TrInc: Small Trusted Hardware for Large Distributed Systems

Dave Levin — University of Maryland

John R. Douceur
Jacob R. Lorch — Microsoft Research
Thomas Moscibroda

Trust in distributed systems

Selfish Participants

Malicious Participants

Trust in distributed systems

Selfish Participants

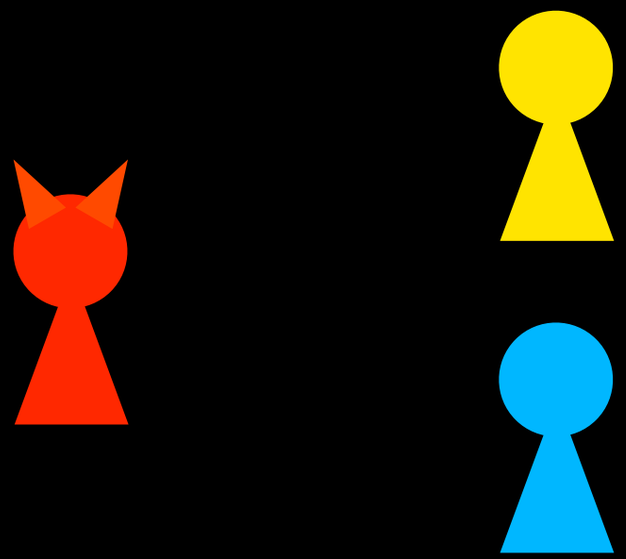
Malicious Participants

Powerful tool: Equivocation

A participant "equivocates"
by sending conflicting messages to others

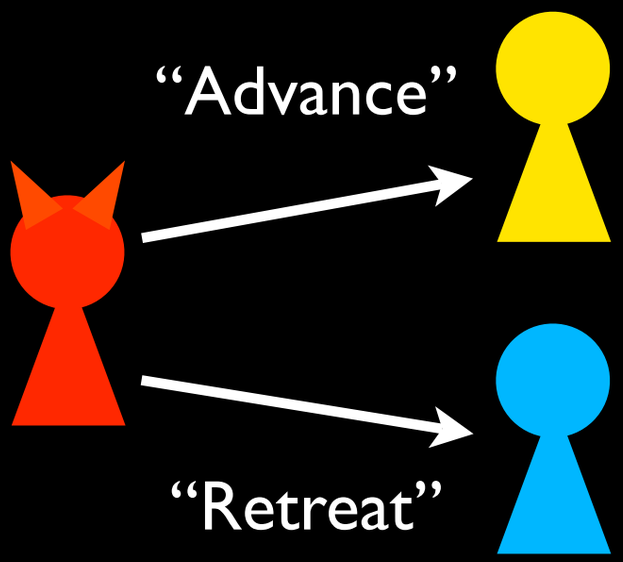
Equivocation is common and powerful

Byz. Generals



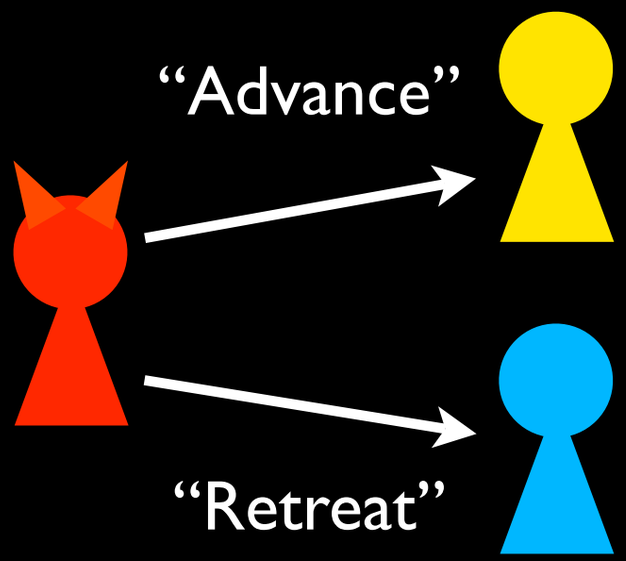
Equivocation is common and powerful

Byz. Generals

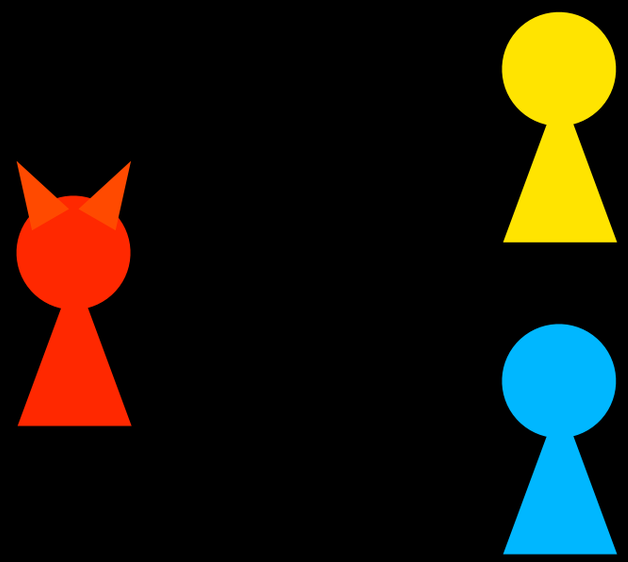


Equivocation is common and powerful

Byz. Generals

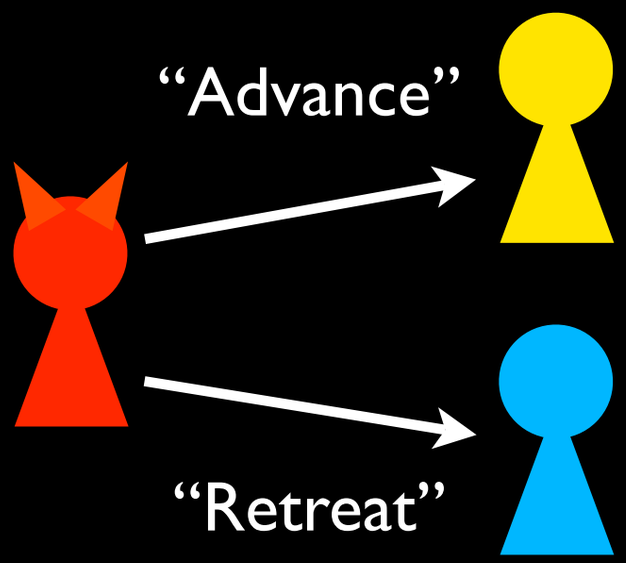


Voting

