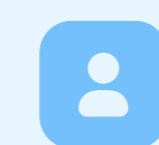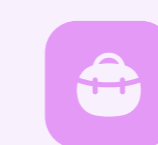# Decentralized Collaborative Version Control

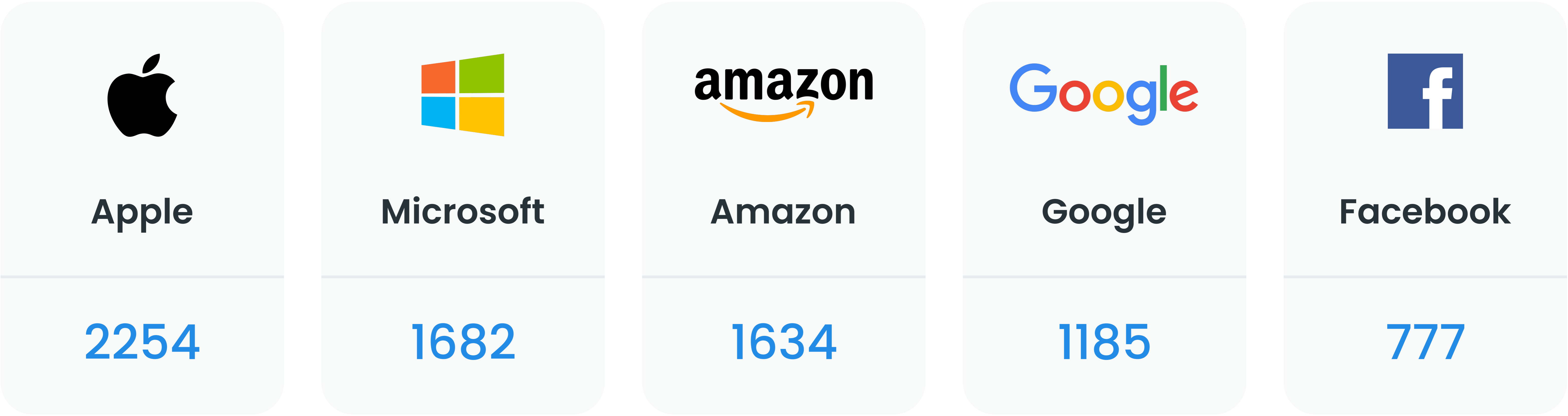## DICG'21

Bulat Nasrulin, Johan Pouwelse

TU Delft

Centralised Services Dominate the Market

5 Largest Corporations by Market Capitalisation

Apple	Microsoft	Amazon	Google	Facebook
