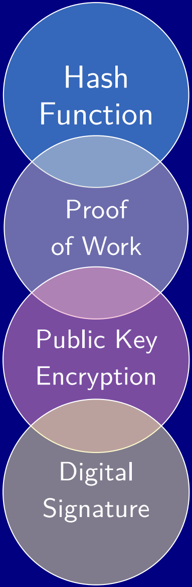


Dissect: Magical Ingredients & Recipe




Chained **key-value** database
Hash-enforced **integrity**
Peer-to-peer distributed consensus **validation**

Dissect: Magical Ingredients & Recipe

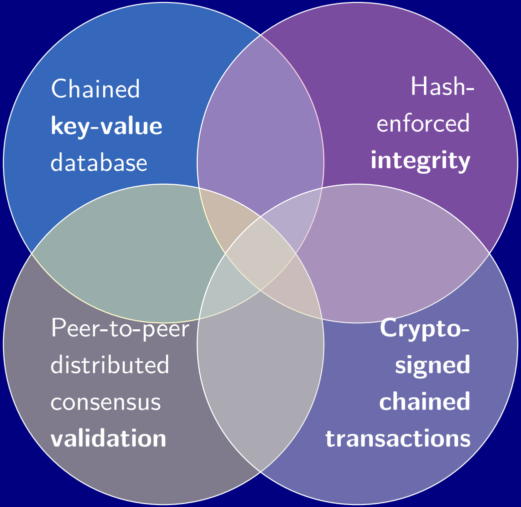


Hash Function

Proof of Work

Public Key Encryption

Digital Signature



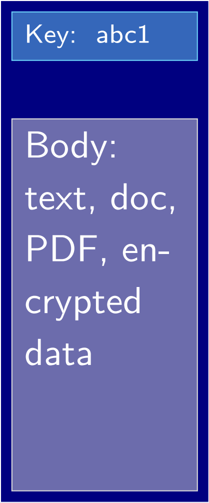
Chained **key-value** database

Hash-enforced **integrity**

Peer-to-peer distributed consensus **validation**

Crypto-signed chained transactions

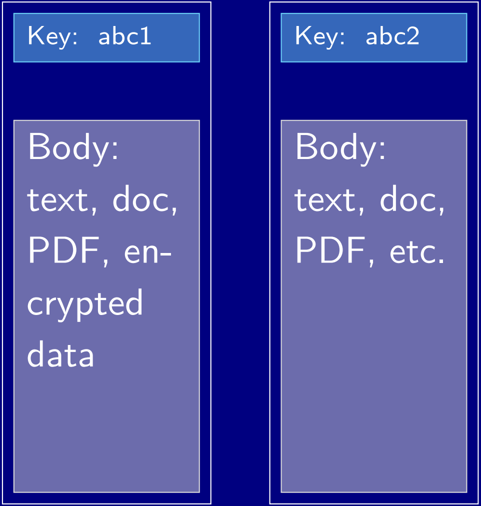
Ingredient 1: Chained Key-Value (Distributed) Database



Key: abc1

Body: text, doc, PDF, encrypted data

Ingredient 1: Chained Key-Value (Distributed) Database



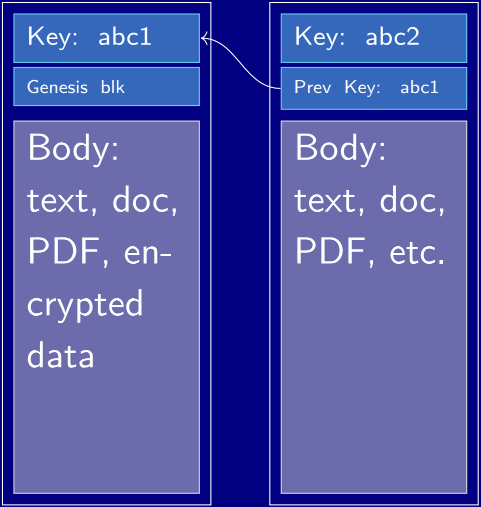
Key: abc1

Body: text, doc, PDF, encrypted data

Key: abc2

Body: text, doc, PDF, etc.

Ingredient 1: Chained Key-Value (Distributed) Database



Key: abc1

Genesis blk

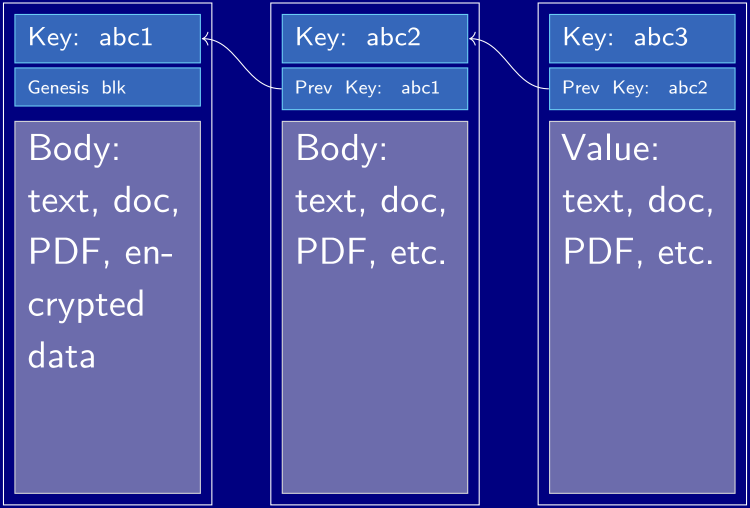
Body: text, doc, PDF, encrypted data

Key: abc2

Prev Key: abc1

Body: text, doc, PDF, etc.

Ingredient 1: Chained Key-Value (Distributed) Database



Key: abc1

Genesis blk

Body:
text, doc, PDF, encrypted data

Key: abc2

Prev Key: abc1

Body:
text, doc, PDF, etc.

Key: abc3

Prev Key: abc2

Value:
text, doc, PDF, etc.

Ingredient: Hash Functions

A **hash** H maps data of arbitrary size to a fixed size such that

- $H(x)$ is an easy to compute, deterministic function
- If $x \neq y$ then $H(x) \neq H(y)$ with high probability
- $H(x)$ appears random over its range as x varies
- IT hash function: first five letters of last name + first letter first name
- J. Smith problem
- Phone, zip, social, . . .

Ingredient: Hash Functions

A **hash** H maps data of arbitrary size to a fixed size such that

- $H(x)$ is an easy to compute, deterministic function
- If $x \neq y$ then $H(x) \neq H(y)$ with high probability
- $H(x)$ appears random over its range as x varies
- IT hash function: first five letters of last name + first letter first name
- J. Smith problem
- Phone, zip, social, . . .