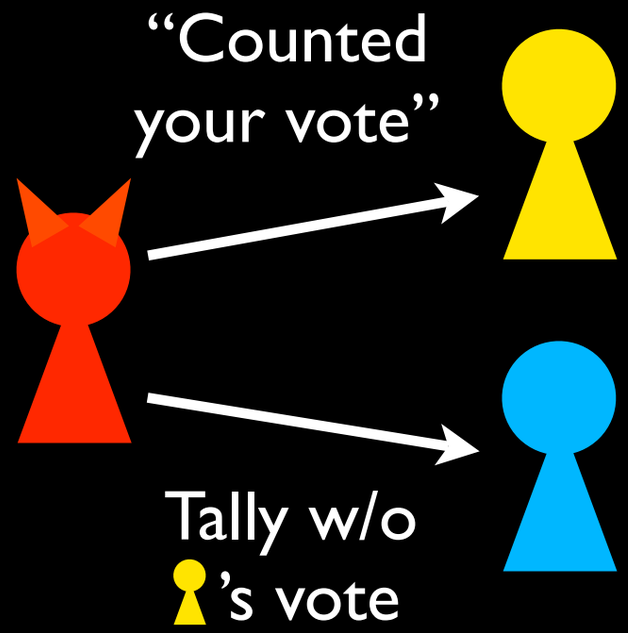


Equivocation is common and powerful

Byz. Generals

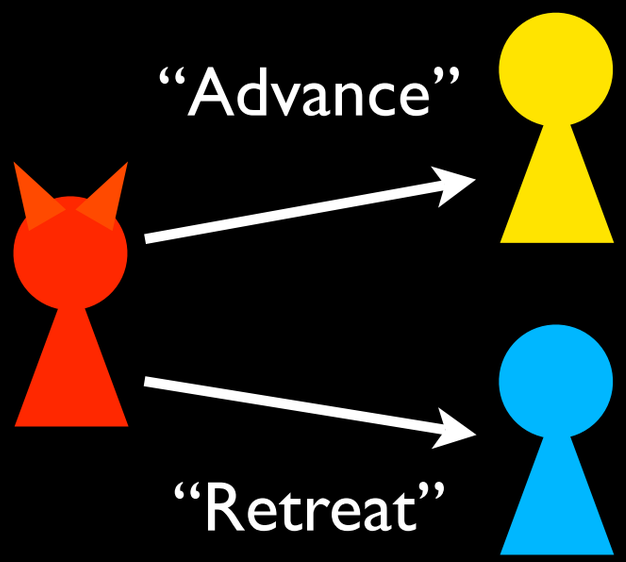


Voting

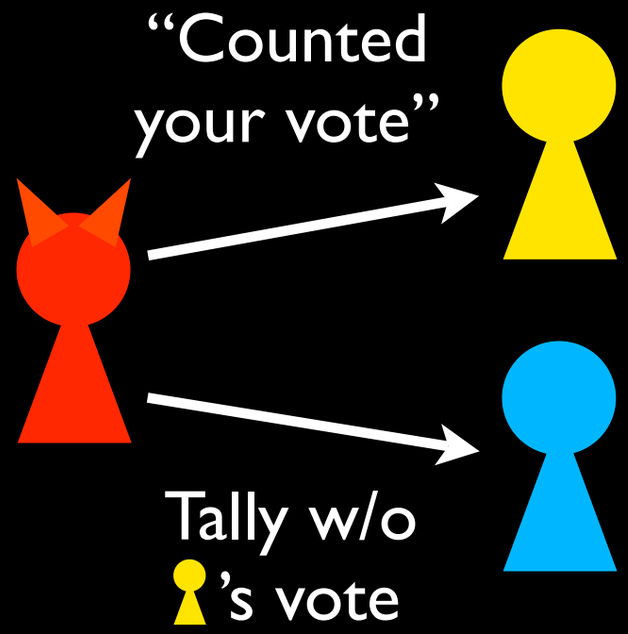


Equivocation is common and powerful

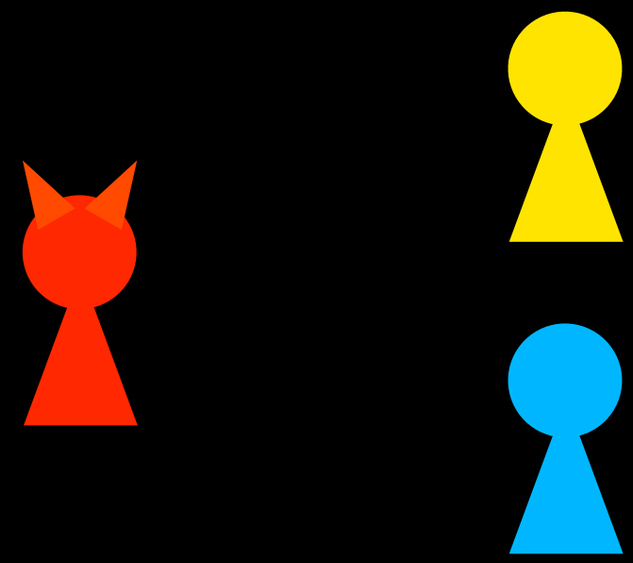
Byz. Generals



Voting

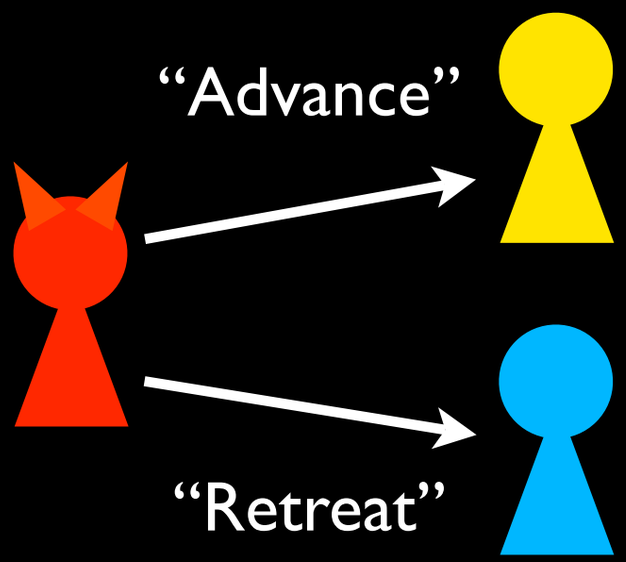


BitTorrent

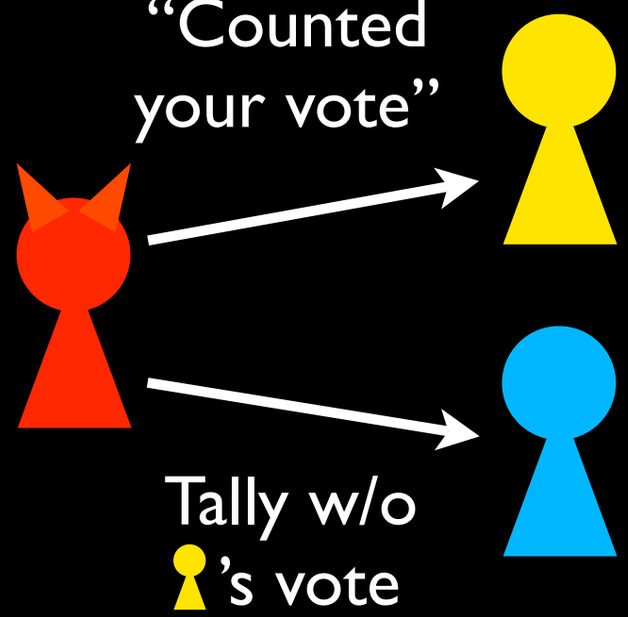


Equivocation is common and powerful

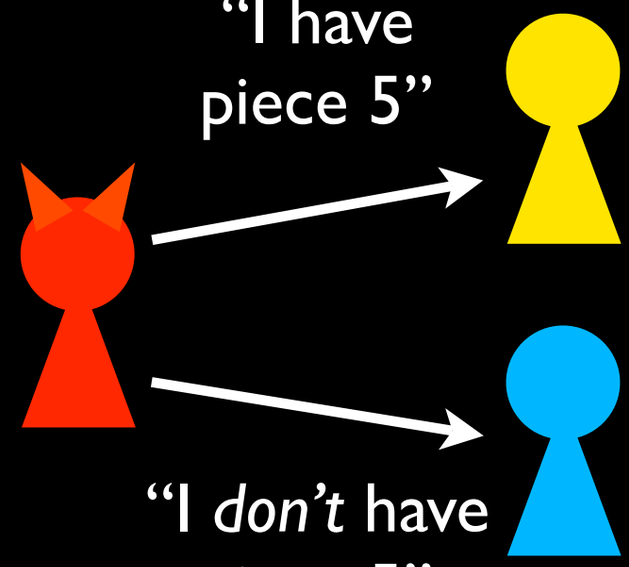
Byz. Generals



Voting

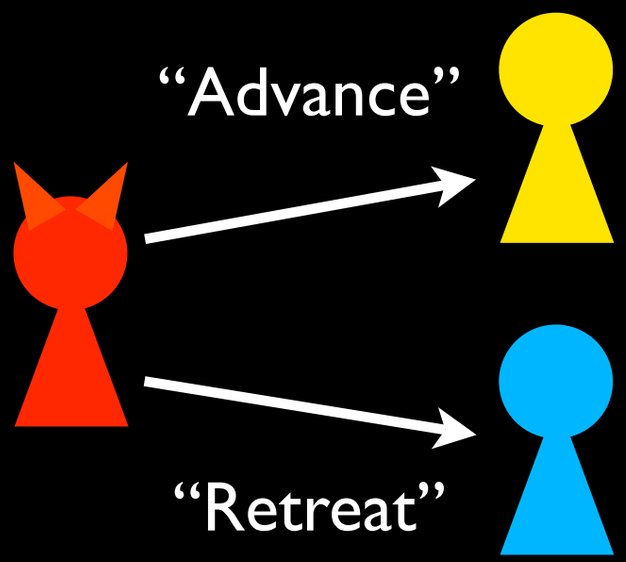


BitTorrent

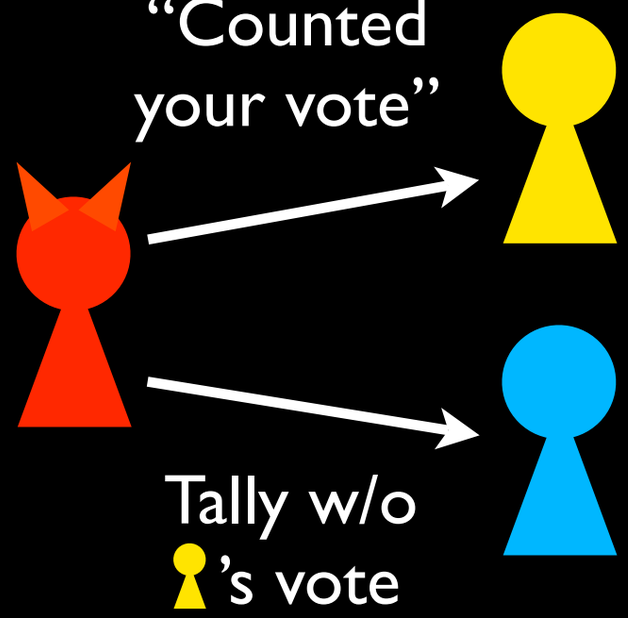


Equivocation is common and powerful

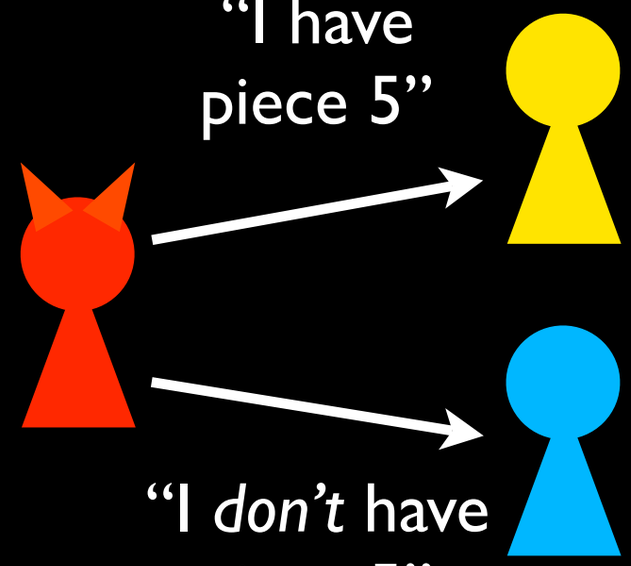
Byz. Generals



Voting



BitTorrent



Leader election

Trusted logs

soBGP

Online games

Digital cash

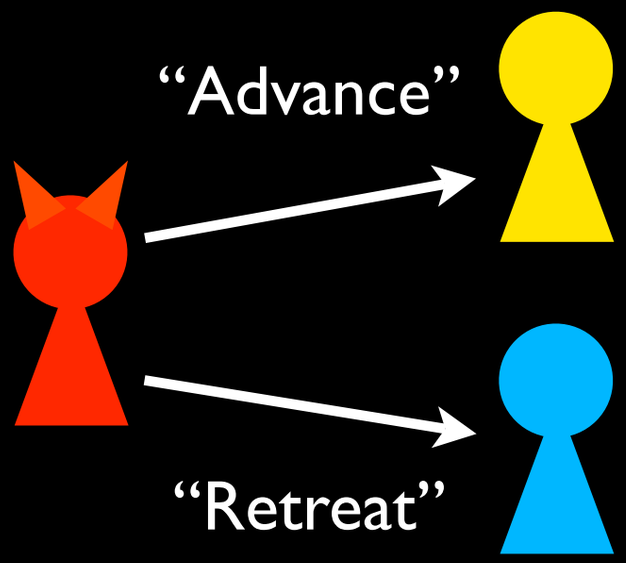
Version control