2254	1682	1634	1185	777

*Asset evaluation in billions of dollars

Decentralized Services as an Alternative

Decentralized Services as an Alternative

All Crypto

2450

Number of Projects

11260

* As for December 5, 2021

Decentralized Services as an Alternative
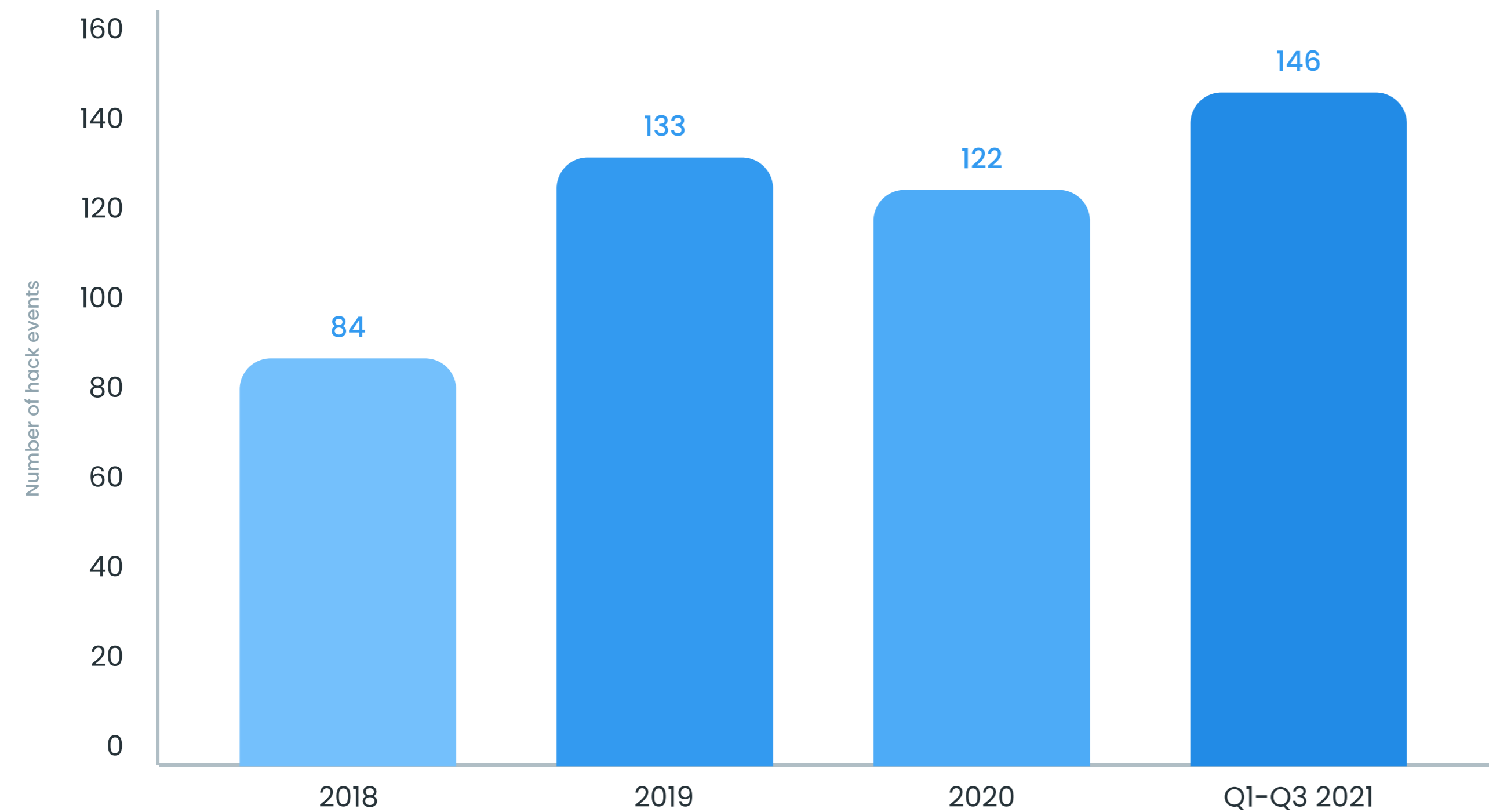
All Crypto

