Blockchains

Christian Grothoff

Berner Fachhochschule

2.6.2023

Learning Objectives

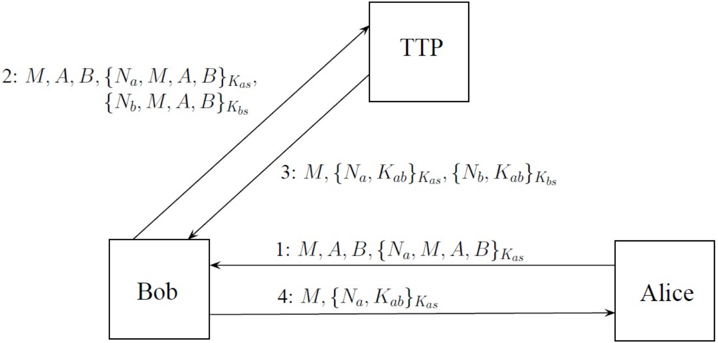
Blockchain

Name Systems: Background

Key Revocation

Homework (1/3)

The Otway-Ress protocol has a vulnerability described in `https://en.wikipedia.org/wiki/Otway-Rees`. Explain how that vulnerability is exploited.


TTP
2: $M, A, B, \{N_a, M, A, B\}_{K_{as}}, \{N_b, M, A, B\}_{K_{bs}}$
3: $M, \{N_a, K_{ab}\}_{K_{as}}, \{N_b, K_{ab}\}_{K_{bs}}$
1: $M, A, B, \{N_a, M, A, B\}_{K_{as}}$
4: $M, \{N_a, K_{ab}\}_{K_{as}}$
Bob
Alice

Wikipedia...

- Gürgens and Peralta describe an attack which they name an arity attack. In this attack the intruder intercepts the second message and replies to B using the two ciphertexts from message 2 in message 3. In the absence of any check to prevent it, M (or perhaps M,A,B) becomes the session key between A and B and is known to the intruder.
- Cole describes both the Gürgens and Peralta arity attack and another attack in his book Hackers Beware. In this the intruder intercepts the first message, removes the plaintext A,B and uses that as message 4 omitting messages 2 and 3. This leaves A communicating with the intruder using M (or M,A,B) as the session key.

Homework (2/3)

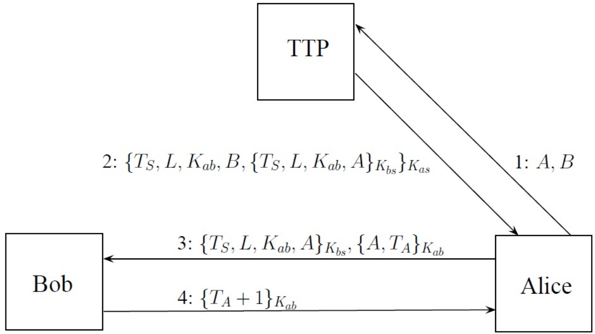
1. Verify that for any of the symmetric key establishment protocols the following holds: If an attacker manages to compromise the long term keys at some point in time, then he can decrypt all past sessions, assuming that he has recorded the key establishment protocol messages.
2. Recapitulate the notion of *known session-key attacks*. Argue why the Kerberos and the Otway-Ress protocol are secure against known session-key attacks.

Known Session Key Attack

Suppose Mallory is able to obtain the old session key K between Alice and Bob.
A protocol is vulnerable to a known session key attack if:

- Mallory can initiate a new session with Bob.
- Bob thinks he has a "new" session key K shared with Alice, but Alice does not know the key, since she might have thrown away K after the previous session with Bob.
- Mallory knows this session key K and communicates with Bob using K, impersonating Alice

Kerberos



TTP

Bob

Alice

1: A, B

2: $\{T_S, L, K_{ab}, B, \{T_S, L, K_{ab}, A\}_{K_{bs}}\}_{K_{as}}$

3: $\{T_S, L, K_{ab}, A\}_{K_{bs}}, \{A, T_A\}_{K_{ab}}$

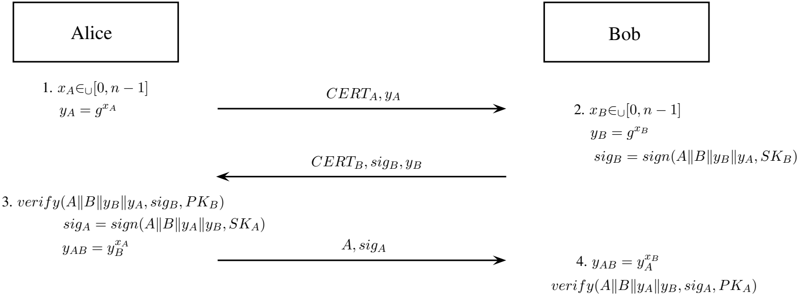
4: $\{T_A + 1\}_{K_{ab}}$

Homework (3/3)

1. Assume that an attacker can control the clocks of Alice, Bob, and the KDC, respectively. Can you come up with any attacks on the Kerberos protocol?
2. Describe a man in the middle attack on the Diffie-Hellman protocol. Why does the attack fail on the station to station protocol?

Station to station key agreement protocol

Common input: $\mathbb{Z}^*{}_p$ and $g \in \mathbb{Z}^*{}_p$, and n such that $g^n \equiv 1 \mod p$



Alice

Bob

1. $x_A \in_{\cup} [0, n-1]$
   $y_A = g^{x_A}$

$CERT_A, y_A$

2. $x_B \in_{\cup} [0, n-1]$
   $y_B = g^{x_B}$
   $sig_B = sign(A \| B \| y_B \| y_A, SK_B)$

$CERT_B, sig_B, y_B$

3. $verify(A \| B \| y_B \| y_A, sig_B, PK_B)$
   $sig_A = sign(A \| B \| y_A \| y_B, SK_A)$
   $y_{AB} = y_B^{x_A}$

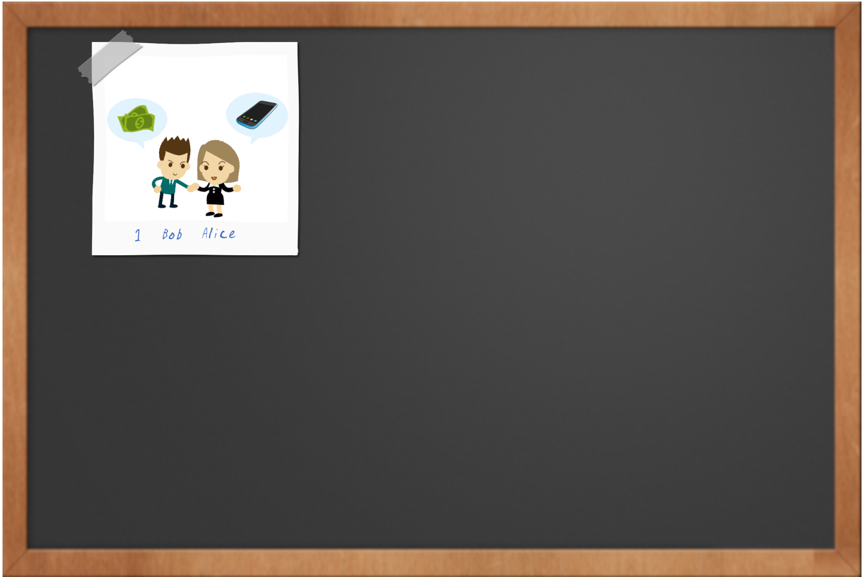
A, sig_A

4. $y_{AB} = y_A^{x_B}$
   $verify(A \| B \| y_A \| y_B, sig_A, PK_A)$

Blockchain[1]

[1] Illustrations by Alexandra Dirksen, IAS, TUBS [1]