Auctions

DHTs

Equivocation is common and powerful

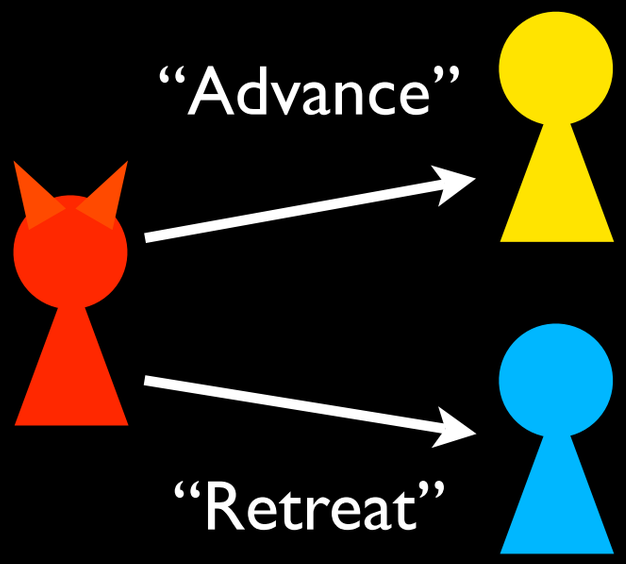
Byz. Generals



- f malicious users
- If completely untrusted, 3f+1 users needed for consensus [Lamport et al, 1982]

Equivocation is common and powerful

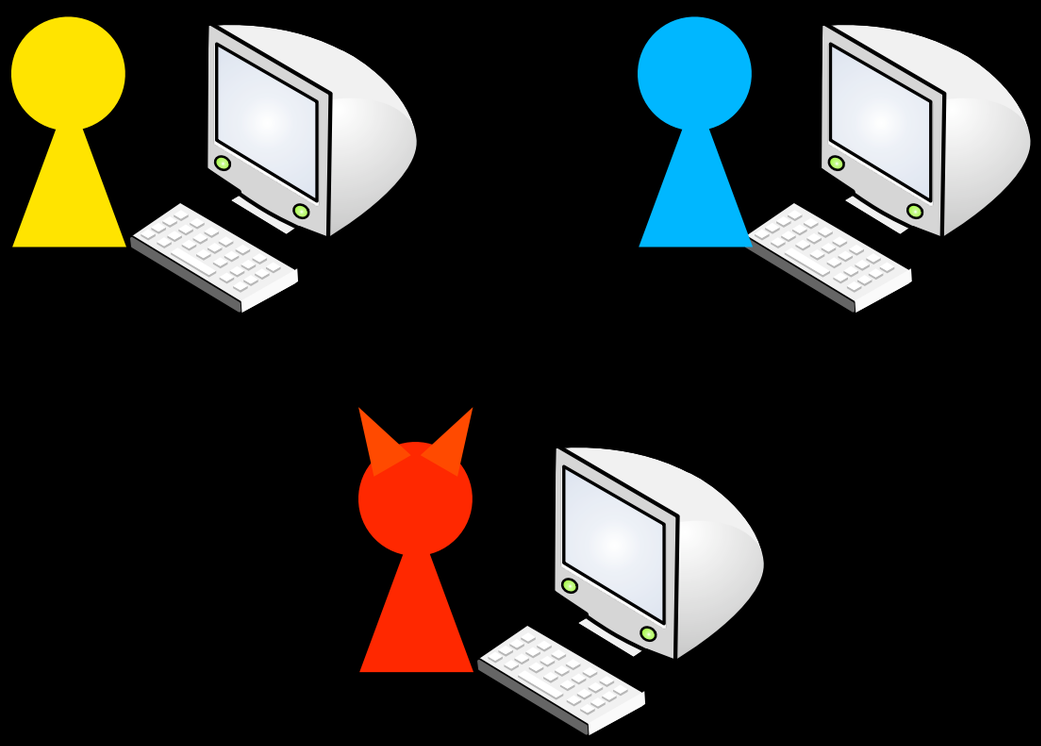
Byz. Generals



- f malicious users
- If completely untrusted, 3f+1 users needed for consensus [Lamport et al, 1982]
- If users cannot equivocate, only 2f+1 users are needed [Chun et al, 2007]

Enter Trusted Hardware

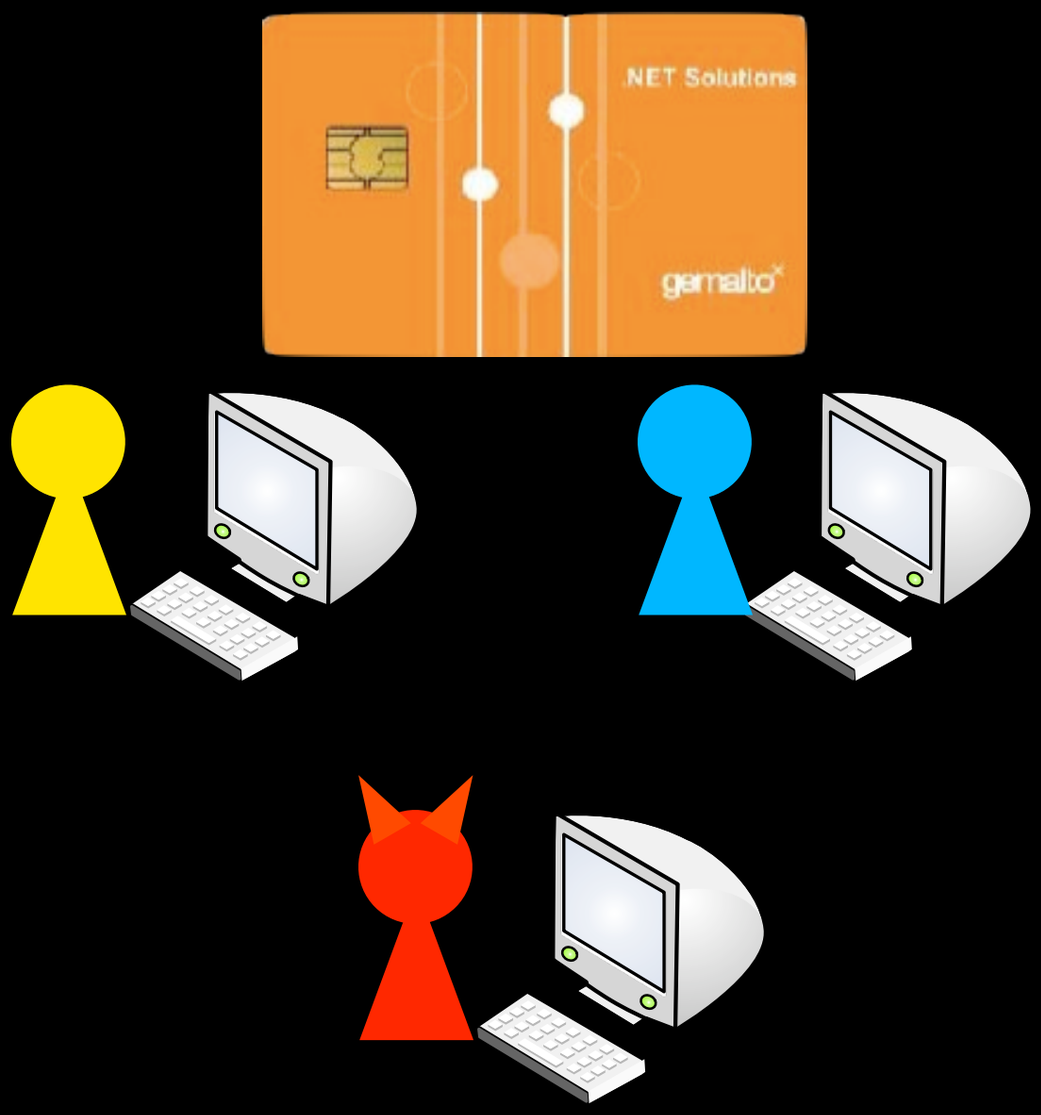
Equivocation can be rendered impossible with trusted hardware



- New design space
- All participants have a trusted component

Enter Trusted Hardware

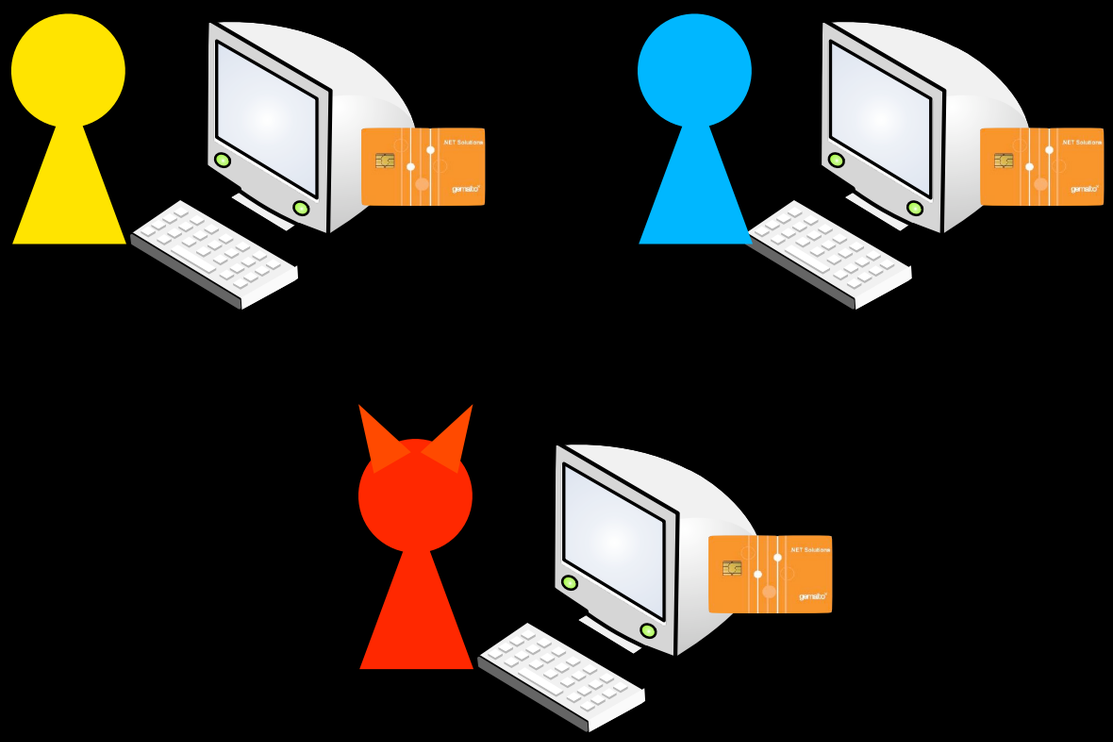
Equivocation can be rendered impossible with trusted hardware