Cryptographic Hash Function

- Given y it is **very hard** to find x with $H(x) = y$
- **Fuggedaboutit** hard

SHA256 Cryptographic Hash Function

```
import hashlib

hashlib.sha256(b'The quick brown fox jumps over the lazy dog').hexdigest()
>>> 'd7a8fbb307d7809469ca9abcb0082e4f8d5651e46d3cdb762d02d0bf37c9e592'

hashlib.sha256(b'The quick brown fox jumps over the lazy dog.').hexdigest()
>>> 'ef537f25c895bfa782526529a9b63d97aa631564d5d789c2b765448c8635fb6c'
```

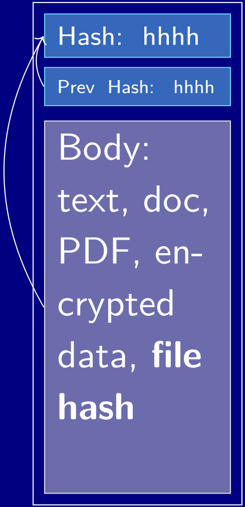
- Output = **very** large integer, between 0 and $2^{256} \approx 10^{77}$
- Specify input and output formats **very carefully**
- Probability of J. Smith collision: not even a Dumb and Dumber chance

The Birthday Problem and Hash Collisions

- Birthday problem: 23 people for 50/50 chance of same birthday
- Number of documents before p probability of collision given a hash space size of N is $\approx \sqrt{2Np}$ for small p[1]
- For SHA256, $N = 2^{256} = 10^{77}$ is very large
- A 10^{-3} collision probability requires about 1.5×10^{37} documents, enough for
    - Every person on earth to...
    - Compute 1 billion hashes per second...
    - For five times the age of the universe

[1] E.g. for birthday problem $p = 1/2$, $N = 365$ and $\sqrt{2Np} = 19$. Approximation relies on $p \approx -\log(1-p)$, only true for smaller p. Using $(-2N\log(1-p))^{1/2} = 22.49$ is very close to correct answer, 23.

Ingredient 2: Hash-Enforced **Integrity**

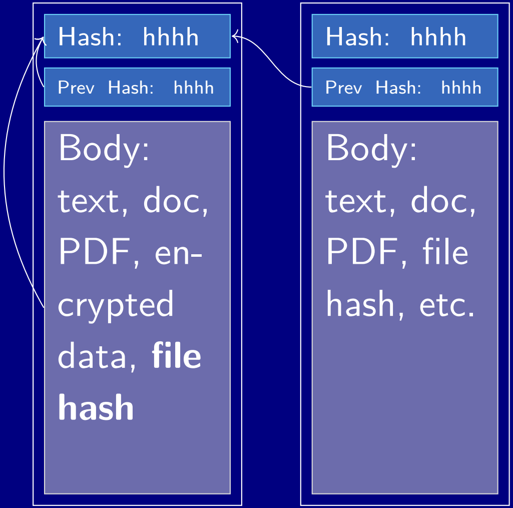


Hash: hhhh

Prev Hash: hhhh

Body: text, doc, PDF, encrypted data, **file hash**

Ingredient 2: Hash-Enforced **Integrity**



Hash: hhhh

Prev Hash: hhhh

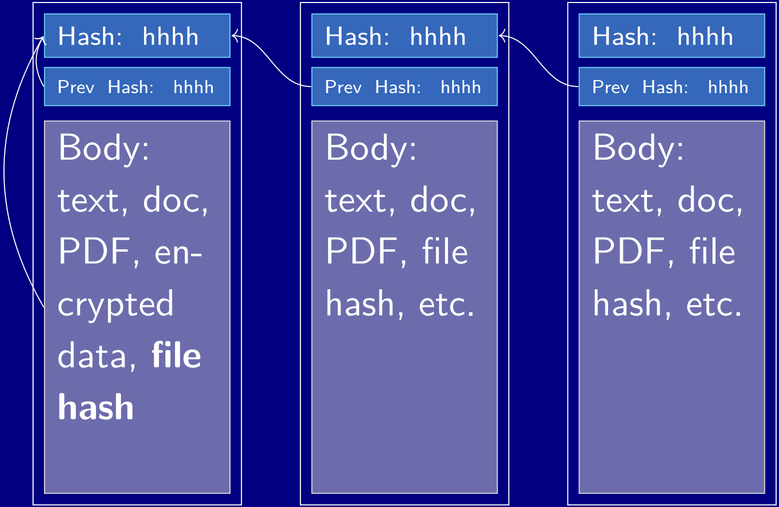
Body: text, doc, PDF, encrypted data, **file hash**

Hash: hhhh

Prev Hash: hhhh

Body: text, doc, PDF, file hash, etc.

Ingredient 2: Hash-Enforced **Integrity**



Hash: hhhh

Prev Hash: hhhh

Body: text, doc, PDF, encrypted data, **file hash**

Hash: hhhh

Prev Hash: hhhh

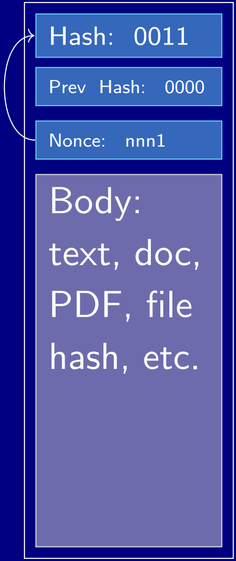
Body: text, doc, PDF, file hash, etc.

Hash: hhhh

Prev Hash: hhhh

Body: text, doc, PDF, file hash, etc.

Ingredient 3: Distributed Validation and **Proof-Of-Work**



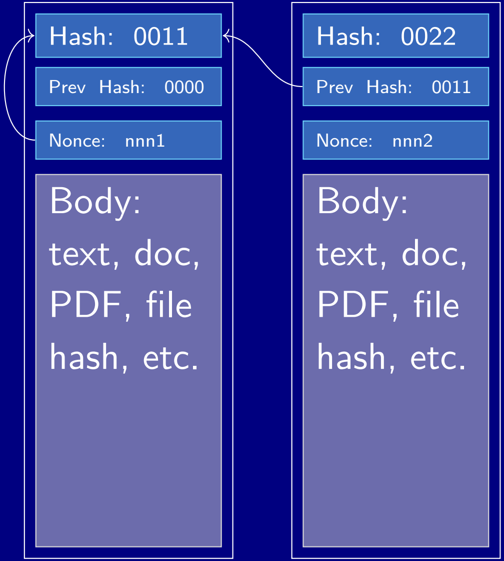
Hash: 0011

Prev Hash: 0000

Nonce: nnn1

Body:
text, doc,
PDF, file
hash, etc.

Ingredient 3: Distributed Validation and **Proof-Of-Work**



Hash: 0011

Prev Hash: 0000

Nonce: nnn1

Body: text, doc, PDF, file hash, etc.