2450

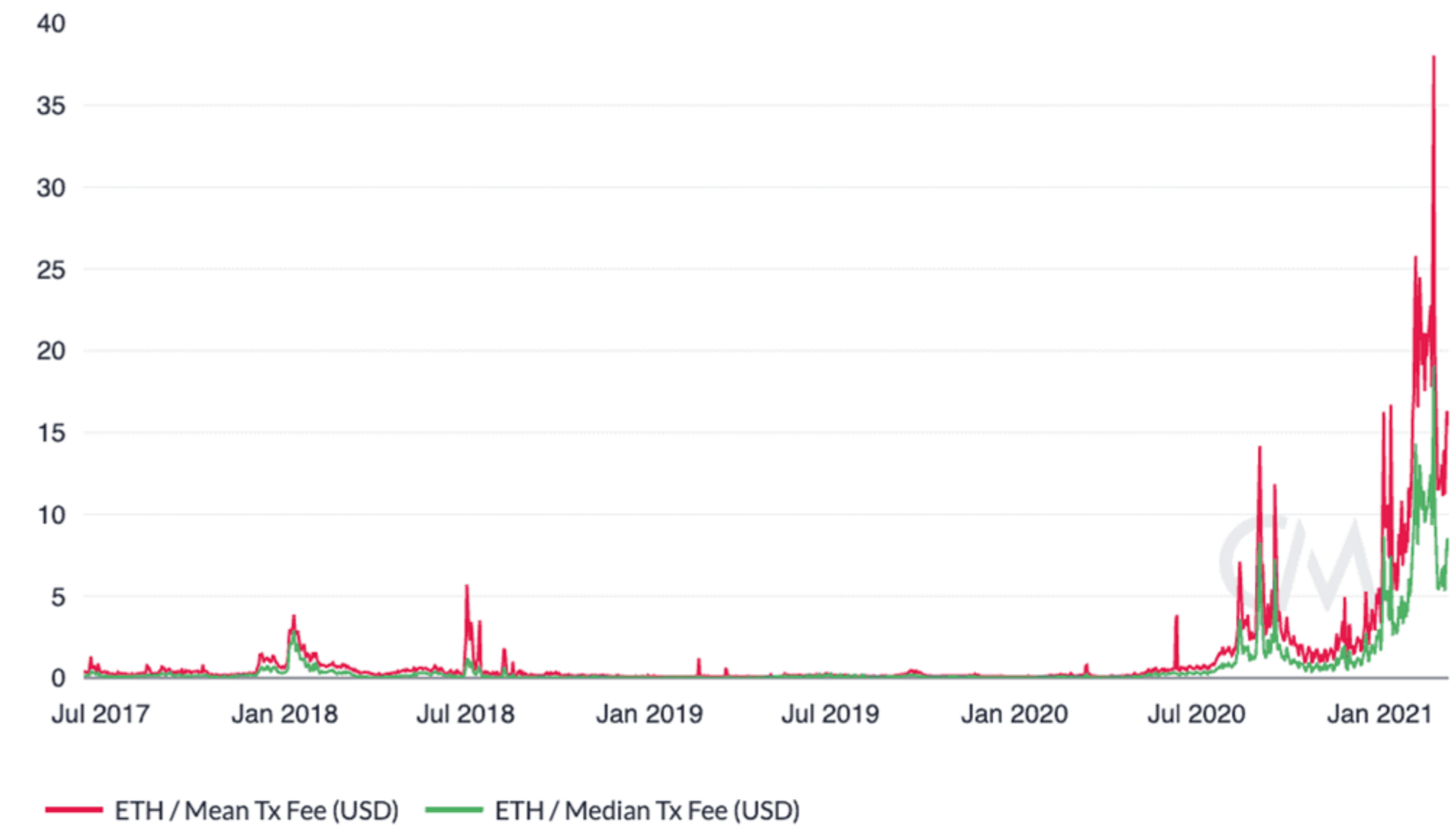**Number of Projects**

11260

* As for December 5, 2021

Bad Design Decisions → Big Consequences

Blockchain-related hack events by year (2018 - Q3 2021)
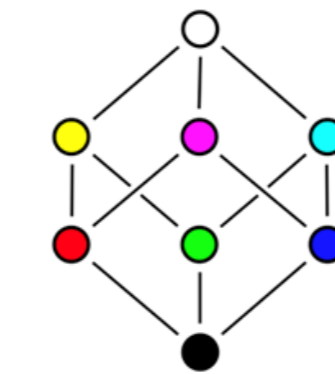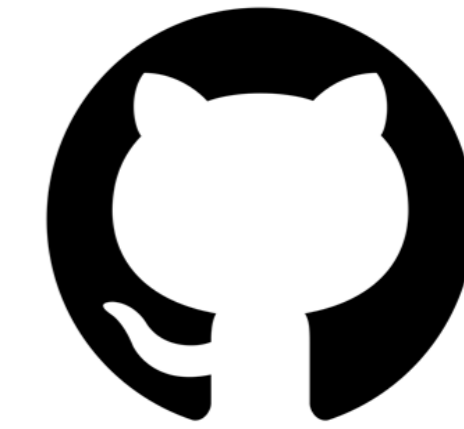

Source: hacked.slowmist.io

ETH Mean & Median Fee, USD

ETH / Mean Tx Fee (USD) ETH / Median Tx Fee (USD)

Source: COINMETRICS

4

Alternative Design?

- Better data organization
- No global consensus
- Non-masking Fault Tolerance
- No prevention but detect, recover, resolve

💡 Inspiration: Make them Accountable

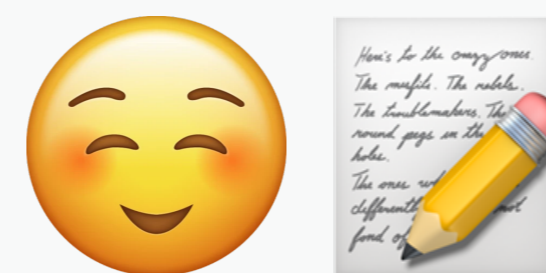# Malicious Peers and Accountability

MALICIOUS PEER

Changed Protocol P'