- New design space
  - All participants have a trusted component

Enter Trusted Hardware

Equivocation can be rendered impossible with trusted hardware



- New design space
- All participants have a trusted component

Enter Trusted Hardware

Equivocation can be rendered impossible with trusted hardware

- New design space
  - All participants have a trusted component
- To be practical, the hardware must be small
  - Ubiquity via low cost
  - Tamper-resilient
  - Easier to verify a small TCB

Contributions

1. TrInc – A new, practical primitive for eliminating equivocation

2. Applications of TrInc

3. Implementation in currently available hardware

Contributions

1. TrInc – A new, practical primitive for eliminating equivocation

2. Applications of TrInc

3. Implementation in currently available hardware

Motivating question

What is the minimal abstraction needed
to make equivocation impossible?

Motivating question

What is the minimal abstraction needed to make equivocation impossible?

A counter and a key are enough

TrInc: Trusted Incrementer

1. Monotonically increasing counter
2. Key for signing attestations

TrInc: Trusted Incrementer

1. Monotonically increasing counter
2. Key for signing attestations

Attestations bind data to counters

TrInc: Trusted Incrementer

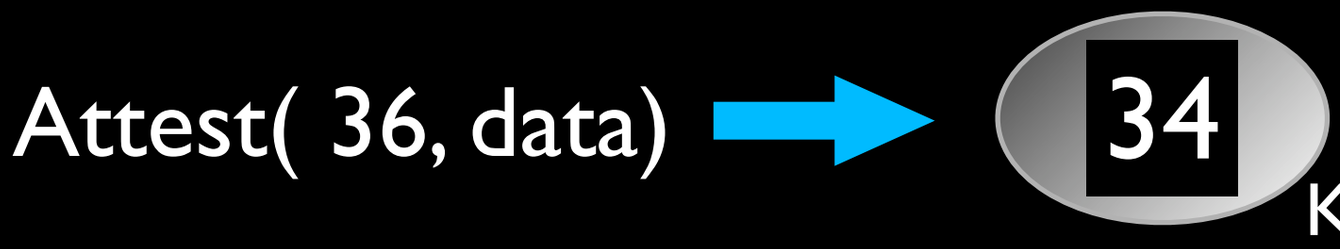
1. Monotonically increasing counter
2. Key for signing attestations

Attestations bind data to counters

"Bind this data to counter value 36"

TrInc: Trusted Incrementer

1. Monotonically increasing counter
2. Key for signing attestations

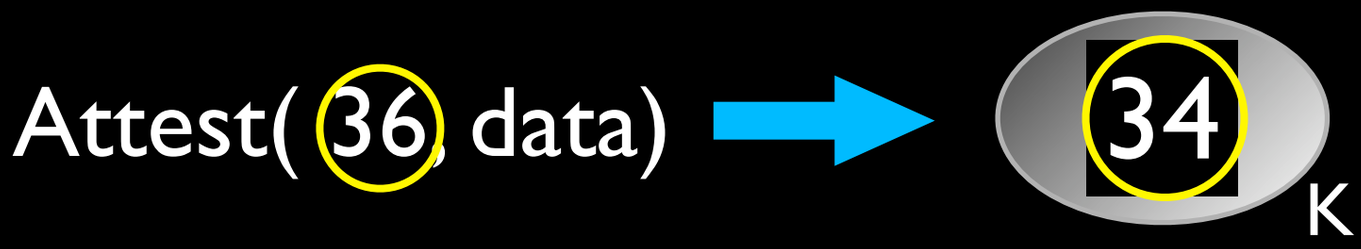


Attestations bind data to counters

"Bind this data to counter value 36"

TrInc: Trusted Incrementer

1. Monotonically increasing counter
2. Key for signing attestations

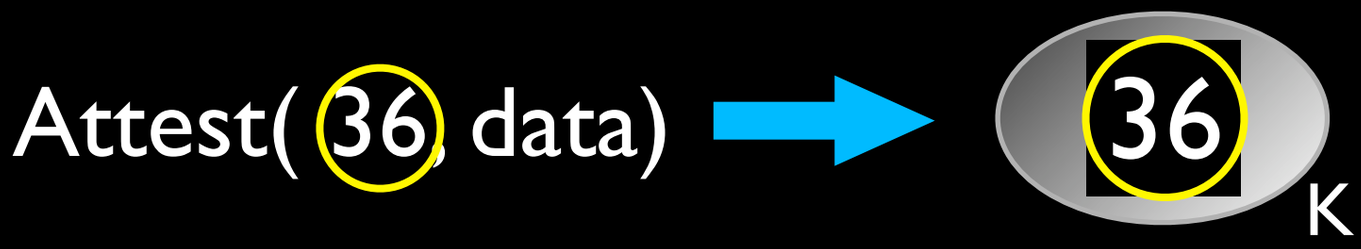


Attestations bind data to counters

"Bind this data to counter value 36"

TrInc: Trusted Incrementer

1. Monotonically increasing counter
2. Key for signing attestations

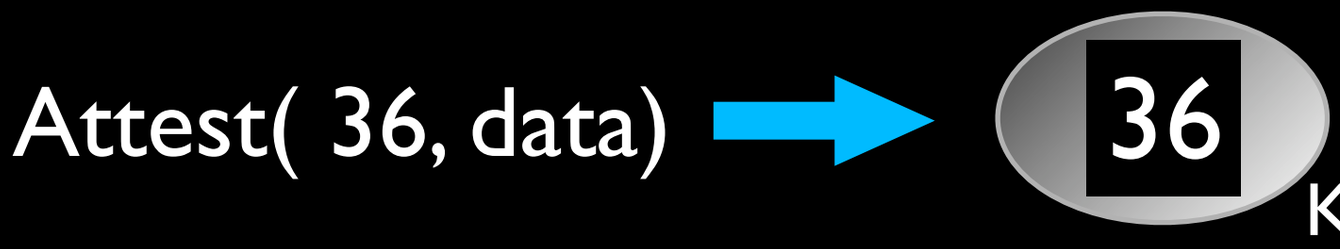


Attestations bind data to counters

"Bind this data to counter value 36"

TrInc: Trusted Incrementer

1. Monotonically increasing counter
2. Key for signing attestations

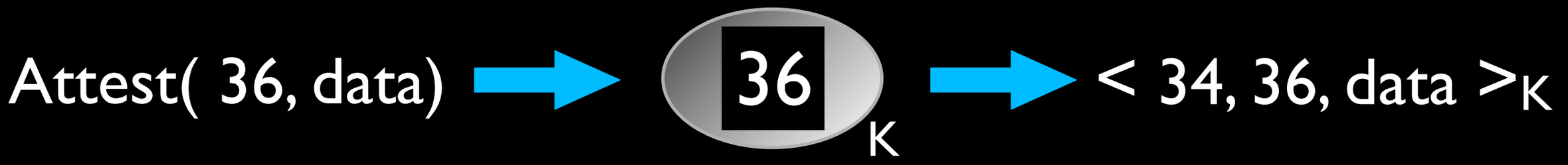


Attestations bind data to counters

"Bind this data to counter value 36"

TrInc: Trusted Incrementer

1. Monotonically increasing counter
2. Key for signing attestations



Attestations bind data to counters

"Bind this data to counter value 36"

TrInc Attestations

$< 34, 36, \texttt{data} >_K$

$< 36, 36, \text{nonce} >_K$

TrInc Attestations

Advance attestation

$< 34, 36, data >_K$

- Can only move to a state once
- "*data*" is forever bound to 36
- There was nothing bound to 35

Status attestation

$< 36, 36, nonce >_K$

- "What is your current counter?"
  - Nonces assure freshness
- There is nothing beyond 36 (yet)

Multiple counters

- Need multiple trusted counters
  - Systems running concurrently
  - Some systems benefit from more counters

Multiple counters

- Need multiple trusted counters
  - Systems running concurrently
  - Some systems benefit from more counters