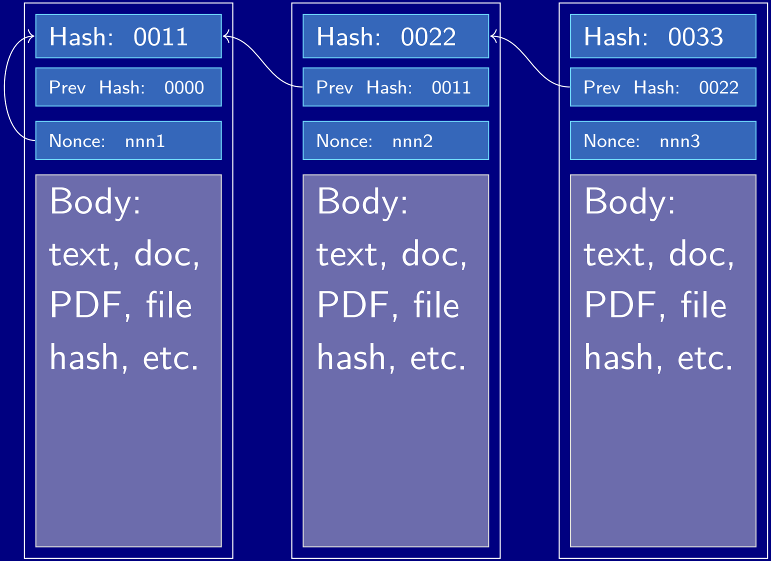
Hash: 0022

Prev Hash: 0011

Nonce: nnn2

Body: text, doc, PDF, file hash, etc.

Ingredient 3: Distributed Validation and **Proof-Of-Work**



Hash: 0011

Prev Hash: 0000

Nonce: nnn1

Body: text, doc, PDF, file hash, etc.

Hash: 0022

Prev Hash: 0011

Nonce: nnn2

Body: text, doc, PDF, file hash, etc.

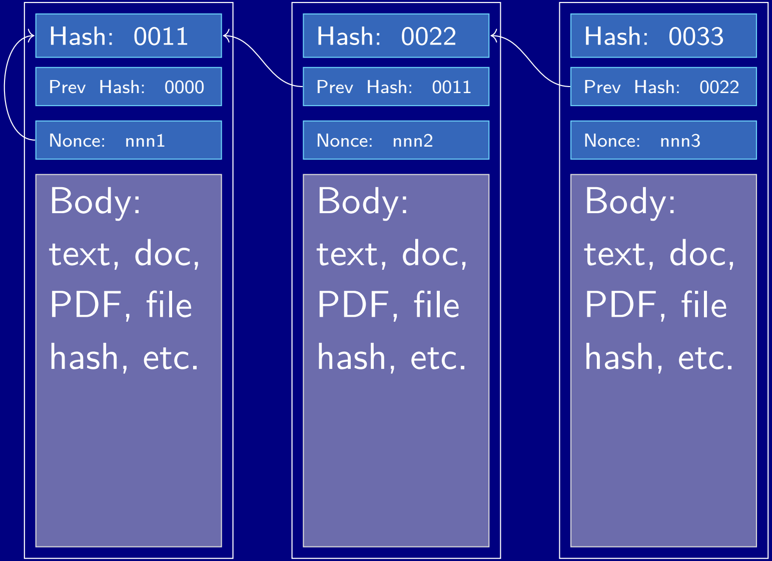
Hash: 0033

Prev Hash: 0022

Nonce: nnn3

Body: text, doc, PDF, file hash, etc.

Ingredient 3: Distributed Validation and **Proof-Of-Work**



Hash: 0011

Prev Hash: 0000

Nonce: nnn1

Body:
text, doc,
PDF, file
hash, etc.

Hash: 0022

Prev Hash: 0011

Nonce: nnn2

Body:
text, doc,
PDF, file
hash, etc.

Hash: 0033

Prev Hash: 0022

Nonce: nnn3

Body:
text, doc,
PDF, file
hash, etc.

Proof of Work and Bitcoin Mining = Compute Hashes

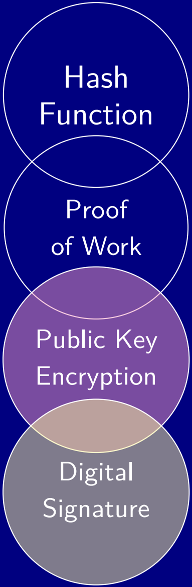
```python
import hashlib

running_min = 2**256
ans = []
base = b'The quick brown fox jumps over the lazy dog'

for nonce in range(2000000000):  # 2 billion
    h = hashlib.sha256(nonce.to_bytes(4, byteorder='big') + base).hexdigest()
    n = int(h, 16)
    if n <  running_min:
        running_min = n
        ans.append([nonce, n])
        print(f'{nonce:12,d}   {n:077}')
```

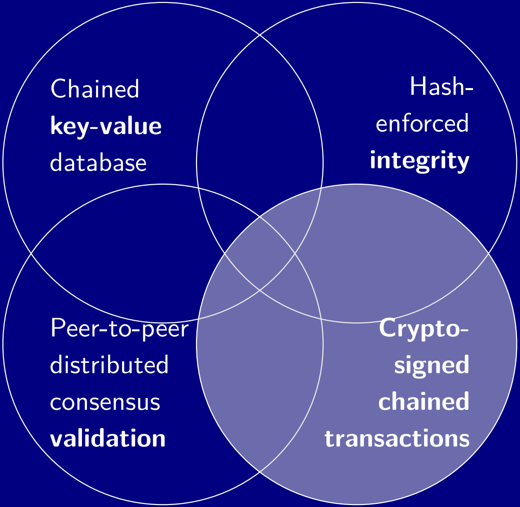
Proof of Work and Bitcoin Mining = Compute Hashes

```
Nonce          Hash
            0    29115579230639891023898657467946481563928575965694753738500728003067276450760
            3    21833633896494697913657452817095065461049276120751755746016193921330837964982
            8    17391853960576662285627567225372501697536440120814058733709287576654299269058
            9   00 491741673371171570027367996335736784622791320015893772572199978008540614786
          817   00 207113148484537618144604663416437589440289273319027116671254033065643419132
          827   000 35029650895291714754047120679492927968250654901817817434081241936987361735
        3,292   000 30590294895123458493702891527069975442971551875566805022772671084264919745
        6,362   000 23157006908555232018903879877754051315219896322661305099606253143774488785
        7,634   000 11843095073522994422561274720857316931066719486382550615573171404879921966
       22,034   0000 6045160764465103256154815045992679930360222615550766779824452388654984639
       32,737   0000 3218718010716516807246023638919032202673987969434384430166215105132280583
       43,078   0000 3066940367111277087798394765784480513227788830972580117541505418890948712
       50,740   00000 344804005194498392473362848134761831134304453202875173759130216105619080
      260,109   00000 149043122808237032345561872905133216060467384369910593113997965062602336
      610,827   000000 25441204939268765420155917698735840343496809686969451042687651132777655
    3,553,698   000000 12372585984995238023081534031026808791454761919139475665549030259593011
   16,603,005   0000000 4682308792444739613119316155033986067282587356863979013510780284611482
   45,767,445   0000000 4295135810439807939037487563409966578108755229939605598485594694500274
   56,389,936   0000000 1219890553970511010693160459086914039690075265862677724048817741406404
  186,599,009   00000000 741733398915175814111679160159562329641666849535152212310255158283708
  187,060,155   00000000 129027976973068678554136418237268320708790839626316760173444080235551
  209,437,773   000000000 46418792192972977622708878642780226280538977482131916077098153688658
  554,751,705   000000000 38492057003517052607600918969310106371482316138230835578404460555913
1,724,412,865   000000000 20951411954830677538112338658105096359813168232452740277675602777590
```