Blockchain

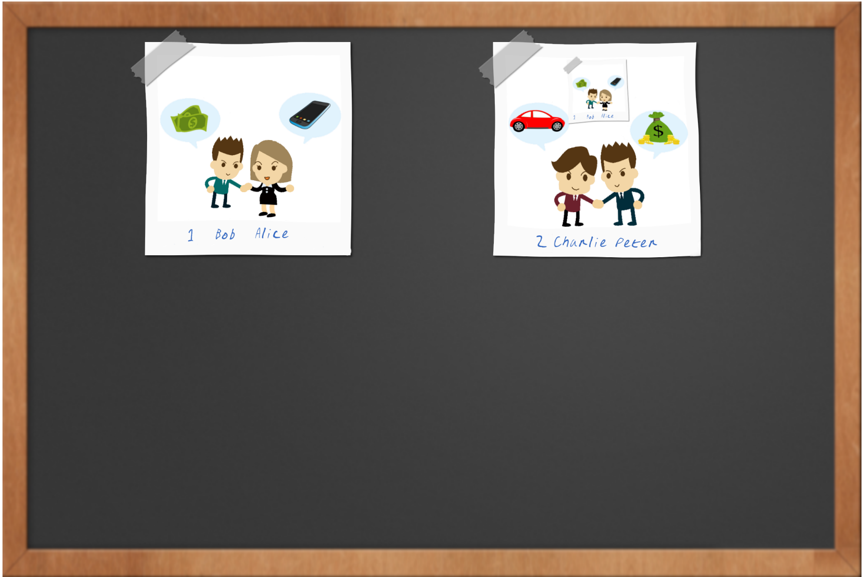


Blockchain

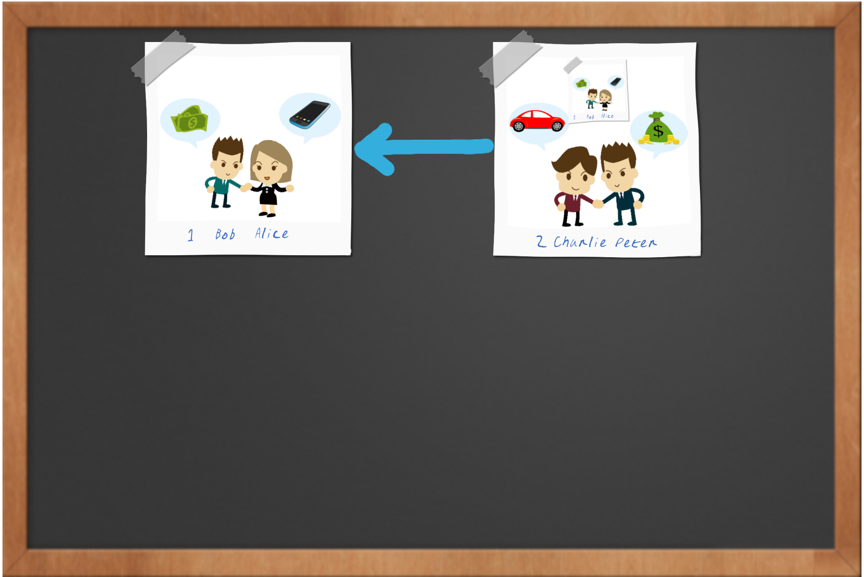
Charlie
Peter

Blockchain

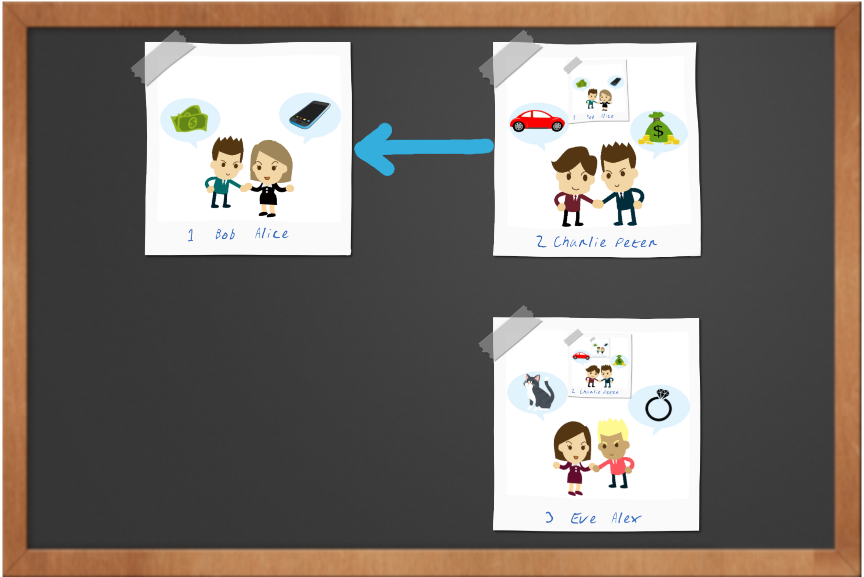
Blockchain



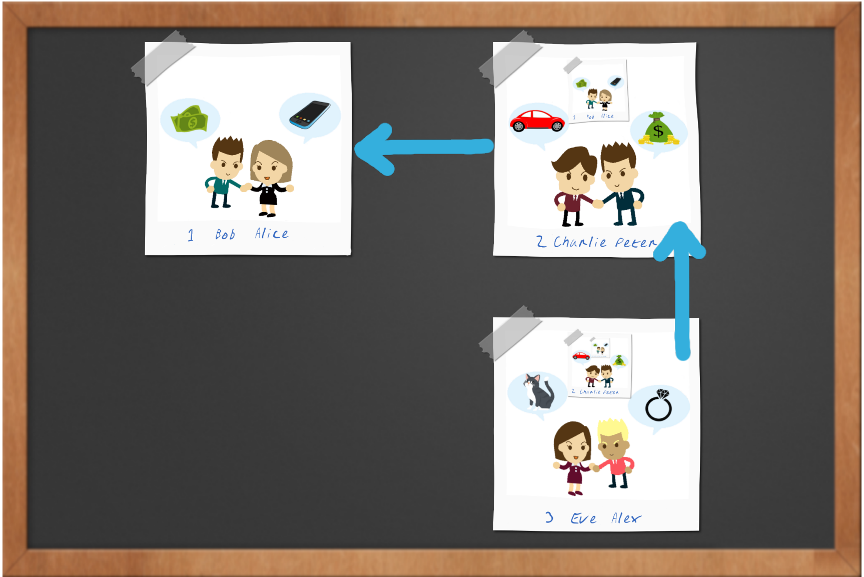
Blockchain



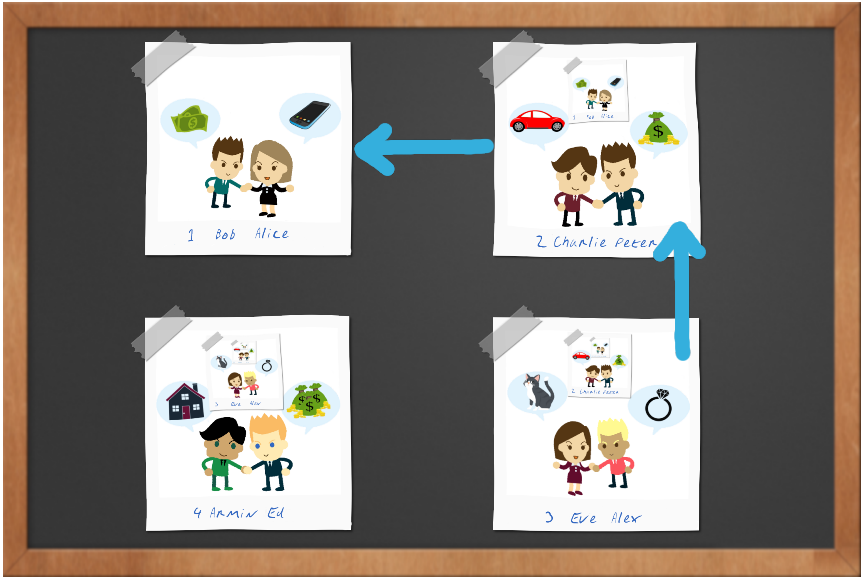
Blockchain



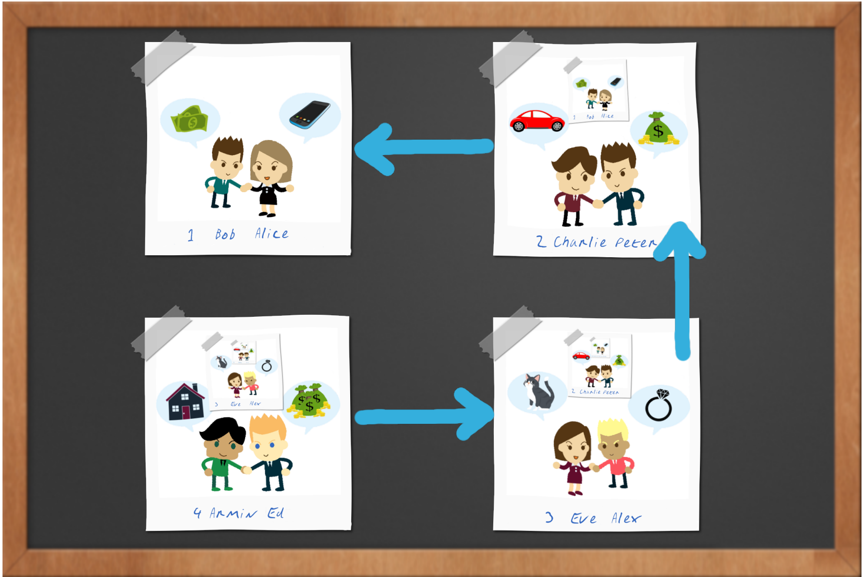
Blockchain



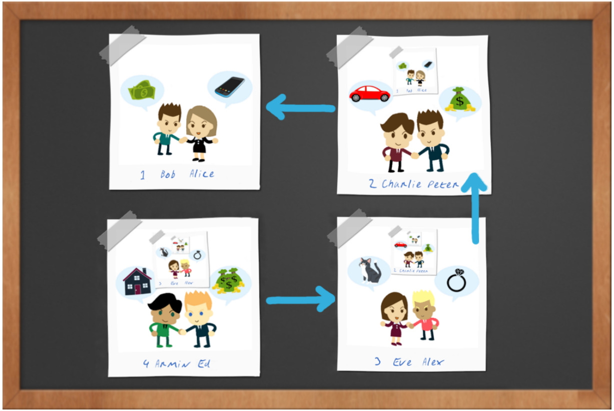
Blockchain



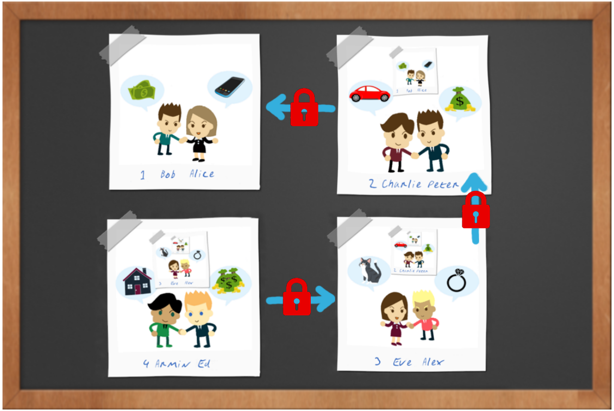
Blockchain



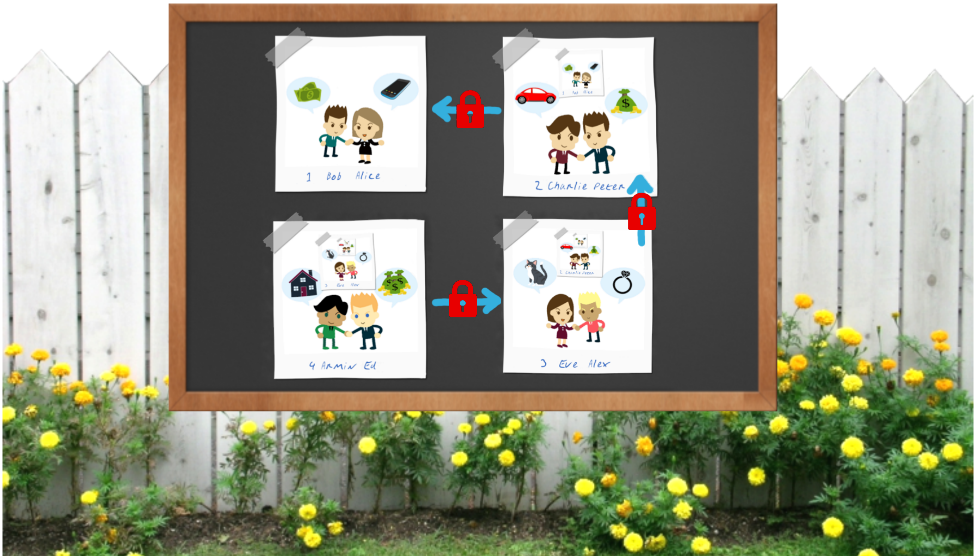
Advertised Blockchain "properties"