Trinket

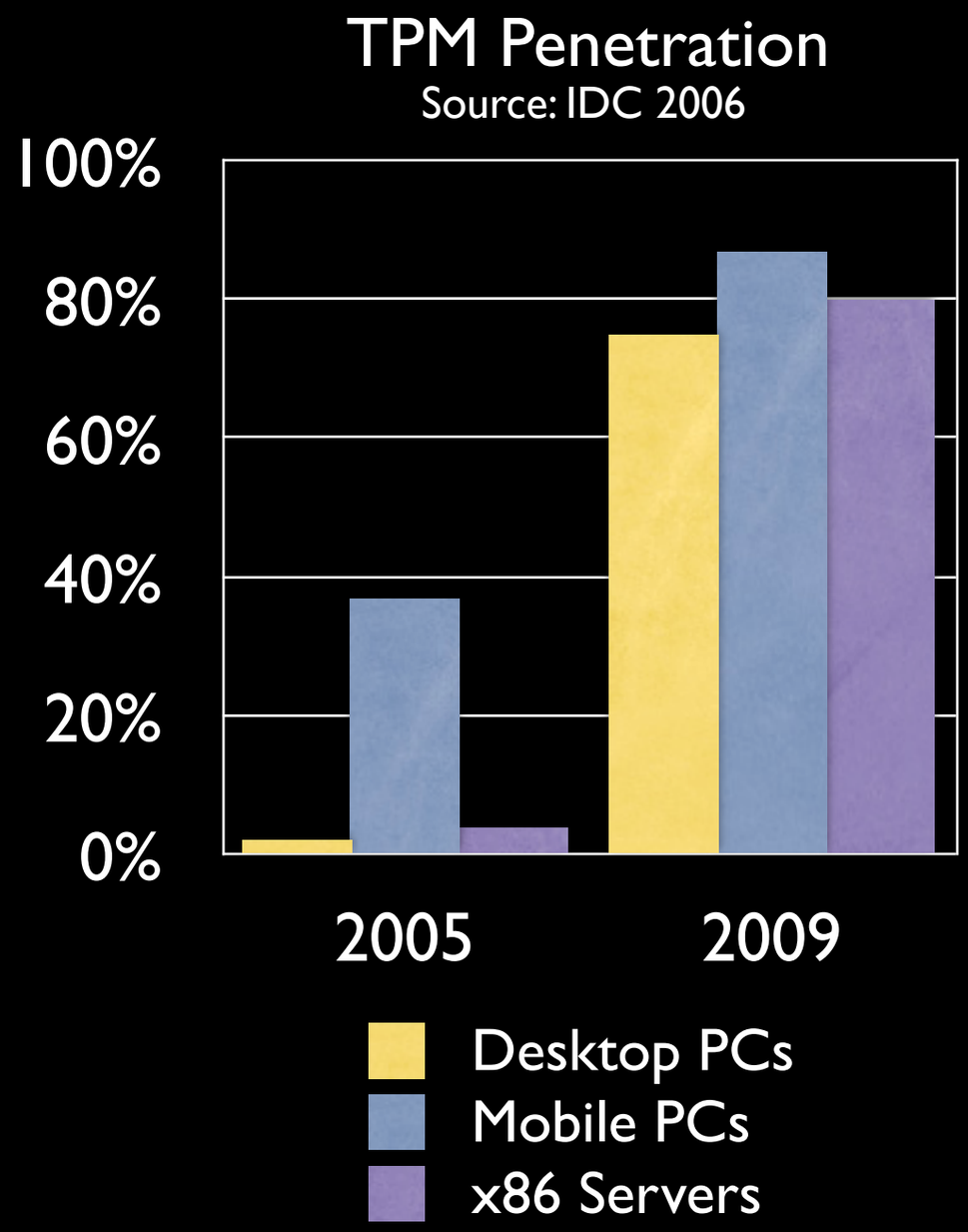
- Hardware that contains ≥ 1 counter is a Trinket
  - Allocates and frees counters
  - Establishes session keys

TrInc is practical

- Trusted Platform Module (TPM) is ubiquitous

- Has what we need
  - Tamper-resistance
  - Counters (currently 4)
  - Crypto
  - Small amount of storage

- It just lacks the right interface



Contributions

1. TrInc – A new, practical primitive for eliminating equivocation

2. Applications of TrInc

3. Implementation in currently available hardware

Contributions

1. TrInc – A new, practical primitive for eliminating equivocation

2. Applications of TrInc

3. Implementation in currently available hardware

What can TrInc do?

- Trusted append-only logs
- Prevent under-reporting in BitTorrent
- Reduces communication in PeerReview
- BFT with fewer nodes and messages
- Ensure fresh data in DHTs
- Prevent Sybil attacks

What can TrInc do?

- Trusted append-only logs
- Prevent under-reporting in BitTorrent
- Reduces communication in PeerReview
- BFT with fewer nodes and messages
- Ensure fresh data in DHTs
- Prevent Sybil attacks

What can TrInc do?

- Trusted append-only logs
- Prevent under-reporting in BitTorrent
- Reduces communication in PeerReview
- BFT with fewer nodes and messages
- Ensure fresh data in DHTs
- Prevent Sybil attacks

Implementing a trusted log in TrInc

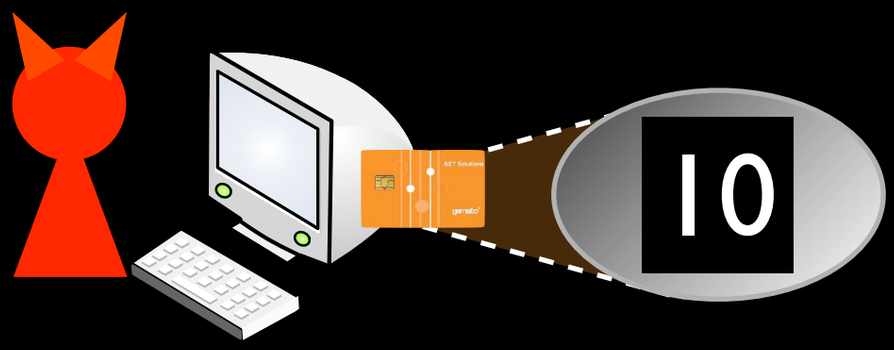
Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

Implementing a trusted log in TrInc



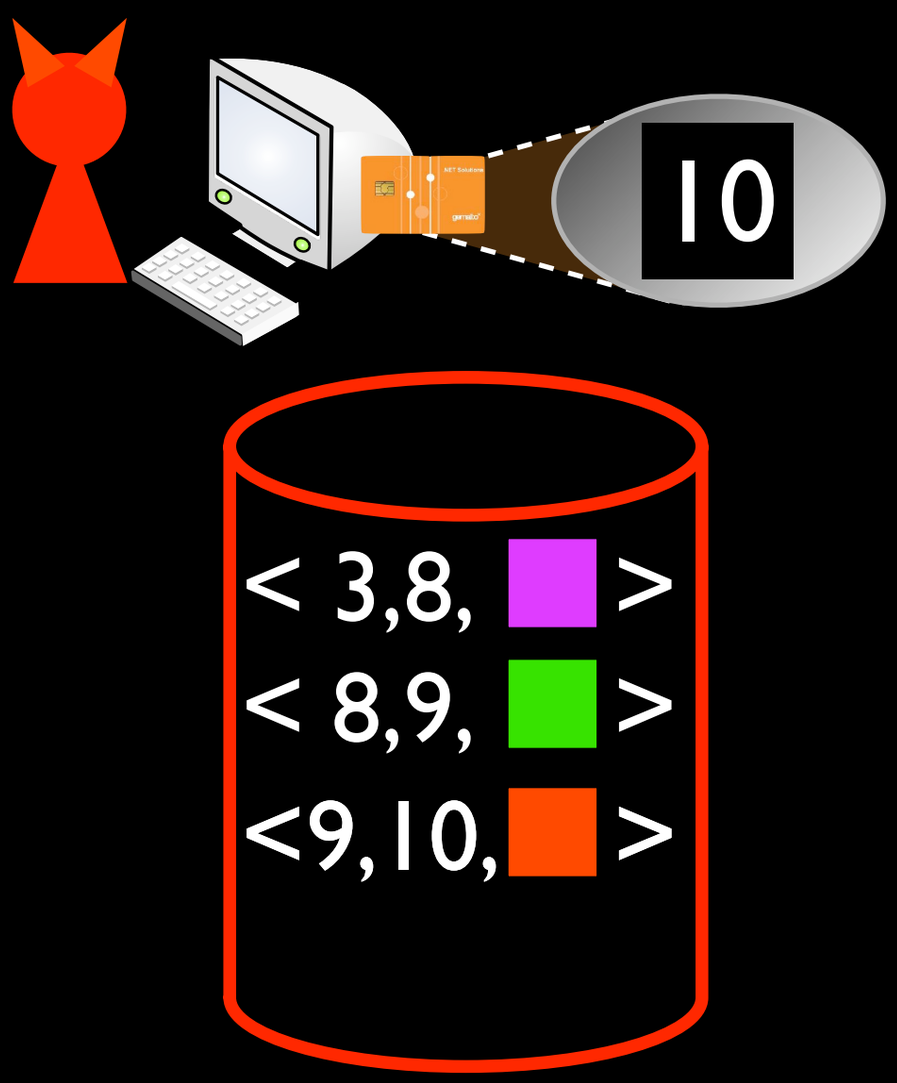
Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