- Current network hash rate 4 × 10^{19} hashes per second
- Electricity consumption = Austria
- Block hash: 0x 0000 0000 0000 0000 0051 a841 86ab c5df . . . .

Dissect: Cryptographic Ingredients




Chained **key-value** database
Hash-enforced **integrity**
Peer-to-peer distributed consensus **validation**
Crypto-signed chained transactions

Discrete Logarithm Problem

- **Discrete logarithm problem** says

  $$\text{given } g^a \equiv n \pmod{p} \text{ can't find } a$$

  is a one-way function
- mod p means remainder after dividing by prime p

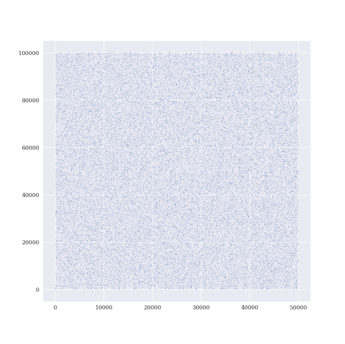
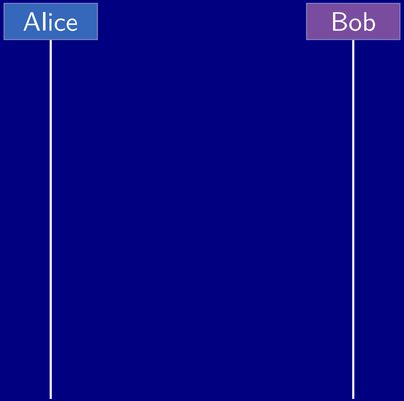


Figure 1: Powers of 3 modulo 100043; 100042 = 2 x 50021 is twice a prime.