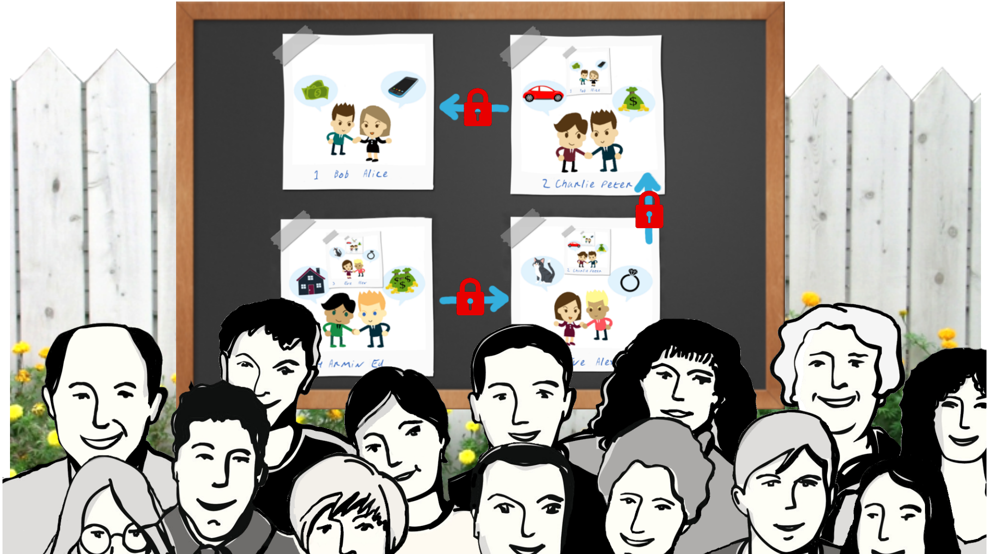
Immutability



Transparency



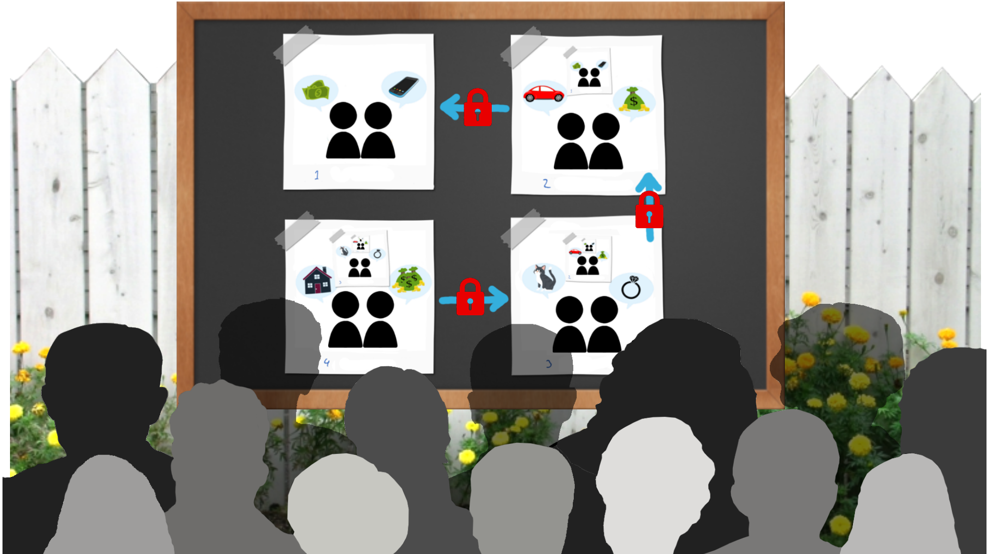
Decentralisation



Autonomy

1 Bob Alice
2 Charlie Peter
4 Armin Ed
3 Eve Alex

Anonymity

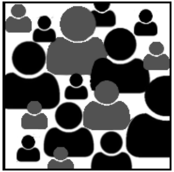


Blockchain "properties"[2]

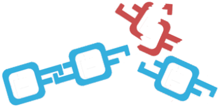
Immutability

Transparency

Anonymity



Decentralisation



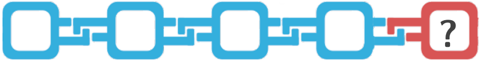
Irreversibility

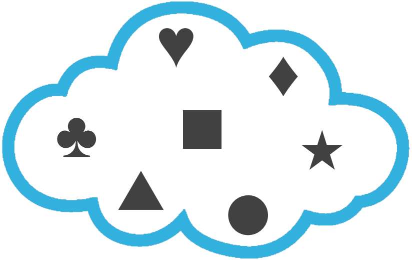
Autonomy

[2] These **only** hold with many **significant caveats**!