Implementing a trusted log in TrInc



Untrusted storage

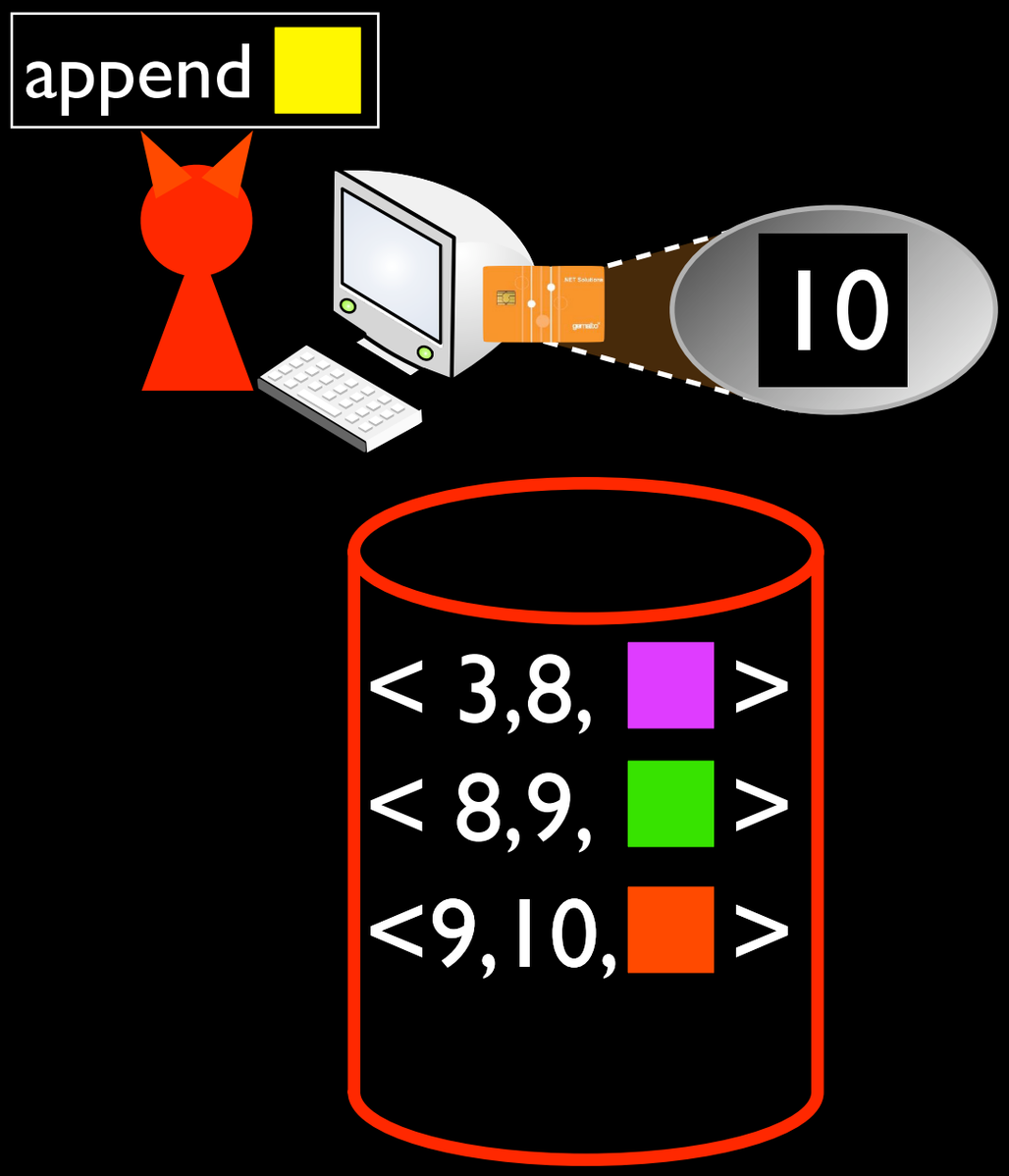
Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

Implementing a trusted log in TrInc



Untrusted storage

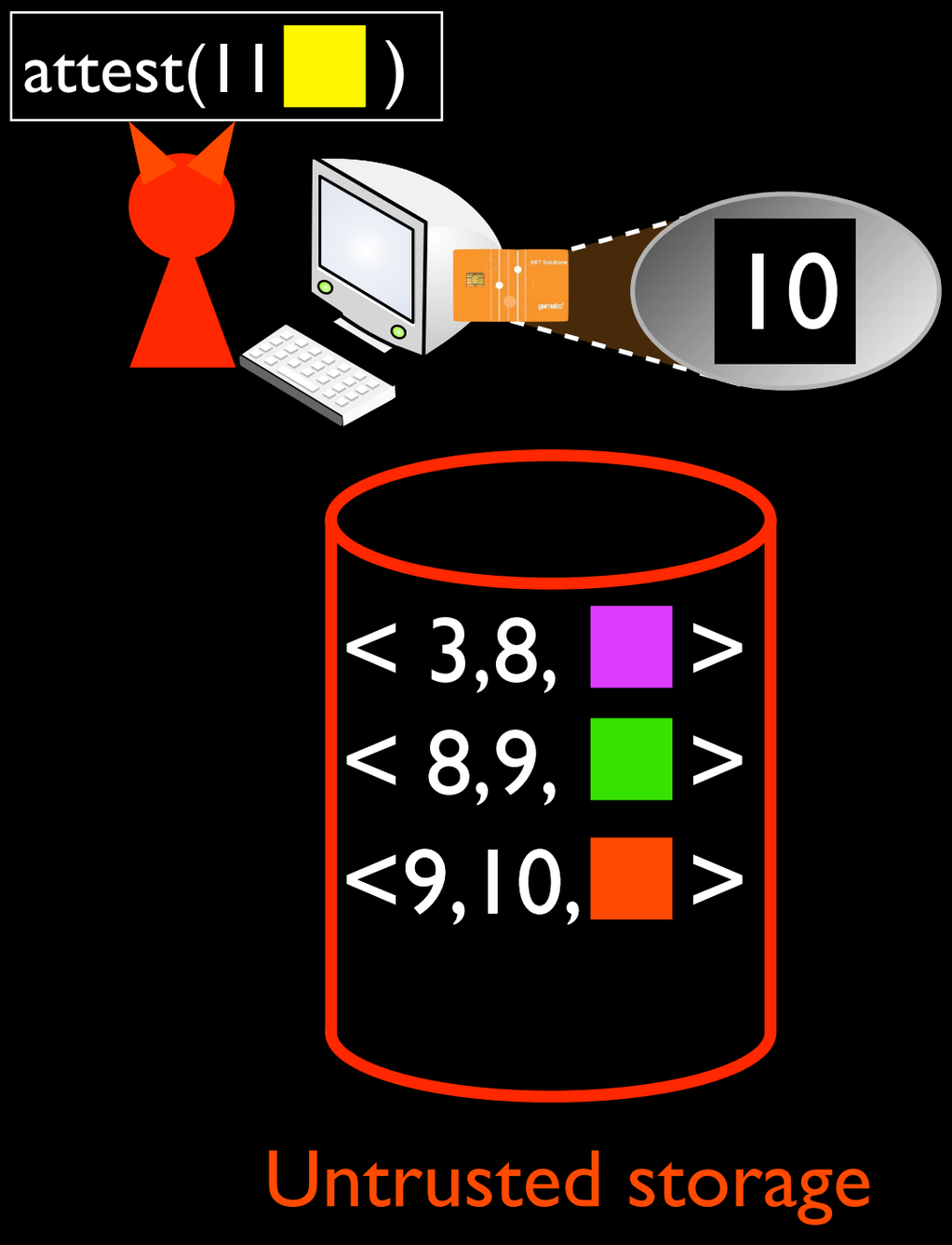
Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

Implementing a trusted log in TrInc



Untrusted storage

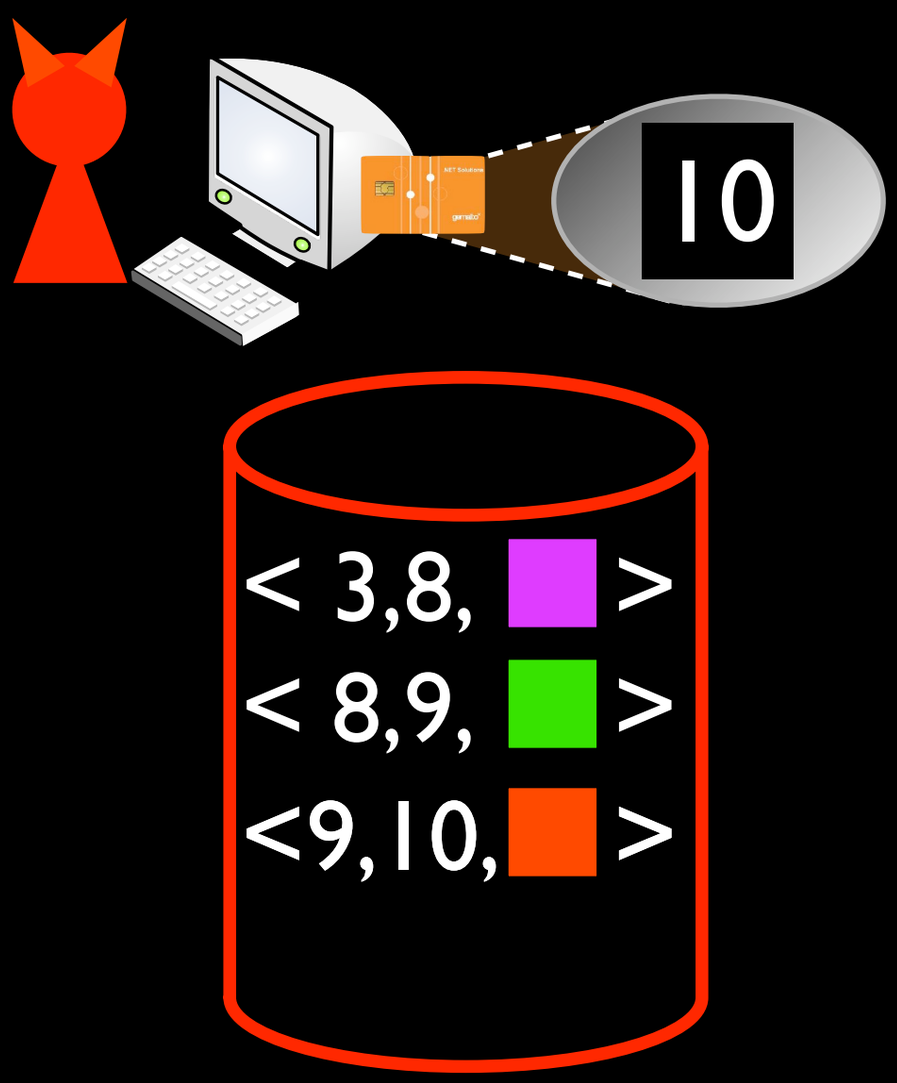
Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