Creating a Shared Secret

Public parameters g and p

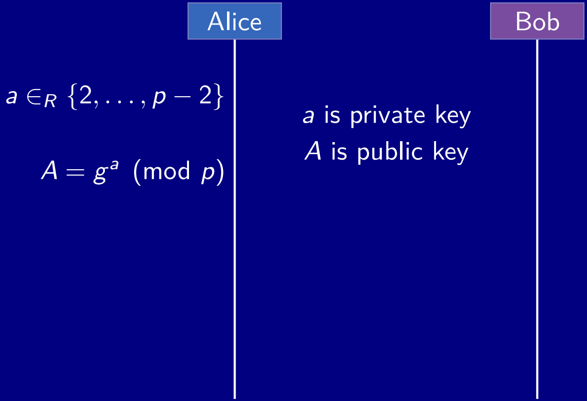


Alice

Bob

Creating a Shared Secret

Public parameters g and p



Alice

Bob

$a \in_R \{2, \ldots, p-2\}$

$A = g^a \pmod{p}$

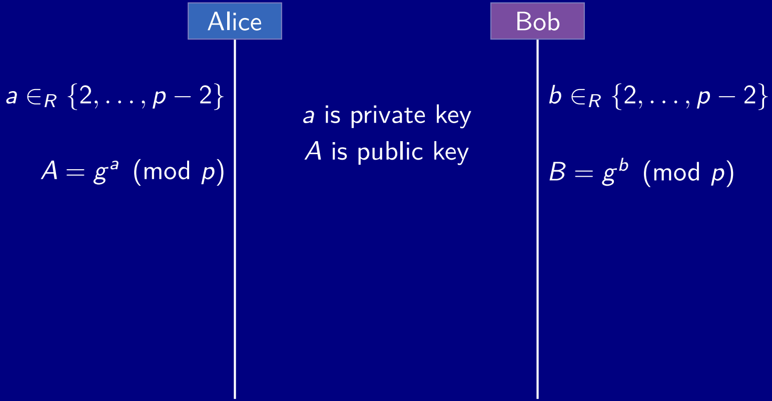
a is private key

A is public key

Public/private pair (A, a) are cryptographically linked but a is hidden

Creating a Shared Secret

Public parameters g and p



Alice

Bob

$a \in_R \{2, \ldots, p-2\}$

$A = g^a \pmod{p}$

a is private key

A is public key

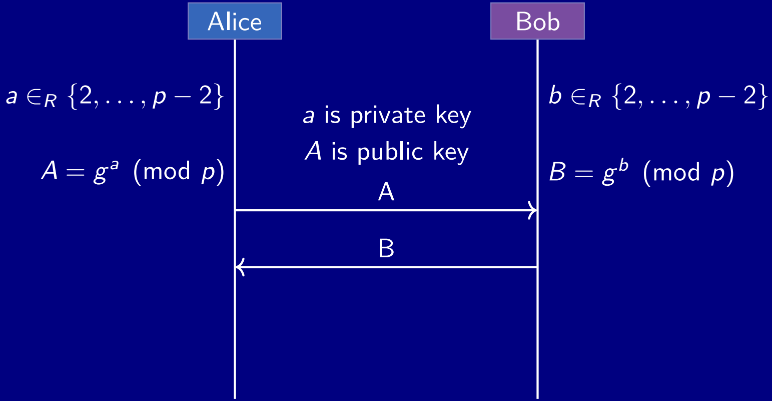
$b \in_R \{2, \ldots, p-2\}$

$B = g^b \pmod{p}$

Public/private pair (A, a) are cryptographically linked but a is hidden

Creating a Shared Secret

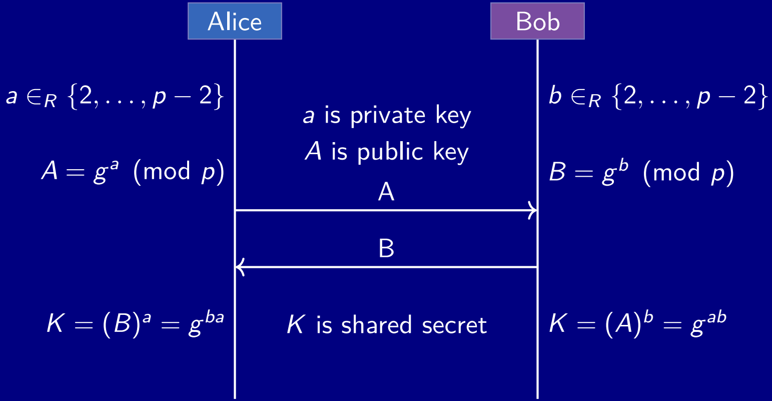
Public parameters g and p



Public/private pair (A, a) are cryptographically linked but a is hidden

Creating a Shared Secret

Public parameters g and p



$a \in_R \{2, \ldots, p-2\}$

$b \in_R \{2, \ldots, p-2\}$

a is private key

A is public key

$A = g^a \pmod{p}$

$B = g^b \pmod{p}$

$K = (B)^a = g^{ba}$

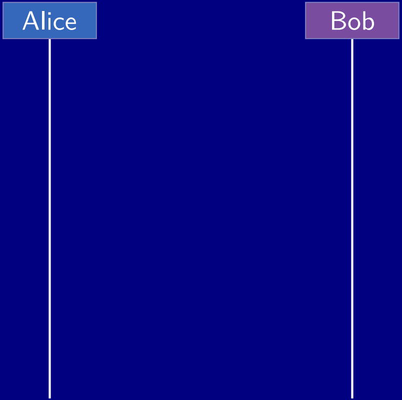
K is shared secret

$K = (A)^b = g^{ab}$

Public/private pair (A, a) are cryptographically linked but a is hidden

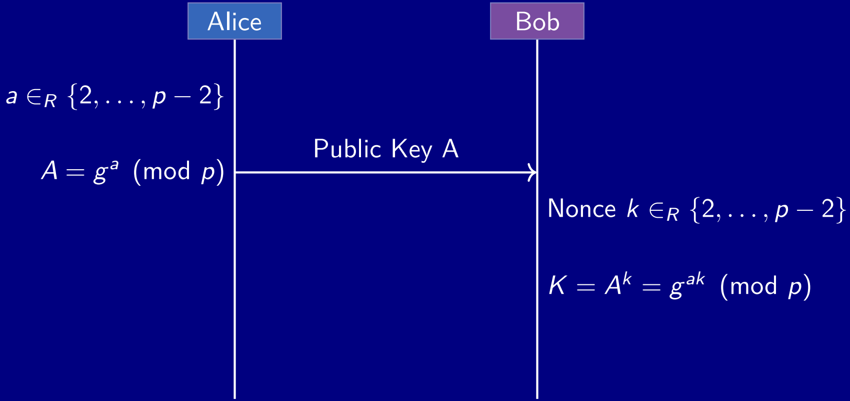
ElGamel Public Key Encryption

Public parameters g and p
Send message m from Bob to Alice



ElGamel Public Key Encryption

Public parameters g and p
Send message m from Bob to Alice



Alice

Bob

$a \in_R \{2, \ldots, p-2\}$

$A = g^a \pmod{p}$

Public Key A

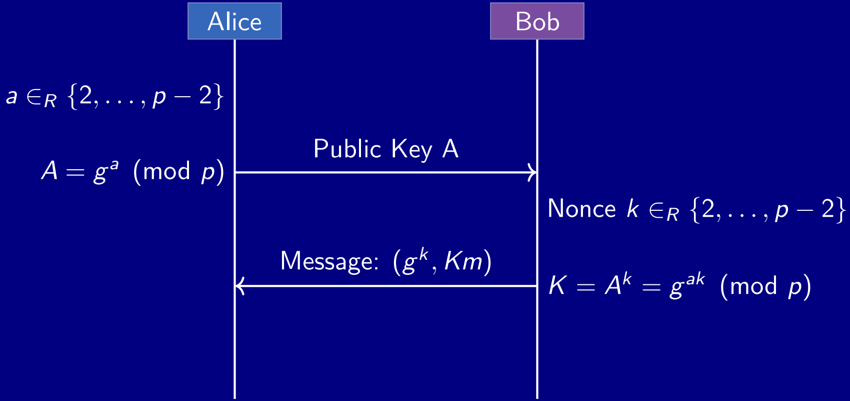
Nonce $k \in_R \{2, \ldots, p-2\}$

$K = A^k = g^{ak} \pmod{p}$

ElGamel Public Key Encryption

Public parameters g and p

Send message m from Bob to Alice



Alice

Bob

$a \in_R \{2, \ldots, p-2\}$

$A = g^a \pmod{p}$

Public Key A

Nonce $k \in_R \{2, \ldots, p-2\}$

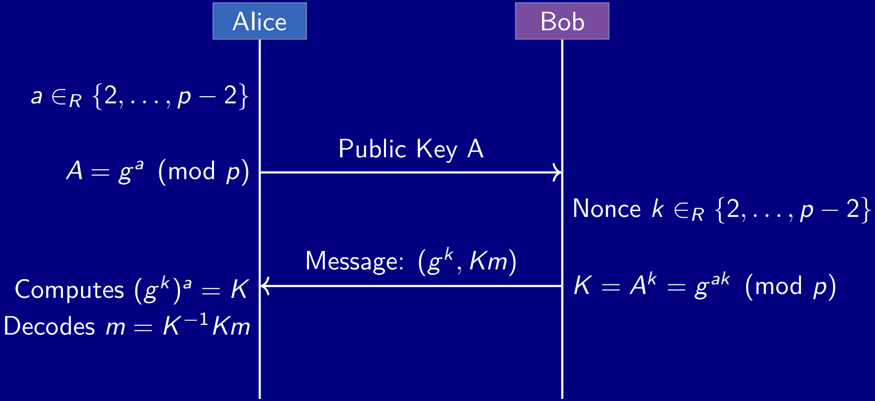
Message: (g^k, Km)

$K = A^k = g^{ak} \pmod{p}$

ElGamel Public Key Encryption

Public parameters g and p

Send message m from Bob to Alice



Alice

Bob

$a \in_R \{2, \ldots, p-2\}$

$A = g^a \pmod{p}$

Public Key A

Nonce $k \in_R \{2, \ldots, p-2\}$

Message: (g^k, Km)