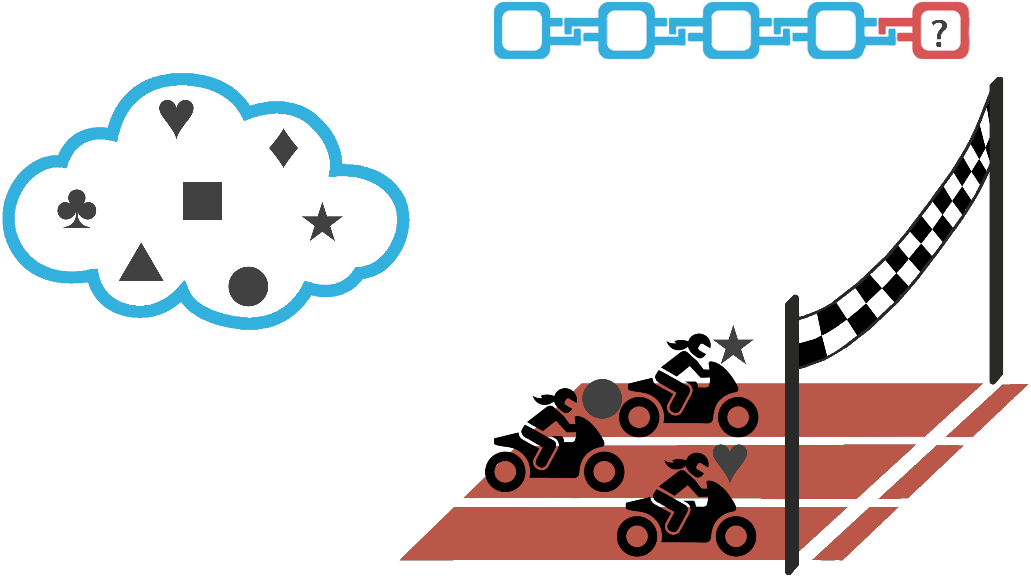
Who gets to append the next block?

Proof of Work

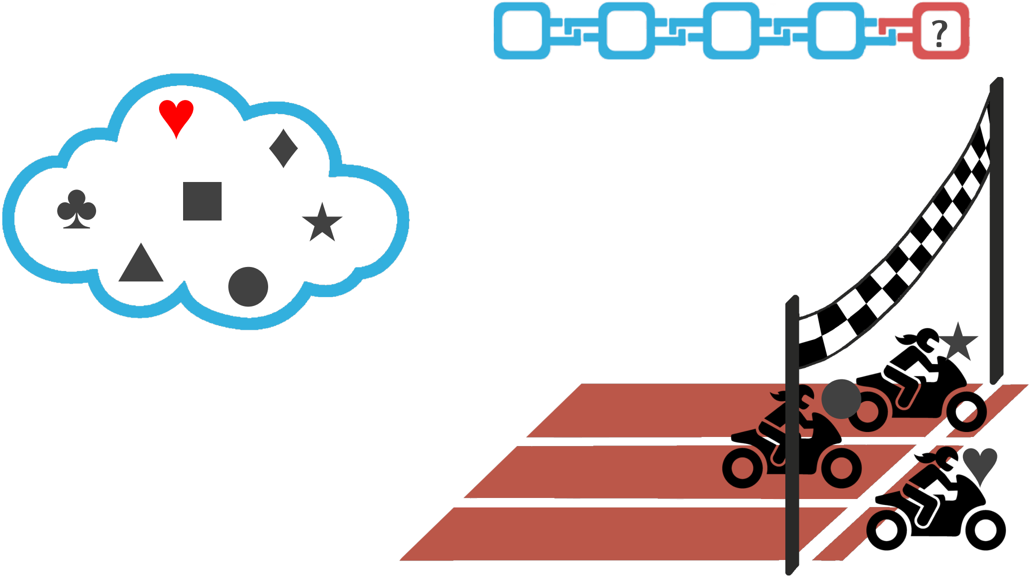




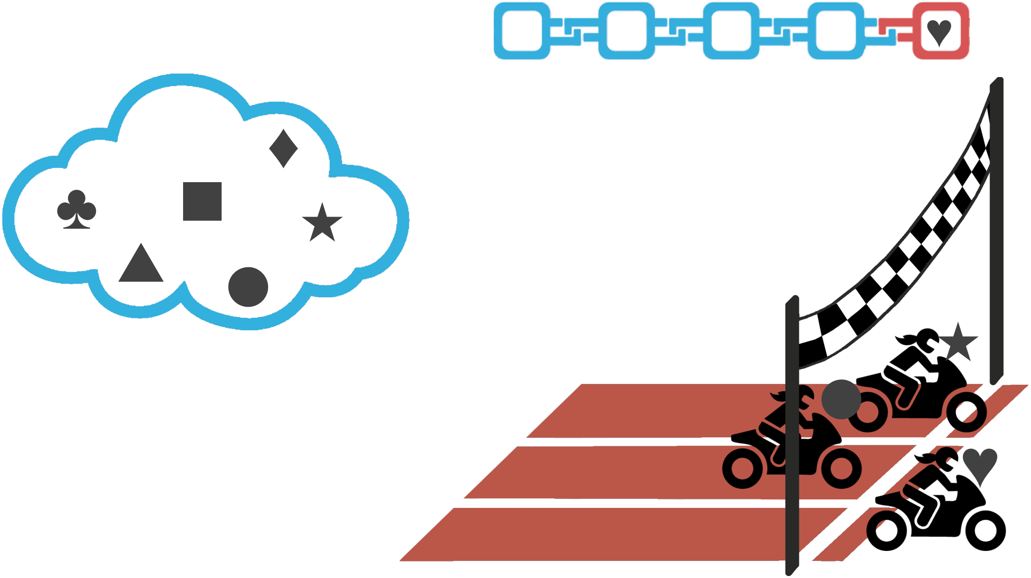
Proof of Work



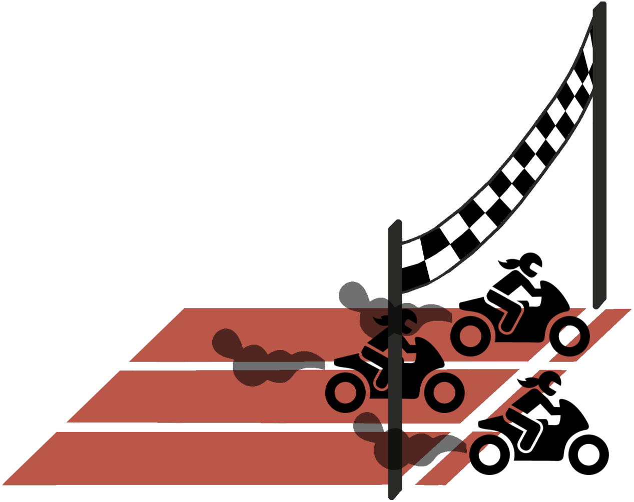
Proof of Work



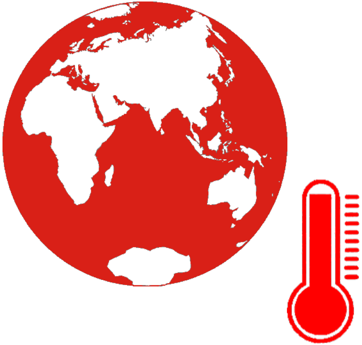
Proof of Work

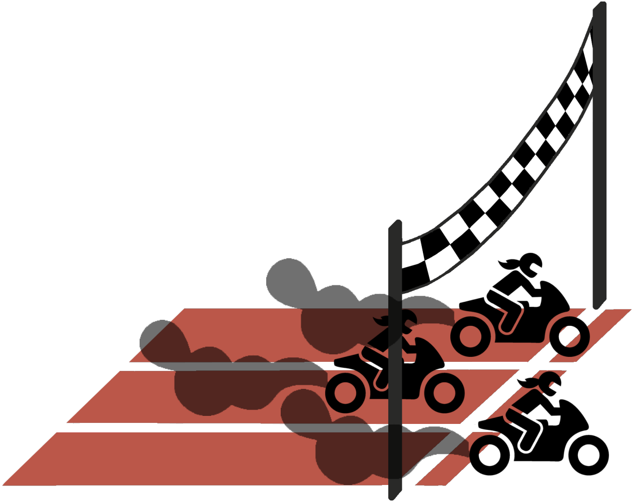


Proof of Work



Proof of Work





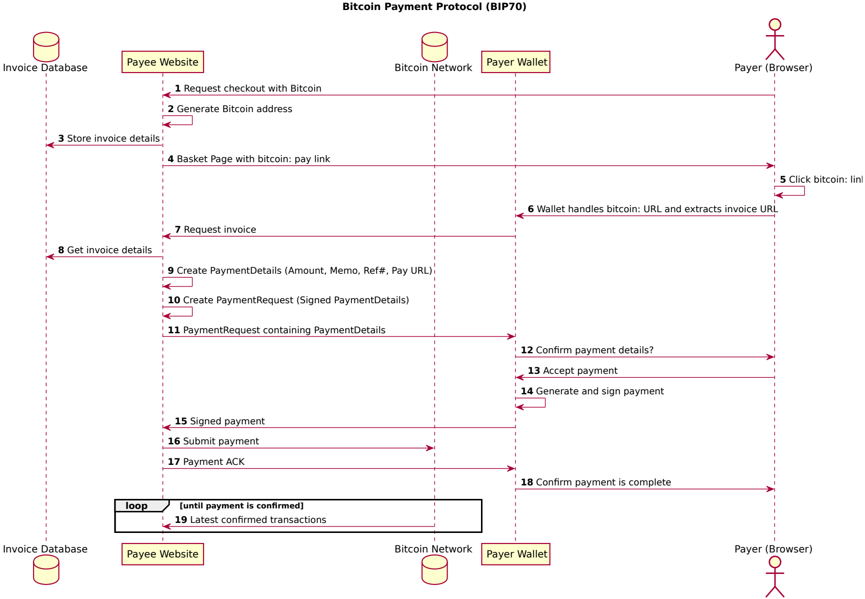
Break

Bitcoin for Payments

Bitcoin claims to be a payment system using a block chain:

- Public keys identify accounts, private keys used to send money from the account into other accounts.
- Set of internally consistent transactions form block
- Each block includes a transaction creating fresh coins and transferring applicable fees to block creator
- Difficulty adjusts to mining power to mine a block in ≈ 10 minutes
- Amount of coins created per block is exponentially decreasing

Bitcoin Payment flow (by W3C Payment Interest Group)



Bitcoin Payment Protocol (BIP70)

Invoice Database

Payee Website

Bitcoin Network

Payer Wallet

Payer (Browser)

1 Request checkout with Bitcoin

2 Generate Bitcoin address

3 Store invoice details

4 Basket Page with bitcoin: pay link

5 Click bitcoin: linl

6 Wallet handles bitcoin: URL and extracts invoice URL

7 Request invoice

8 Get invoice details

9 Create PaymentDetails (Amount, Memo, Ref#, Pay URL)

10 Create PaymentRequest (Signed PaymentDetails)

11 PaymentRequest containing PaymentDetails

12 Confirm payment details?

13 Accept payment

14 Generate and sign payment

15 Signed payment

16 Submit payment

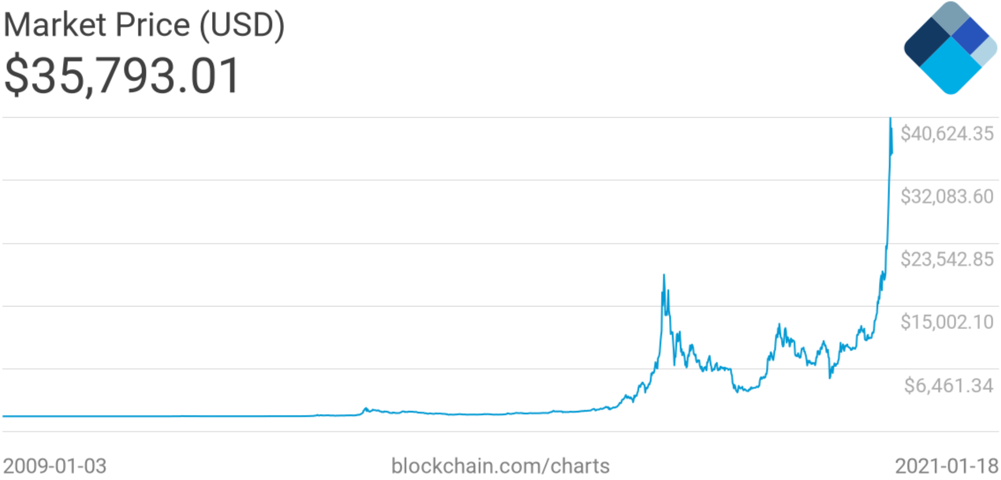
17 Payment ACK

18 Confirm payment is complete

loop **[until payment is confirmed]**

19 Latest confirmed transactions

The Value of Bitcoin



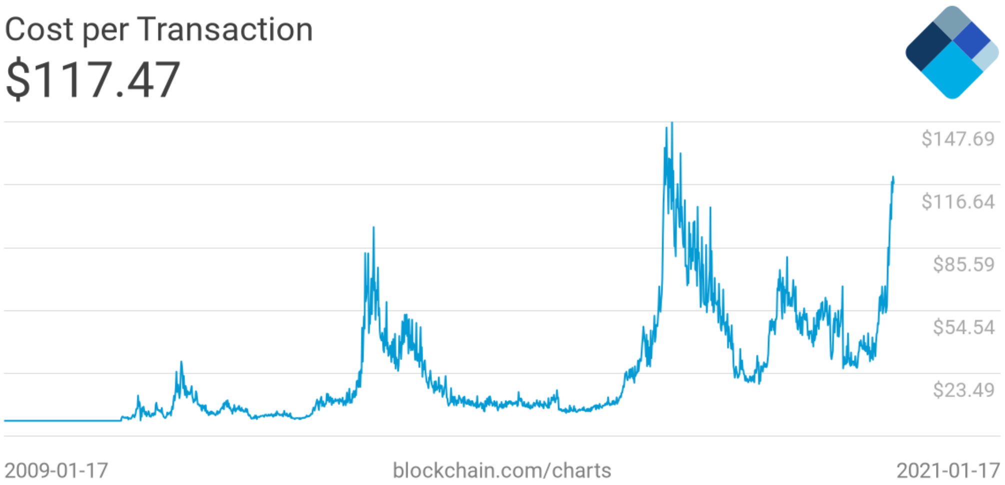
Mining

Mining requires:

- Learning pending transactions from peers
- Selecting a subset of of transactions which is valid (no double spending) by computing current account balances against the entire history
- Finding a hash collision (with adaptive difficulty)
- Propagating the new block to other miners

Usually specialized systems are used for finding hash collisions.

Mining cost



Cost per Transaction

$117.47

Current average transaction value: ≈ 1000 USD

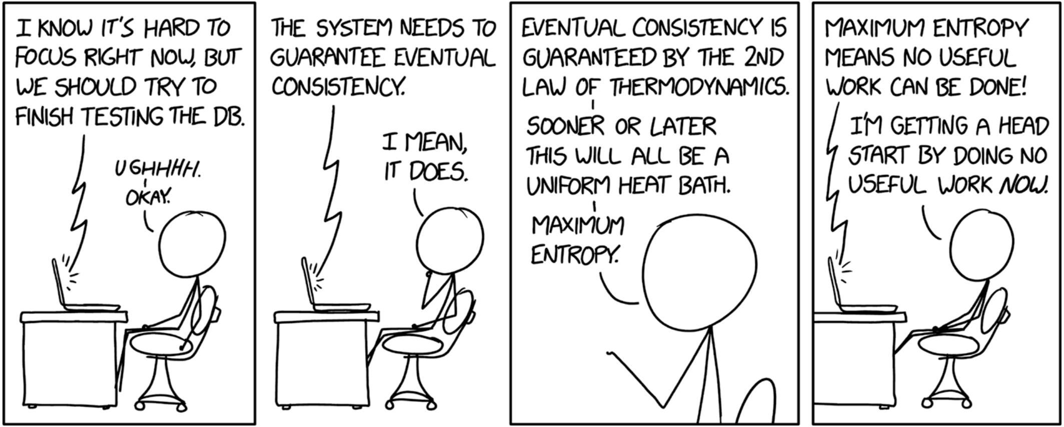
CAP & Bitcoin

Bitcoin is inconsistent:

- Conflicting blocks can be mined at the same time
- This can happen by accident, or on purpose!
- Coins could be spent twice, once on each fork of the chain!
- Longest chain is considered “valid”
- Original paper suggests to consider transaction confirmed only after at least 6 blocks past the transaction.

⇒ Bitcoin is not consistent.

⇒ Competitively long alternative chains void durability even after 6 blocks!


I KNOW IT'S HARD TO FOCUS RIGHT NOW, BUT WE SHOULD TRY TO FINISH TESTING THE DB.
UGHHHH.
OKAY.
THE SYSTEM NEEDS TO GUARANTEE EVENTUAL CONSISTENCY.
I MEAN, IT DOES.
EVENTUAL CONSISTENCY IS GUARANTEED BY THE 2ND LAW OF THERMODYNAMICS.
SOONER OR LATER THIS WILL ALL BE A UNIFORM HEAT BATH.
MAXIMUM ENTROPY.
MAXIMUM ENTROPY MEANS NO USEFUL WORK CAN BE DONE!
I'M GETTING A HEAD START BY DOING NO USEFUL WORK *NOW*.

https://xkcd.com/2315/

Bitcoin performance

- Privacy: all transactions happen in the clear in public view
- Latency: transactions take 1h to kind-of be confirmed
- Storage: grows linearly forever, no garbage collection
- Power: mining consumes more than the entire state of Denmark today
- Rate: Network handles at most about 7 transactions per second
- Accountability: use of public keys as addresses enables criminal use

$\Rightarrow$ Bitcoin fever lasting for years. Why?