Implementing a trusted log in TrInc



Untrusted storage

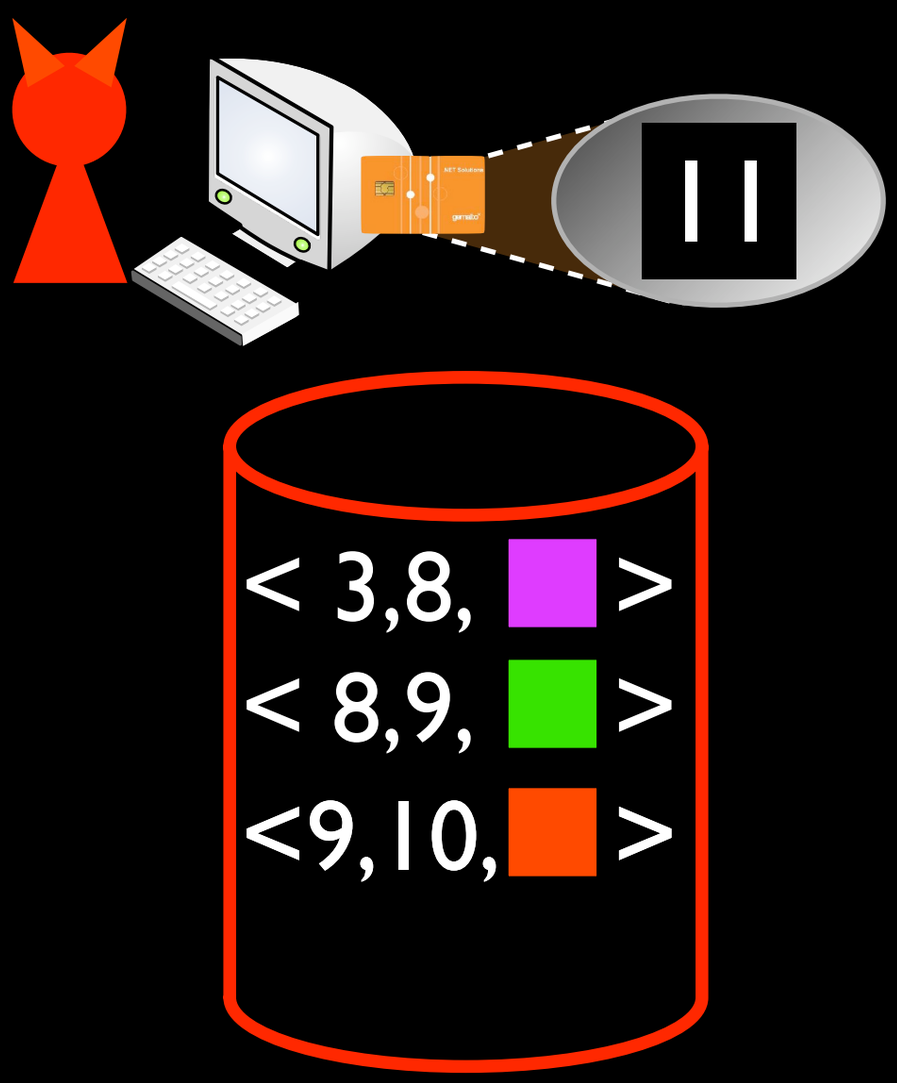
Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

Implementing a trusted log in TrInc



Untrusted storage

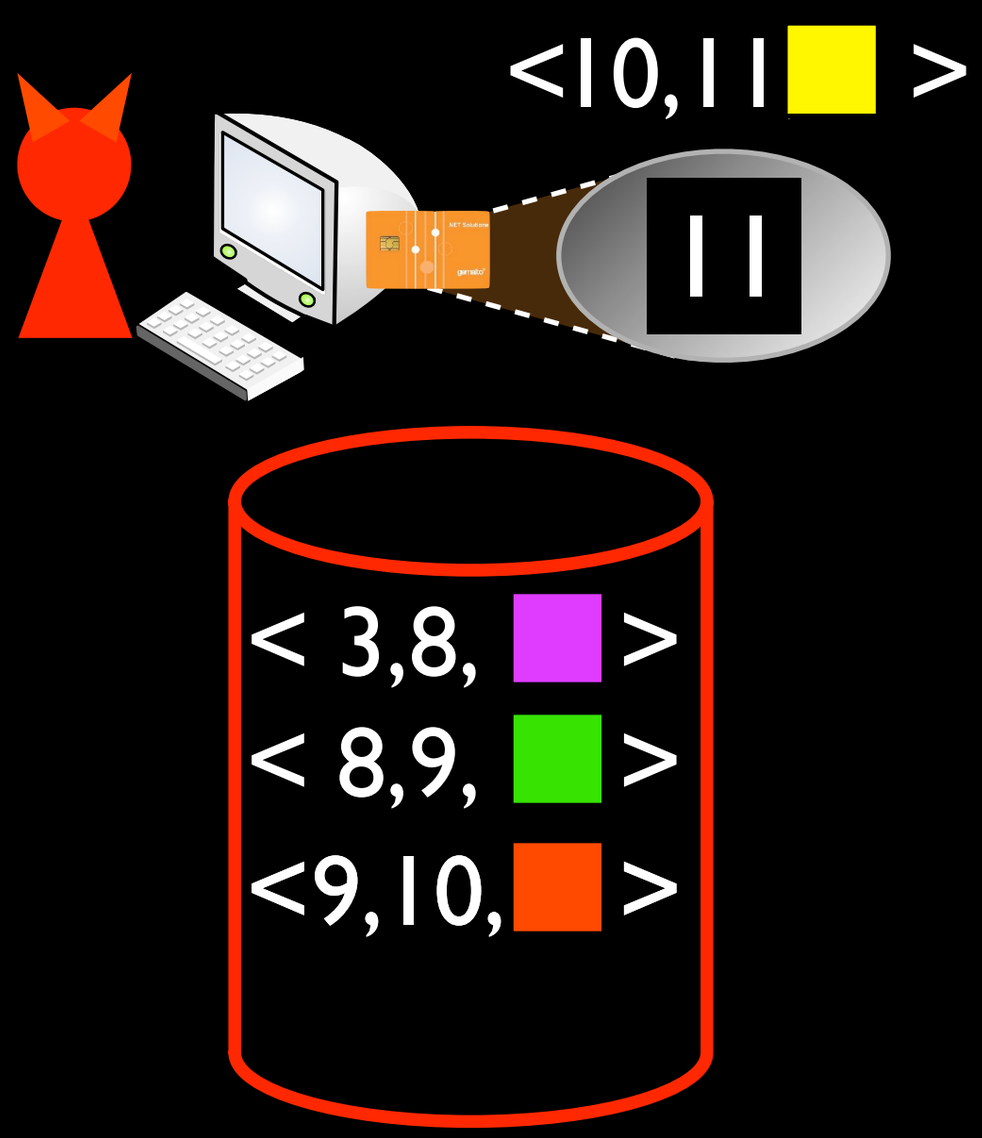
Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

Implementing a trusted log in TrInc



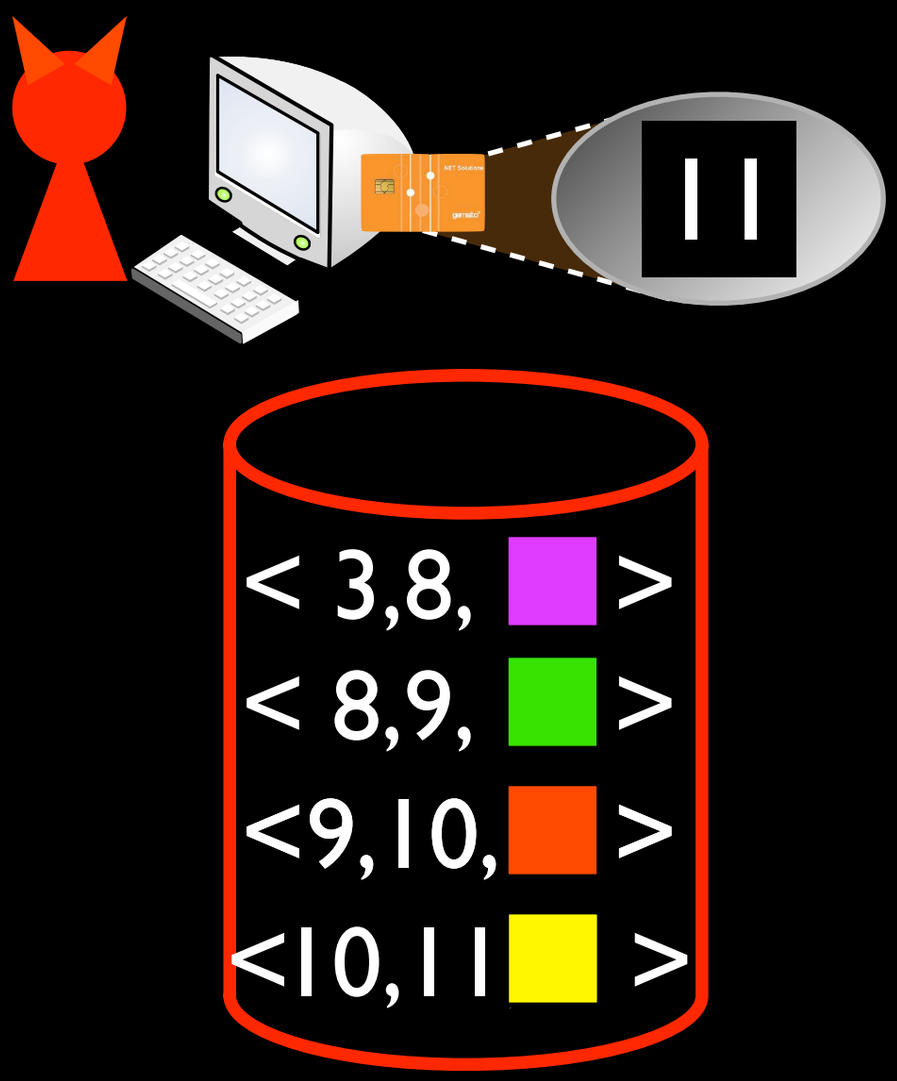
Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