$K = A^k = g^{ak} \pmod{p}$

Computes $(g^k)^a = K$

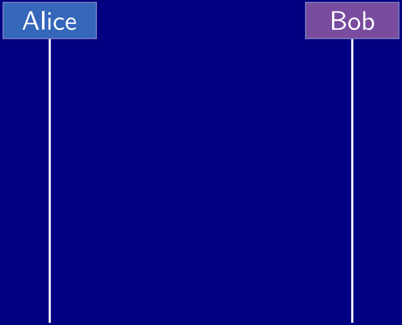
Decodes $m = K^{-1}Km$

g^k conveys information about k but shields its value; K hides message m

Digital Signature

Alice to sign message m, Bob to verify

$g, p, A = g^a, m$ all public, a is secret



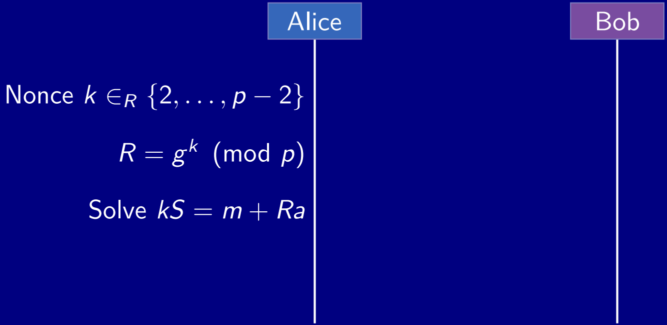
Alice

Bob

Digital Signature

Alice to sign message m, Bob to verify

$g, p, A = g^a, m$ all public, a is secret



Alice

Bob

Nonce $k \in_R \{2, \ldots, p-2\}$

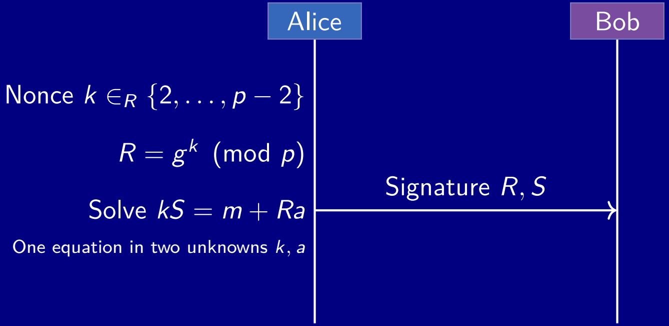
$R = g^k \pmod{p}$

Solve $kS = m + Ra$

Digital Signature

Alice to sign message m, Bob to verify

$g, p, A = g^a, m$ all public, a is secret



Alice

Bob

Nonce $k \in_R \{2, \dots, p-2\}$

$R = g^k \pmod{p}$

Solve $kS = m + Ra$

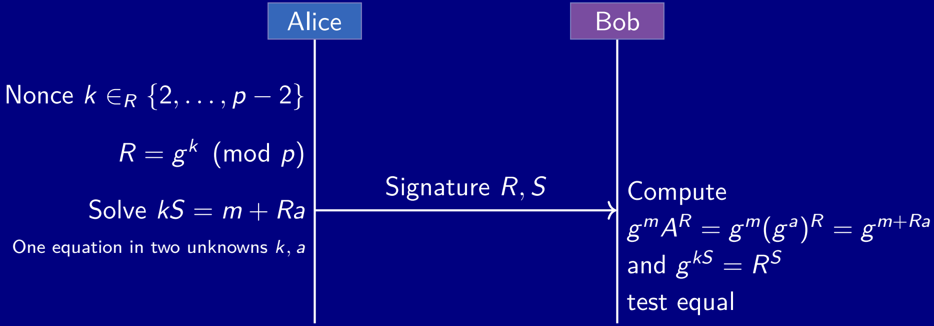
One equation in two unknowns k, a

Signature R, S

Digital Signature

Alice to sign message m, Bob to verify

$g, p, A = g^a, m$ all public, a is secret



Alice

Bob

Nonce $k \in_R \{2, \ldots, p-2\}$

$R = g^k \pmod{p}$

Solve $kS = m + Ra$

One equation in two unknowns k, a

Signature R, S

Compute

$g^m A^R = g^m (g^a)^R = g^{m+Ra}$

and $g^{kS} = R^S$

test equal

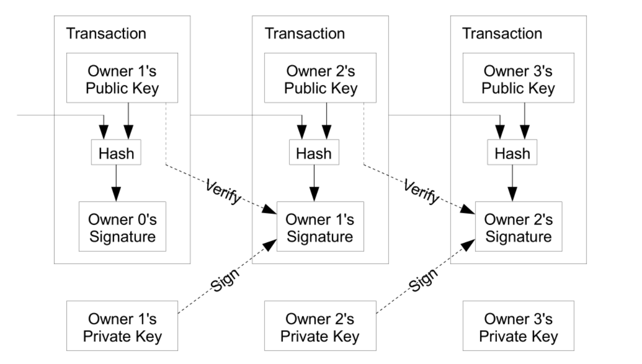
If Alice does not know a she can't find R, S to solve $R^S \; = \; g^m A^R$

Powerful Properties of Digital Signature

- Signer **authentication**: verifier assured that signature has been created only by sender who possess the corresponding secret private key
- Message **integrity**: if message modified, signature fails; signature tamper evident
- **Non-repudiation**: existence of signature proves it came from sender; sender cannot repudiate signing in future
- Wet ink signatures can be forged; document can be altered; signature can be denied

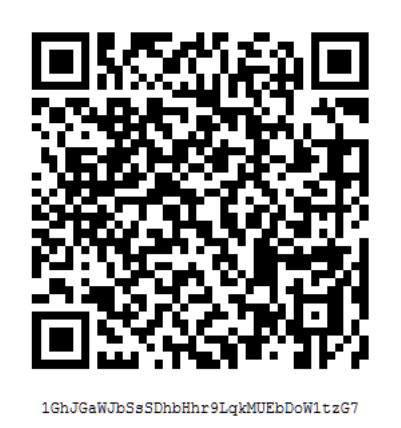
Ingredient 4: Double-spend mechanism

- Bitcoin ledger tracks coin ownership
- Owners can endorse to new owners in cryptographically secure manner
- Public pseudonymous chain of ownership



Bitcoin: A Peer to Peer Electronic Cash System, Satoshi Nakamoto (2008) https://bitcoin.org/bitcoin.pdf

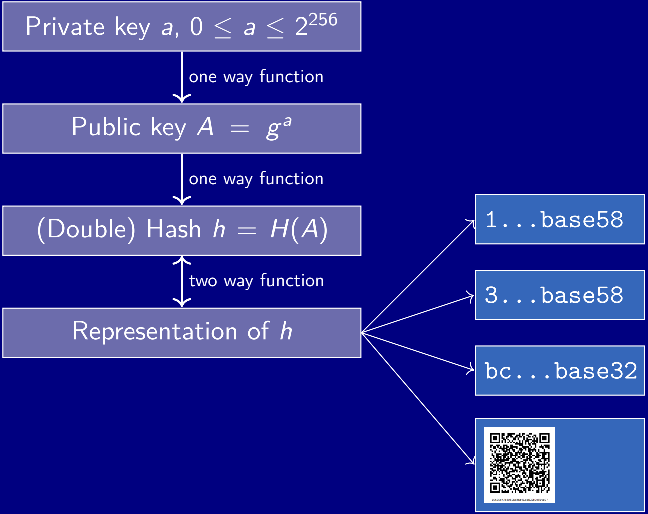
What is a Bitcoin Public Address?