James Mickens on Blockchains

Altcoins

- Dogecoin: same as Bitcoin, just named after a dog meme (an idea that is obviously worth billions!)
- Zcash: uses ZKSNARKs[3] to hide transactions (criminal activity on Bitcoin was too low)
- Ethereum: run Turing-complete virtual machine logic in the blockchain to enable "smart" contracts and arbitrary applications, not just payments (is "Accelerando" an utopia or dystopia?)

Experimental designs promising to drastically improve performance (Bolt, Lightning) have so far failed to deliver.

[3] ≈ 1-15 minutes CPU time to create new transaction needed!

Case study: Private payments

> *"A company is developing new software for private payments. This will enable its customers to transact with "complete" privacy (like cash). The solution does not include backdoors, and thus the company cannot block payments to support trade embargos or anti money laundering efforts."*

- Discuss virtues and vices affected.
- Does it make a difference if the software is Free Software developed by an individual or a community instead of proprietary software from a company?
- Have you heard about Alexey Pertsev?

Break

Security Goals for Name Systems

- Query origin anonymity
- Data origin authentication and integrity protection
- Zone confidentiality
- Query and response privacy
- Censorship resistance
- Traffic amplification resistance
- Availability

Approaches Adding Cryptography to DNS

- DNSSEC
- DNSCurve
- DNS-over-TLS
- DNS-over-HTTPS
- RAINS

Case study: DoH

DNS is known to suffer from a lack of end-to-end integrity protections. As a result, Chinese "great firewall" DNS manipulation has been shown to impact name resolution even in Europe.

> *"The IETF is standardizing DNS over HTTPS (DOH), where all DNS queries are sent over the HTTPS protocol to some well-known HTTPS server (such as Google's 8.8.8.8 or Cloudflare's 1.1.1.1). This will prevent local governments from manipulating DNS traffic and improve the user's privacy with respect to their ISPs and governments. However, Google or Cloudflare will see the DNS queries and replies of the users, and they must be expected to have weak privacy policies and are subject to US law which includes secret rules and court orders. The NSA has a history of snooping on (MORECOWBELL) and manipulating (QUANTUMDNS) DNS traffic."*

Discuss virtues and vices affected.

Case study: RAINS

DNS is known to suffer from a lack of end-to-end integrity protections. As a result, Chinese "great firewall" DNS manipulation has been shown to impact name resolution even in Europe.

> *"The ETH Zurich is developing a new name system called RAINS with a new trust anchor operated by the regional Internet service provides, aka the local Isolation Service Domain (ISD). RAINS does not change the privacy of DNS (provides can continue to monitor traffic, all zone data becomes public) and allows the local authorities to block Web sites to improve public safety and enforce local laws (see also: "Glücksspielgesetz in Switzerland"). At the same time, foreign censorship efforts are less likely to be effective (unless they foreign government forces the DNS authority to alter the authoritative records)."*

Discuss virtues and vices affected.

Break

Namecoin

Let’s just put the records into the Blockchain!

Namecoin

Let's just put the records into the Blockchain!

Or rather, put the public key of the owner and signed updates into it.

Namecoin

Let’s just put the records into the Blockchain!

Or rather, put the public key of the owner and signed updates into it.

And let’s have some expiration rules.

Case study: Namecoin

DNS is known to suffer from a lack of end-to-end integrity protections. As a result, Chinese "great firewall" DNS manipulation has been shown to impact name resolution even in Europe.

> *"Namecoin establishes a new name system on the blockchain (where thus zone data is also public), but where public authorities cannot block information. Queries are performed against a local copy of the blockchain and thus also private. There is no WHOIS, so the owner of a name can also be anonymous. However, Namecoin uses much more bandwidth and energy as blockchain payments are used for registration and name resolution. Names are registered on a first-come, first-served basis. Trademarks, copyrights anti-fraud or anti-terrorism judgements cannot be used to force owners of names to relinquish names."*

Discuss virtues and vices affected.

Ethereum Name System[4]

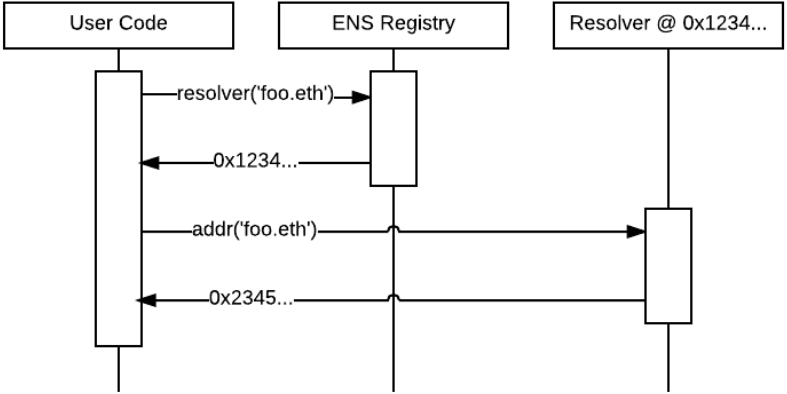
Let's have a smart contract in the Blockchain manage naming!

Blockchain contains smart contract and data who controls which name.

Contract allocates names under `.eth` using auctions.

[4] `https://ens.domains/`

Ethereum Name System[5]



[5] https://ens.domains/

Break

Key Revocation

- Certificate Revocation Lists (X.509)
- Online Certificate Status Protocol (OCSP)
- OCSP stapling (TLS)
- Publish revocation in blockchain?
- Controlled flooding

Key Revocation via Controlled Flooding

- Revocation message signed with private key that is to be revoked
- Flooded on all links in (P2P) overlay, stored forever
- Efficient set reconciliation used when peers connect
- Expensive **proof-of-work** used to limit DoS-potential
- Proof-of-work can be calculated ahead of time
- Revocation messages can be computed and stored off-line if desired

Efficient Set Union

(based on “What’s the difference? Efficient Set Reconciliation without Prior Context” [2])

- Alice and Bob have sets A and B

- The sets are very large
- …but their symmetric difference $\delta = |(A - B) \cup (B - A)|$ is small

- Now Alice wants to know $B - A$ (the elements she’s missing)
- …and Bob $A - B$ (the elements he’s missing)

- How can Alice and Bob do this efficiently?
  - w.r.t. communication and computation

Bad Solution

- Naive approach: Alice sends A to Bob, Bob sends $B - A$ back to Alice
- . . . and vice versa.

- Communication cost: $O(|A| + |B|)$:(
- Ideally, we want to do it in $O(\delta)$.
- First improvement: Don't send elements of A and B, but send/request hashes. Still does not improve complexity :(

- We need some more fancy data structure!

Bloom Filters

Constant size data structure that "summarizes" a set.

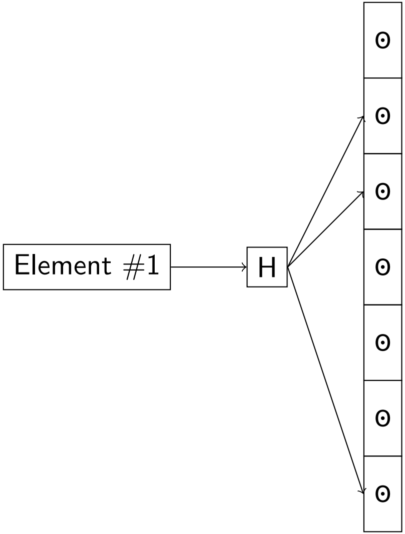
Operations:

$d = NewBF(size)$ Create a new, empty bloom filter.

$Insert(d, e)$ Insert element e into the BF d.