Deviates from the Protocol.
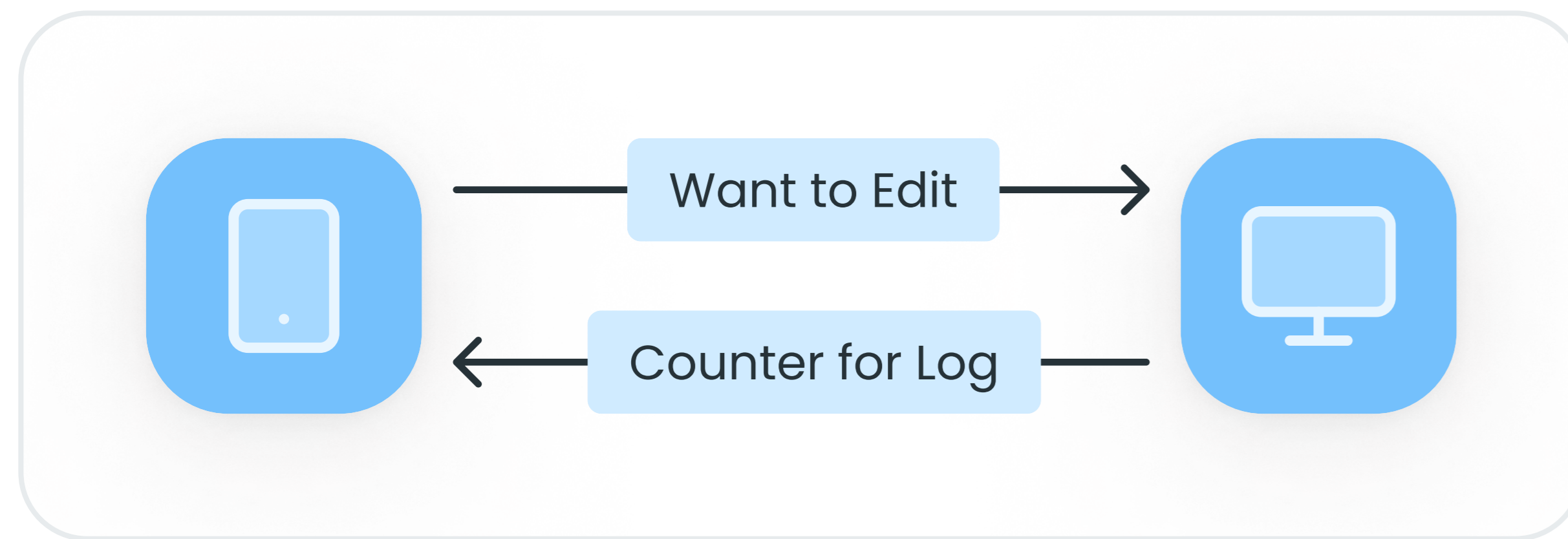Can produce any result

CORRECT PEER

Protocol P

Acts as expected

But we need to define rules!

Wikipedia Case: Data

- Nice Linear History of edits per Page
- Linear Log per User
- Vandalism, malicious actions are transparent

Wikipedia Case: Governance

Edit Distribution of All Bots

How Can We Enable Such Collaboration?

Consistency and Convergence

Accountability Goals

Eventual Delivery	Strong Convergence	Causal Consistency

Accuracy	Unanimity

Design Decisions

1.
Explicit Causality

2.
Resolving Conflicts

3.
Validity Rules

4.
Reconciliation

1. Explicit Causality

- Embed the dependencies in the message, via set of hashes
- Inverted Lamport's 'happens-before'
- Preserve Linear Log where you can

💡 Rule: Any new message "continues" pervious

= data + {id(m_2), id(m_3)}

2. Resolving Divergence

- Special operation: 'Merge'
- If you link multiple → Should merge first
- Correct peer pre-agree on the merge policies