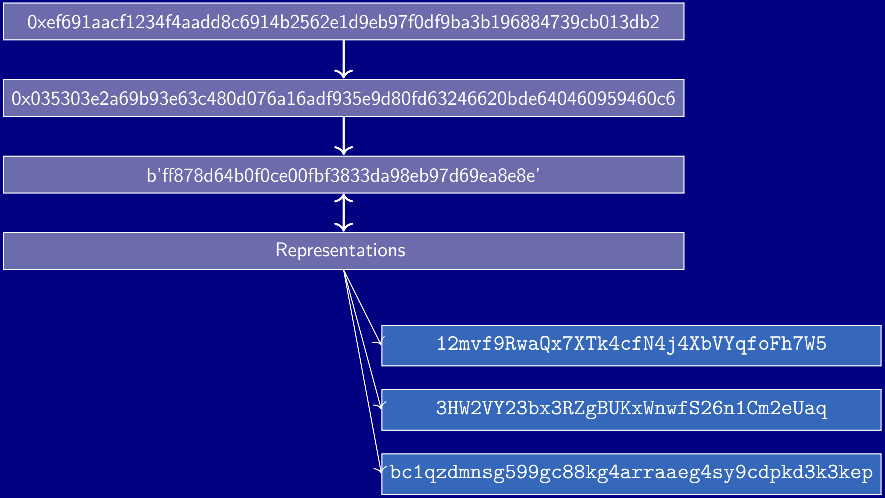
1GhJGaWJbSsSDhbHhr9LqkMUEbDoW1tzG7

Figure 2: Donations gratefully received.

What is a Bitcoin Public Address?



What is a Bitcoin Public Address?



0xef691aacf1234f4aadd8c6914b2562e1d9eb97f0df9ba3b196884739cb013db2

0x035303e2a69b93e63c480d076a16adf935e9d80fd63246620bde640460959460c6

b'ff878d64b0f0ce00fbf3833da98eb97d69ea8e8e'

Representations

12mvf9RwaQx7XTk4cfN4j4XbVYqfoFh7W5

3HW2VY23bx3RZgBUKxWnwfS26n1Cm2eUaq

bc1qzdmnsg599gc88kg4arraaeg4sy9cdpkd3k3kep

If You Know What You Are Doing…

Load into Bitcoin Core Client and get addresses via WIF compressed representation of private key

```
importprivkey
  L5F6PZo9h2RJnGGvztwWEUnwYH1eWhpv63Z5qQEZgqxcy364nBCQj
  yourName

getaddressesbyaccount yourName

[
  "12mvf9RwaQx7XTk4cfN4j4XbVYqfoFh7W5",
  "3HW2VY23bx3RZgBUKxWnwfS26n1Cm2eUaq",
  "bc1qzdmnsg599gc88kg4arraaeg4sy9cdpkd3k3kep"
]
```

Discovery: Solution in Search of a Problem

Using ingredients...

- Hash functions
- Public/private keys
- Digital signatures
- Chained blocks
- Chained transactions
- A clever **incentive reinforcing** recipe

We have created a...

- Distributed...
- Available...
- Public/unsuppressable...
- Immutable database
- No central authority
- Trust between strangers
- Digital scarcity

Discovery: Solution in Search of a Problem

Using ingredients...

- Hash functions
- Public/private keys
- Digital signatures
- Chained blocks
- Chained transactions
- A clever **incentive reinforcing** recipe

We have created a...

- Distributed...
- Available...
- Public/unsuppressable...
- Immutable database
- No central authority
- Trust between strangers
- Digital scarcity

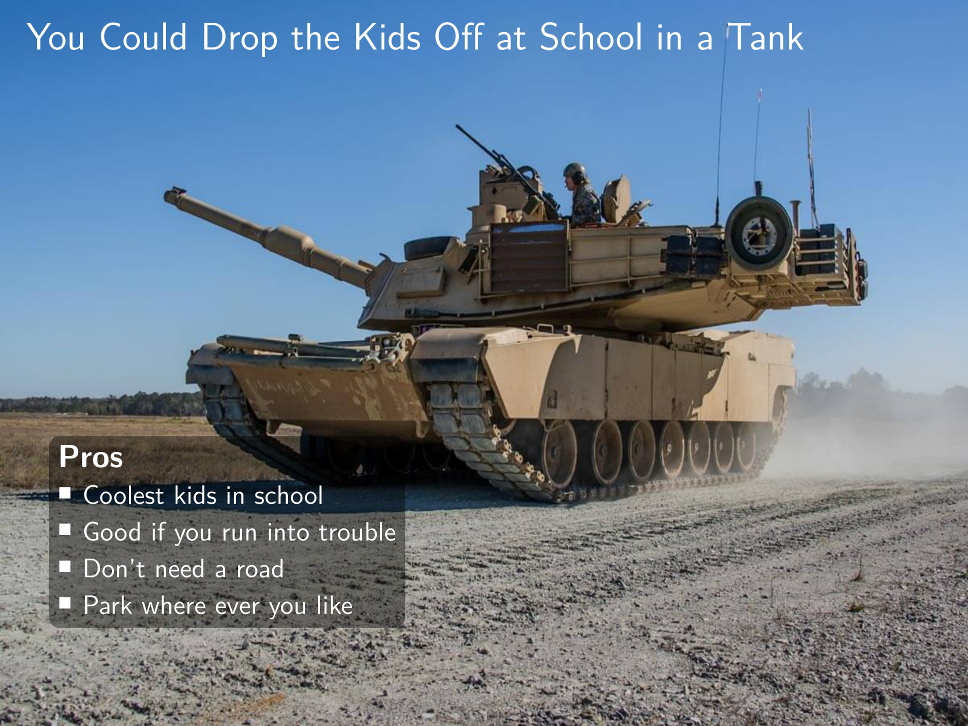
Discover applications requiring new features...