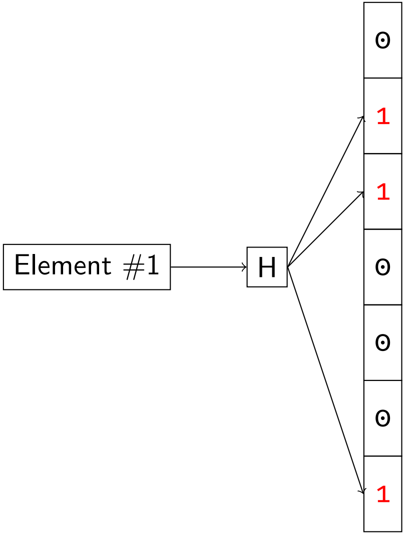
$b = Contains(d, e)$ Check if BF d contains element e.
$b \in \{$"Definitely not in set", "Probably in set"$\}$

BF: Insert



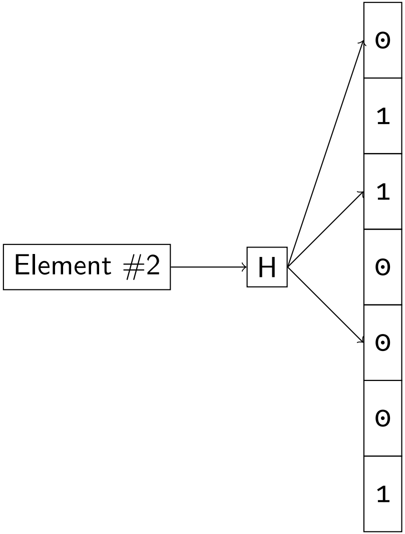
$H(\text{Element } \#1) = (2, 3, 7)$

BF: Insert



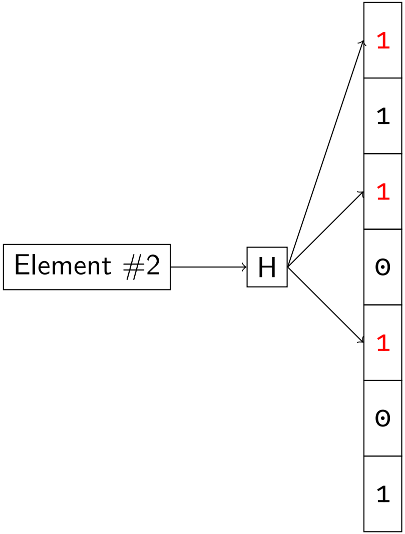
$$H(\text{Element }\#1) = (2, 3, 7)$$

BF: Insert



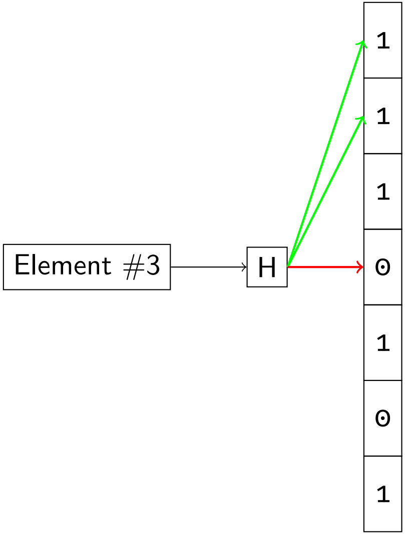
$$H(\text{Element } \#1) = (2, 3, 7)$$
$$H(\text{Element } \#2) = (1, 3, 5)$$

BF: Insert



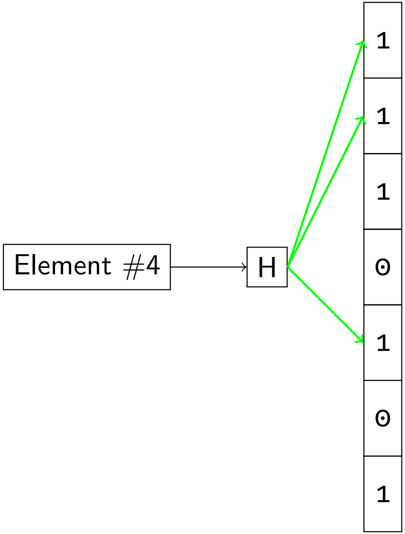
$$H(\text{Element } \#1) = (2, 3, 7)$$
$$H(\text{Element } \#2) = (1, 3, 5)$$

BF: Membership Test



$H(\text{Element }\#1) = (2, 3, 7)$
$H(\text{Element }\#2) = (1, 3, 5)$

BF: Membership Test (false positive)



$H(\text{Element } \#1) = (2, 3, 7)$
$H(\text{Element } \#2) = (1, 3, 5)$

Counting Bloom Filters

BF where buckets hold a **positive integer**.

Additional Operation:

Remove(d, e) Remove element from the CBF d.

$\Rightarrow$ False negatives when removing a non-existing element.

Invertible Bloom Filters

Similar to CBF, but

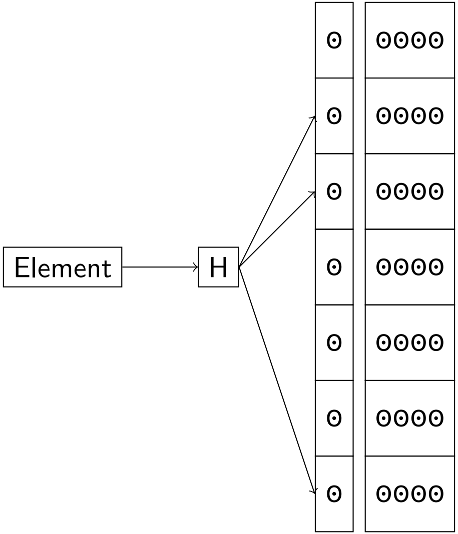
- Allow **negative counts**
- Additionaly store **(XOR-)sum of hashes** in buckets.

Additional Operations:

$(e, r) = \mathit{Extract}(d)$ Extract an element (e) from the IBF d, with result code $r \in \{\mathit{left}, \mathit{right}, \mathit{done}, \mathit{fail}\}$

$d' = \mathit{SymDiff}(d_1, d_2)$ Create an IBF that represents the symmetric difference of d_1 and d_2.

IBF: Insert



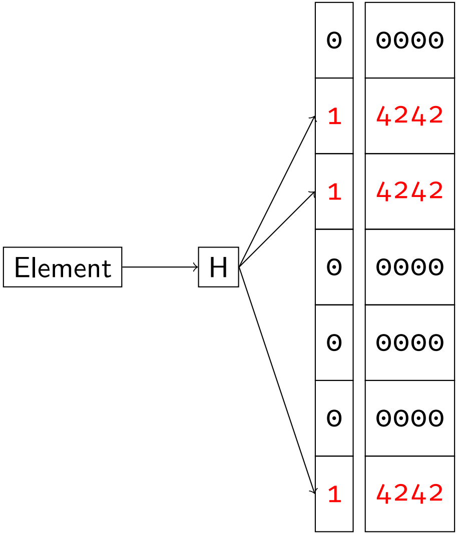
Element

H

0	0000
0	0000
0	0000
0	0000
0	0000
0	0000
0	0000

$$H(\text{Element } \#1) = (2, 3, 7)$$
$$H'(\text{Element } \#1) = 4242$$

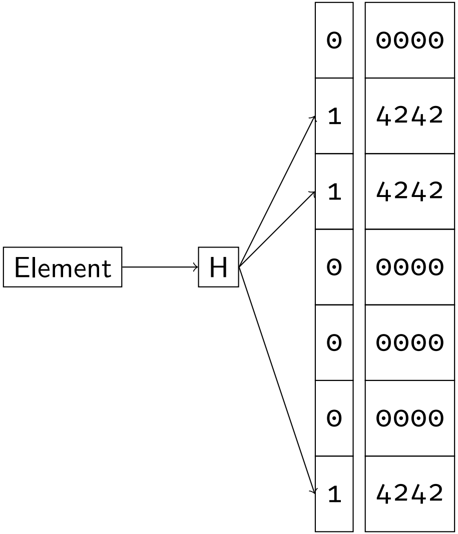
IBF: Insert



0	0000
1	4242
1	4242
0	0000
0	0000
0	0000
1	4242

$$H(\text{Element }\#1) = (2, 3, 7)$$
$$H'(\text{Element }\#1) = 4242$$

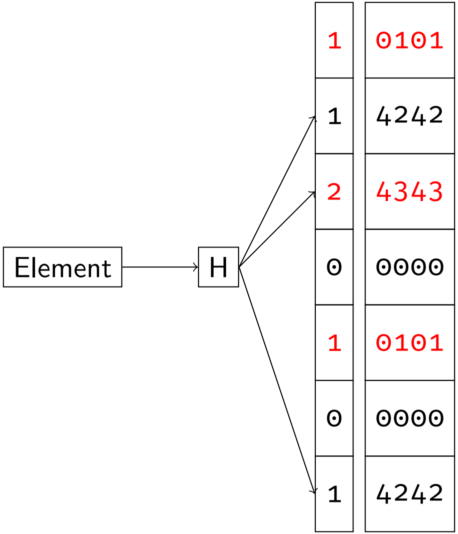
IBF: Insert



0	0000
1	4242
1	4242
0	0000
0	0000
0	0000
1	4242

$$H(\text{Element } \#1) = (2, 3, 7)$$
$$H'(\text{Element } \#1) = 4242$$
$$H(\text{Element } \#2) = (1, 3, 5)$$
$$H'(\text{Element } \#2) = 0101$$