💡 Rule: Resolve divergence with pre-agreed policy

Merge   ⟵ Depends

3. Validity Rules

- Function 'Valid' against the state or messages
- Check message against local knowledge, no coordination required
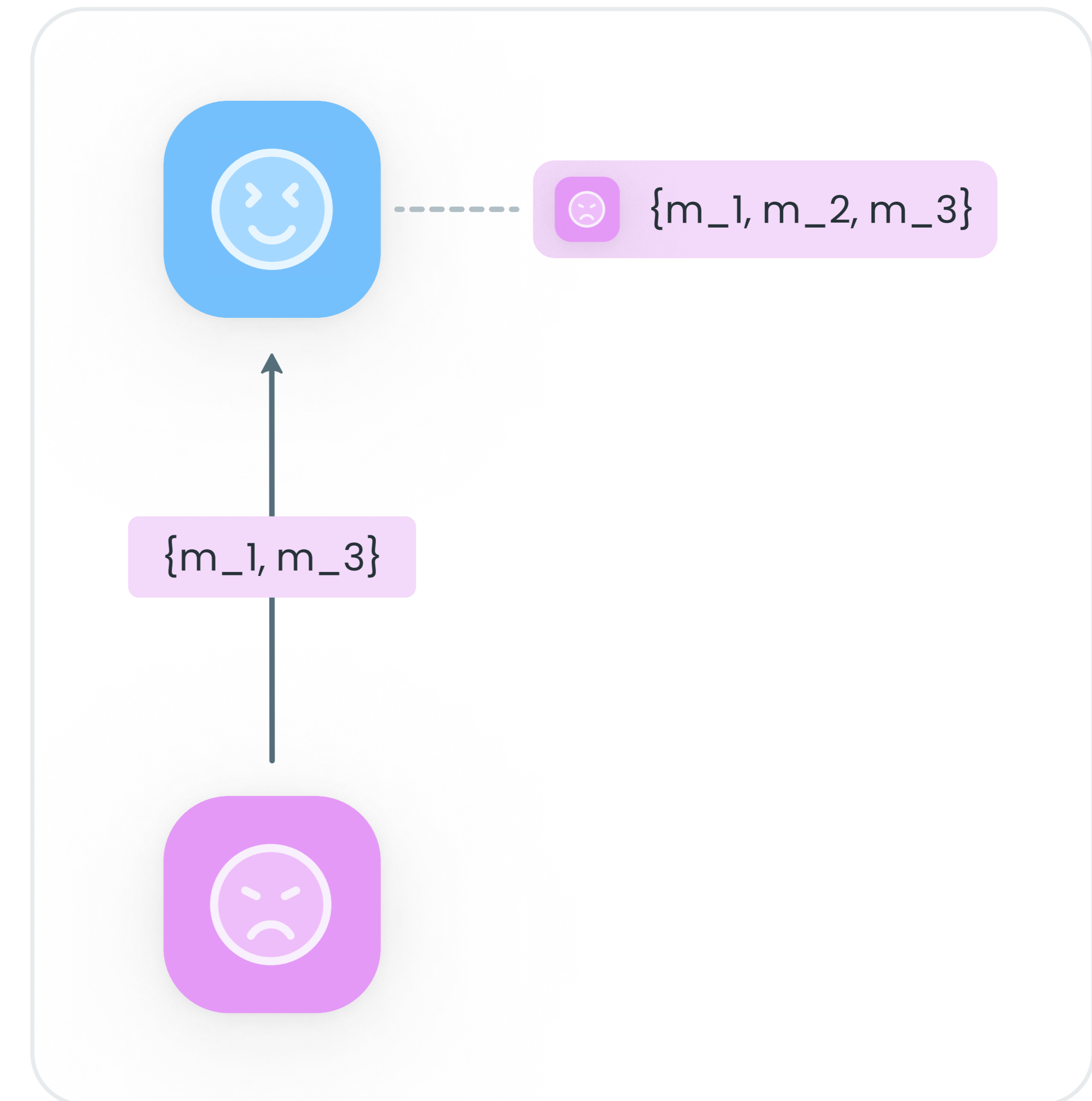- Examples: no double spend, no slur

💡 Rule: Never break pre-defined validity rules

4. Message Set Reconciliation

- 1996: Anti-Entropy
- 2011: Set reconciliation with Bloom Filters
- New: Compare against known last known set and check if node is hiding something

💡 Rule: Report all messages

$\{m\_1, m\_2, m\_3\}$

$\{m\_1, m\_3\}$

Accountable Protocols - Strong Foundation

- Expose peers that violate the rules
- Suspect peers that ignore requests
- Periodically gossip status to others

As a Result

- All observable malicious are exposed
- You only need 2 correct nodes to function

Q&A

✉ b.nasrulin@tudelft.nl