Not just trust = Legal Contract

Not just highly available = DNS, GAFA

You Could Drop the Kids Off at School in a Tank

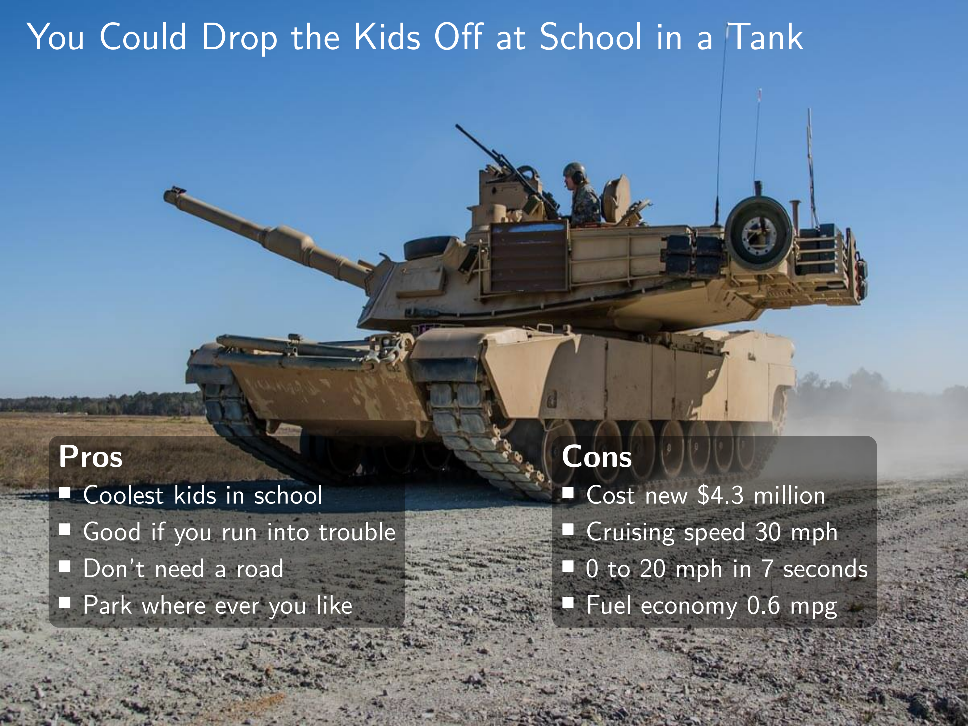
You Could Drop the Kids Off at School in a Tank



Pros

- Coolest kids in school
- Good if you run into trouble
- Don't need a road
- Park where ever you like

You Could Drop the Kids Off at School in a Tank



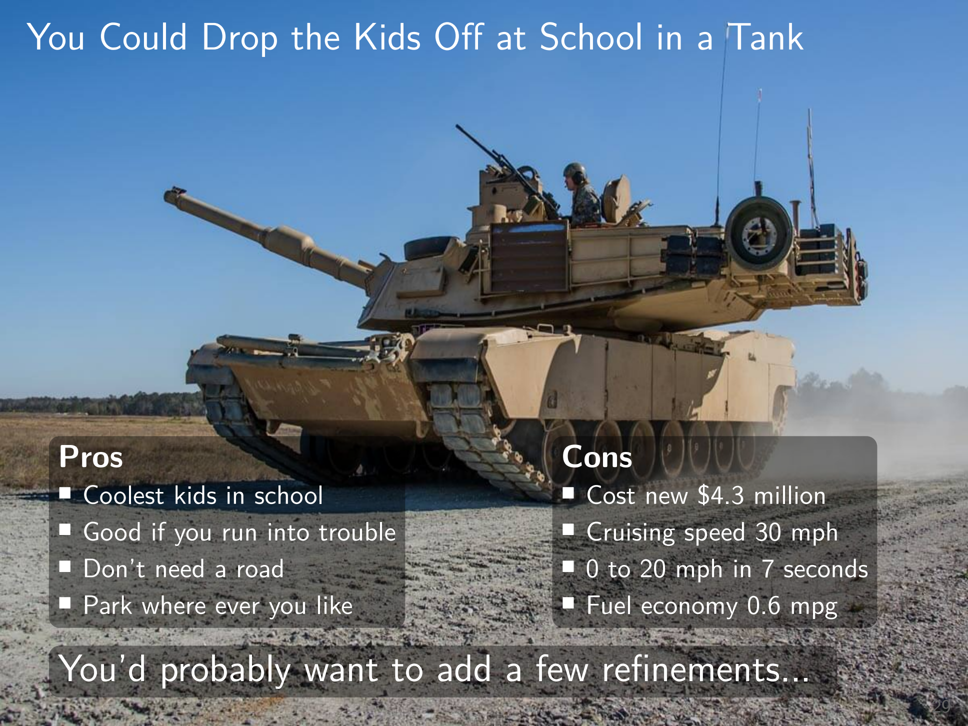
Pros

- Coolest kids in school
- Good if you run into trouble
- Don't need a road
- Park where ever you like

Cons

- Cost new $4.3 million
- Cruising speed 30 mph
- 0 to 20 mph in 7 seconds
- Fuel economy 0.6 mpg

You Could Drop the Kids Off at School in a Tank



Pros

- Coolest kids in school
- Good if you run into trouble
- Don't need a road
- Park where ever you like

Cons

- Cost new $4.3 million
- Cruising speed 30 mph
- 0 to 20 mph in 7 seconds
- Fuel economy 0.6 mpg

You'd probably want to add a few refinements...

...and you'd likely end up with a...

...SQL database

Capability Refinements Are In Conflict

Between	**and**	there is a **conflict**
Obvious TTP	Blockchain	Trusted third party administers SQL DB
Public	Permissioned	Coordinate without blockchain
Open source	Governance	Uncoordinated open network = forks
Privacy	**Verifiability**	Information needed to verify transactions
Trust	Performance	Low/no trust = poor performance
Access	Efficiency	Guaranteed access, distributed = expensive
PII	Public	Expectation of privacy
PII	Immutable	GDPR Right to be forgotten
Me	Everyone else	**Coordination** or **technology** problem?

Capability Refinements Are In Conflict

Between	**and**	there is a **conflict**
Obvious TTP	Blockchain	Trusted third party administers SQL DB
Public	Permissioned	Coordinate without blockchain
Open source	Governance	Uncoordinated open network = forks
Privacy	**Verifiability**	Information needed to verify transactions
Trust	Performance	Low/no trust = poor performance
Access	Efficiency	Guaranteed access, distributed = expensive
PII	Public	Expectation of privacy
PII	Immutable	GDPR Right to be forgotten
Me	Everyone else	**Coordination** or **technology** problem?

- Confidential transactions can keep the amount and type of assets transferred visible only to participants in the transaction (and those they choose to reveal the blinding key to), while still cryptographically guaranteeing that no more coins can be spent than are available

Identity is the Killer App

Self-Sovereign Identity and Decentralized Identifiers (DIDs)

- Permanent
- Resolvable
- Cryptographically Verifiable
- Decentralized

Identity is the Killer App

Self-Sovereign Identity and Decentralized Identifiers (DIDs)

- Permanent
- Resolvable
- Cryptographically Verifiable
- Decentralized

"*No identifier in history has had all four of these properties—because what fundamentally enables DIDs is **blockchain technology***"

- Verifiable credentials, edge devices, no central stores of PII
- Learn more at round table Discussion Tuesday

Drummond Reed, Decentralized Identifiers (DIDs) The Fundamental Building Block of Self-Sovereign Identity https://goo.gl/Au4uBx