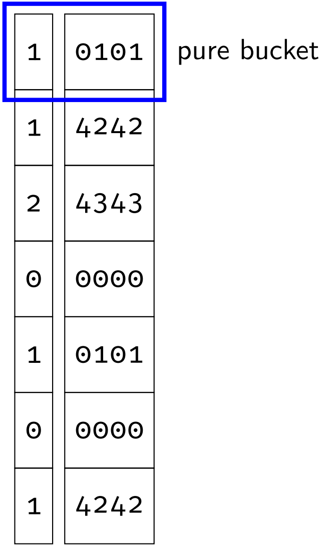
IBF: Insert



1	0101
1	4242
2	4343
0	0000
1	0101
0	0000
1	4242

$$H(\text{Element } \#1) = (2, 3, 7)$$
$$H'(\text{Element } \#1) = 4242$$
$$H(\text{Element } \#2) = (1, 3, 5)$$
$$H'(\text{Element } \#2) = 0101$$

IBF: Extract



1	0101
1	4242
2	4343
0	0000
1	0101
0	0000
1	4242

- Pure bucket ⇒ extractable element hash
- Extraction ⇒ more pure buckets (hopefully/probably)
- Less elements ⇒ more chance for pure buckets

Symmetric Difference

We can directly compute the symmetric difference without extraction.

- Subtract counts
- XOR hashes

The Set Union Protocol [3]

1. ⇒ Create IBFs
2. Compute SymDiff
3. Extract element hashes

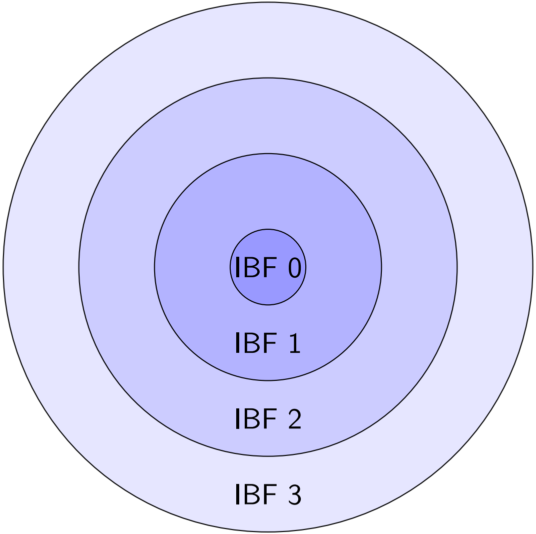
- Amount of communication and computation only depends on δ, not $|A| + |B|$:)
- How do we choose the initial size of the IBF?
- ⇒ Do difference estimation first!

Difference Estimation

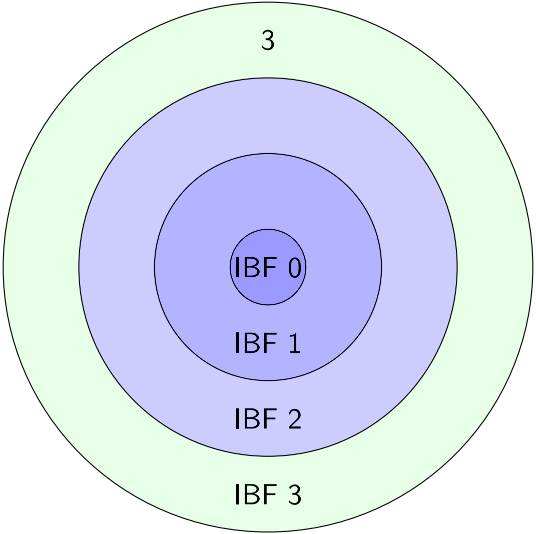
- We need an estimator that's accurate for small differences
- Turns out we can re-use IBFs for difference estimation:

1. Alice and Bob create fixed number of constant-size IBFs by sampling their set. The collection of IBFs is called a Strata Estimator (SE).
   - Stratum 1 contains 1/2 of all elements
   - Stratum 2 contains 1/4 of all elements
   - Stratum *n* contains $1/(2^n)$ all elements
2. Alice receives Bob's strata estimator
3. Alice computes $SE_{diff} = SymDiff(SE_{\mathrm{Alice}}, SE_{\mathrm{Bob}})$
   - by pair-wise *SymDiff* of all IBFs in the SE
4. Alice estimates the size of SE_{diff}.

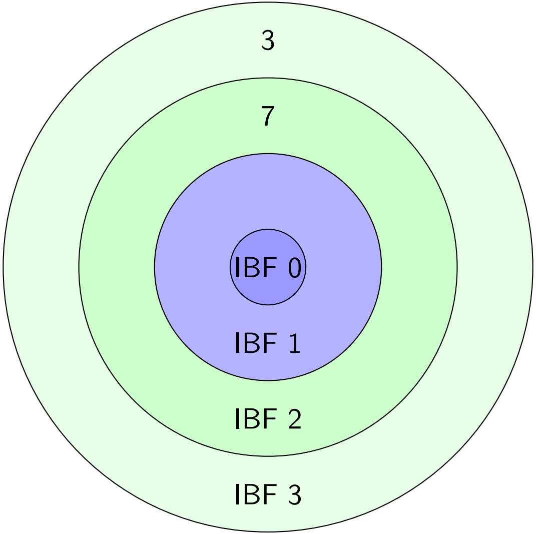
Strata Estimator



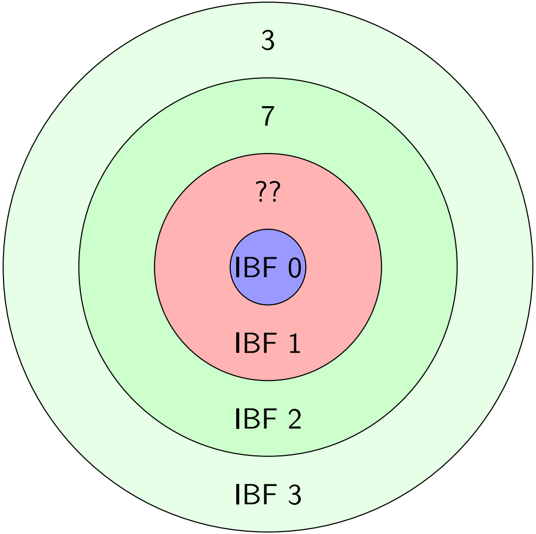
Strata Estimator



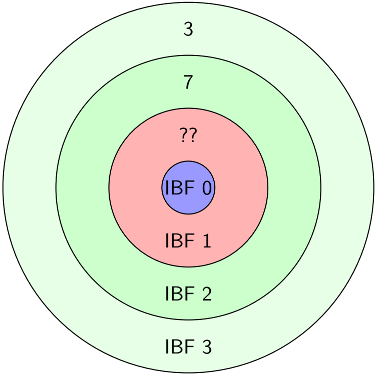
Strata Estimator



Strata Estimator



Estimation



Estimate as $(3 + 7) \cdot 2^4$.
(Number of extracted hashes scaled by expected number of elements in the remaining IBFs)

The Complete Protocol

1. Alice sends SE_{Alice} to Bob
2. Bob estimates the set difference δ
3. Bob computes IBF_{Bob} with size δ and sends it to Alice
4. Alice computes IBF_{Alice}
5. Alice computes $IBF_{diff} = SymDiff(IBF_{\mathrm{Alice}}, IBF_{\mathrm{Bob}})$
6. Alice extracts element hashes from IBF_{diff}.
   - $b = left \Rightarrow$ Send element to to Bob
   - $b = right \Rightarrow$ Send element request to to Bob
   - $b = fail \Rightarrow$ Send larger IBF (double the size) to Bob, go to (3.) with switched roles
   - $b = done \Rightarrow$ We're done ...

References

Alexandra Dirksen.
A blockchain picture book.
https://media.ccc.de/v/35c3-9573-a˙blockchain˙picture˙book), 12 2018.

David Eppstein, Michael T. Goodrich, Frank Uyeda, and George Varghese.
What's the difference?: Efficient set reconciliation without prior context.
SIGCOMM Comput. Commun. Rev., 41(4):218–229, August 2011.

Elias Summermatter and Christian Grothoff.
Byzantine fault tolerant set reconciliation.
https://datatracker.ietf.org/doc/html/draft-summermatter-set-union, 1 2021.