Implementing a trusted log in TrInc



Untrusted storage

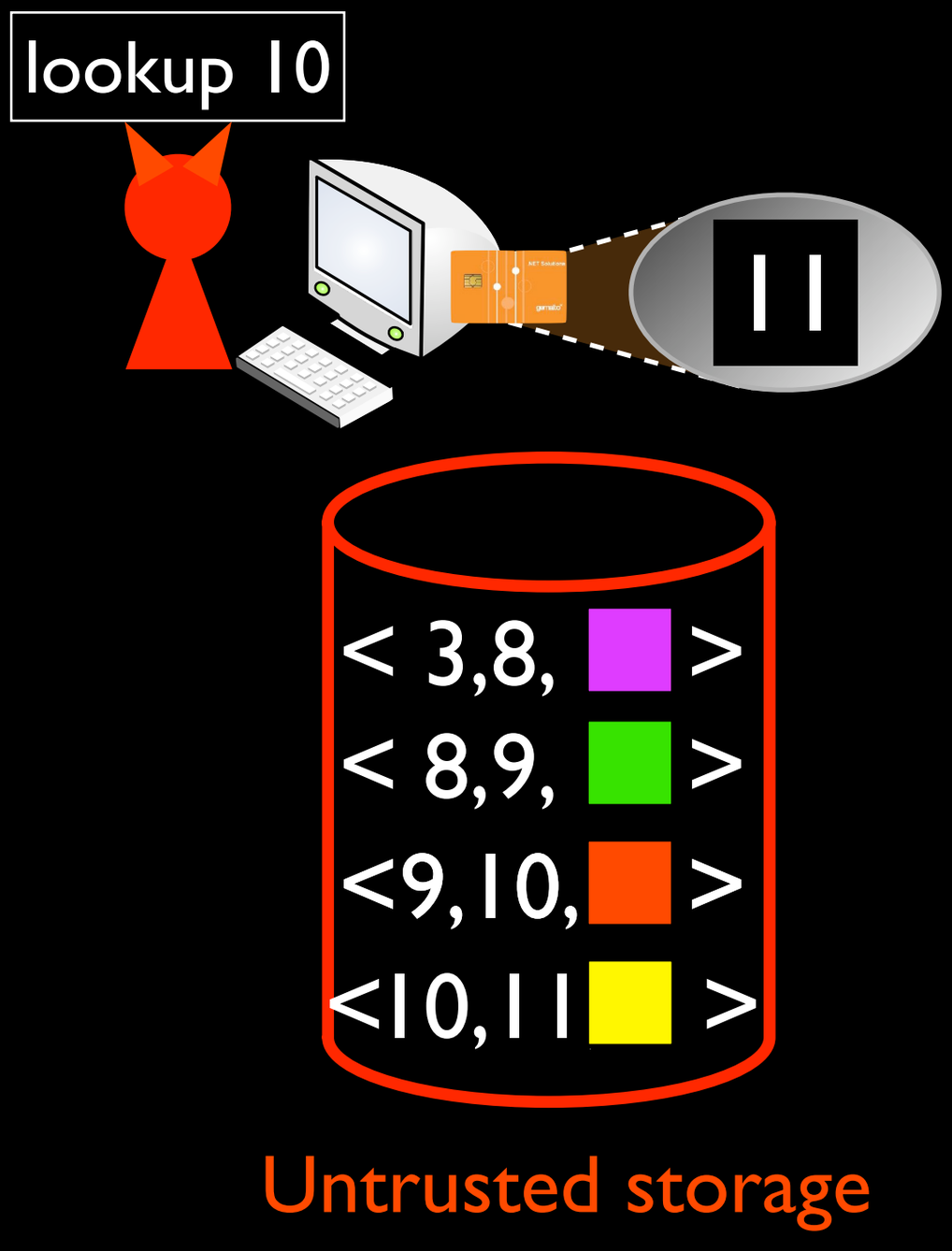
Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

Implementing a trusted log in TrInc



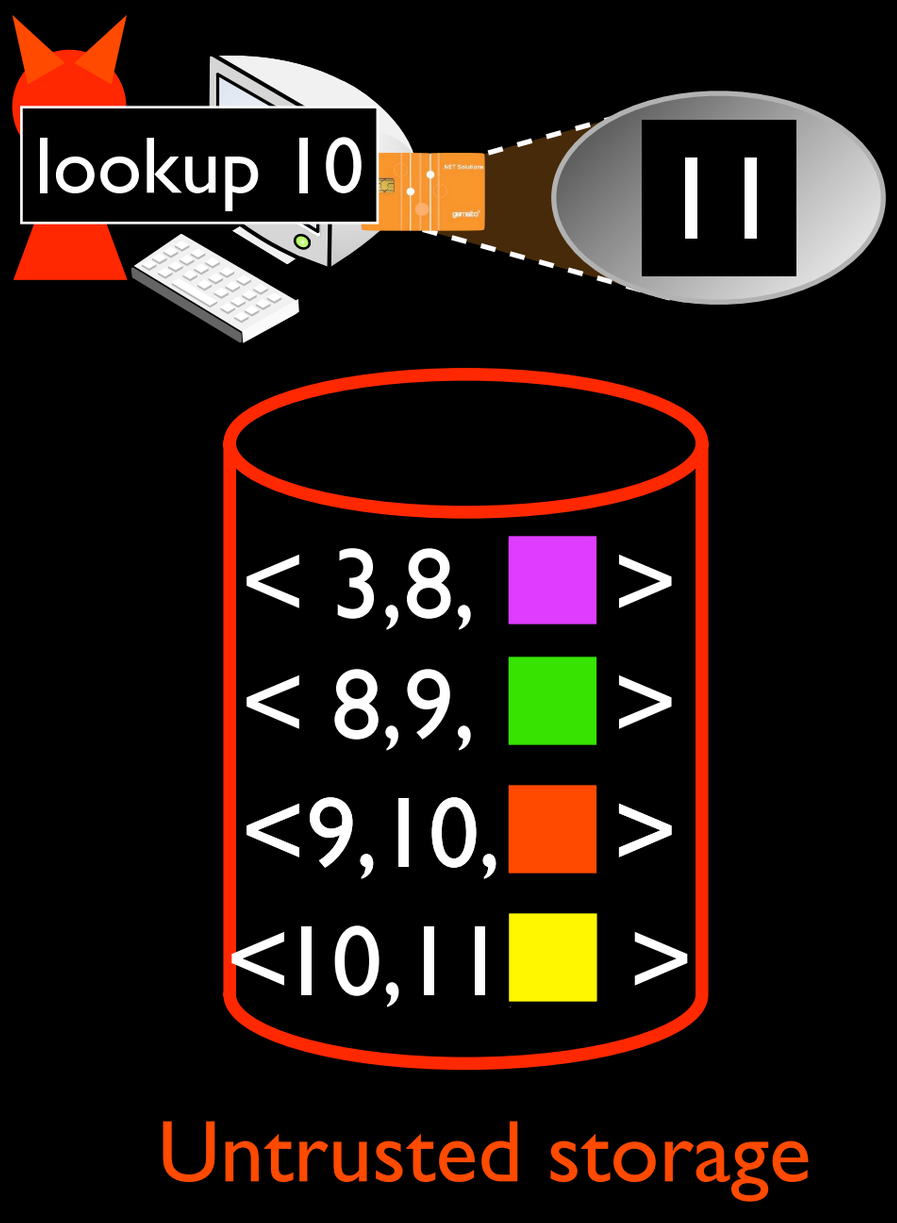
Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

Implementing a trusted log in TrInc



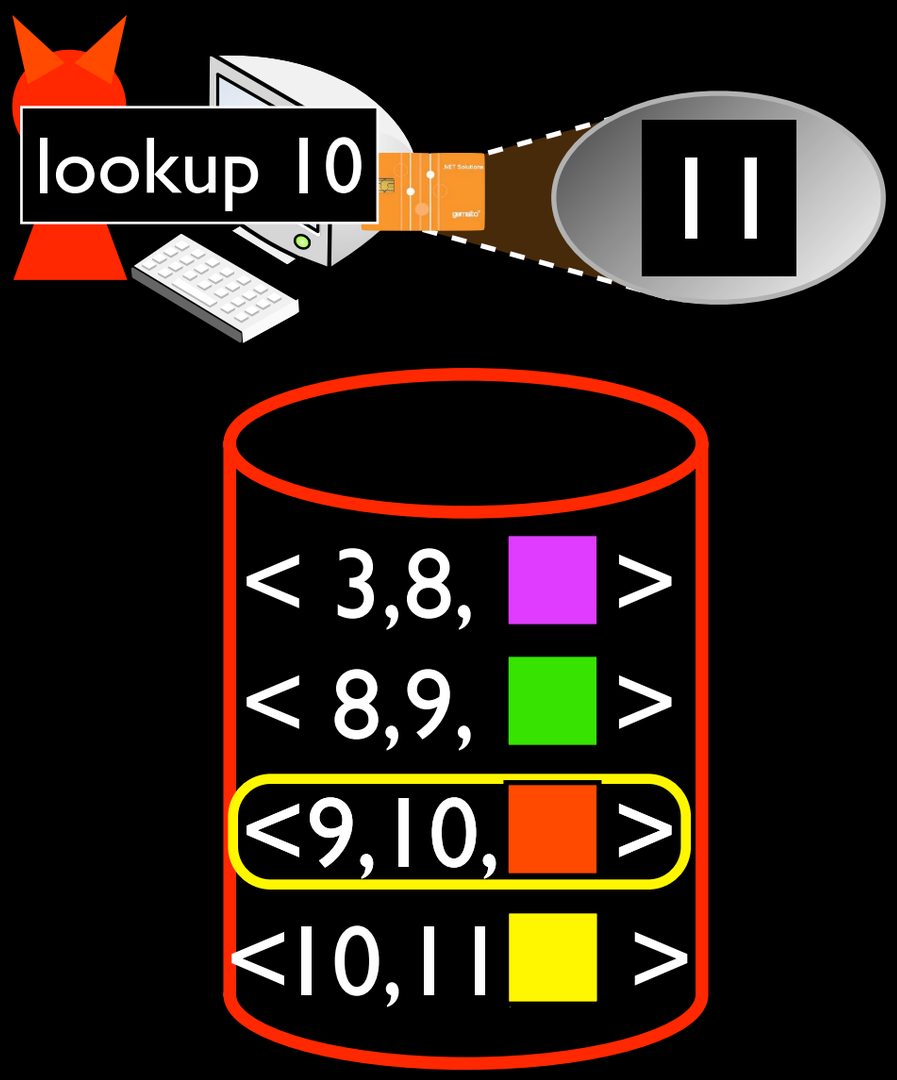
Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

Implementing a trusted log in TrInc



Untrusted storage

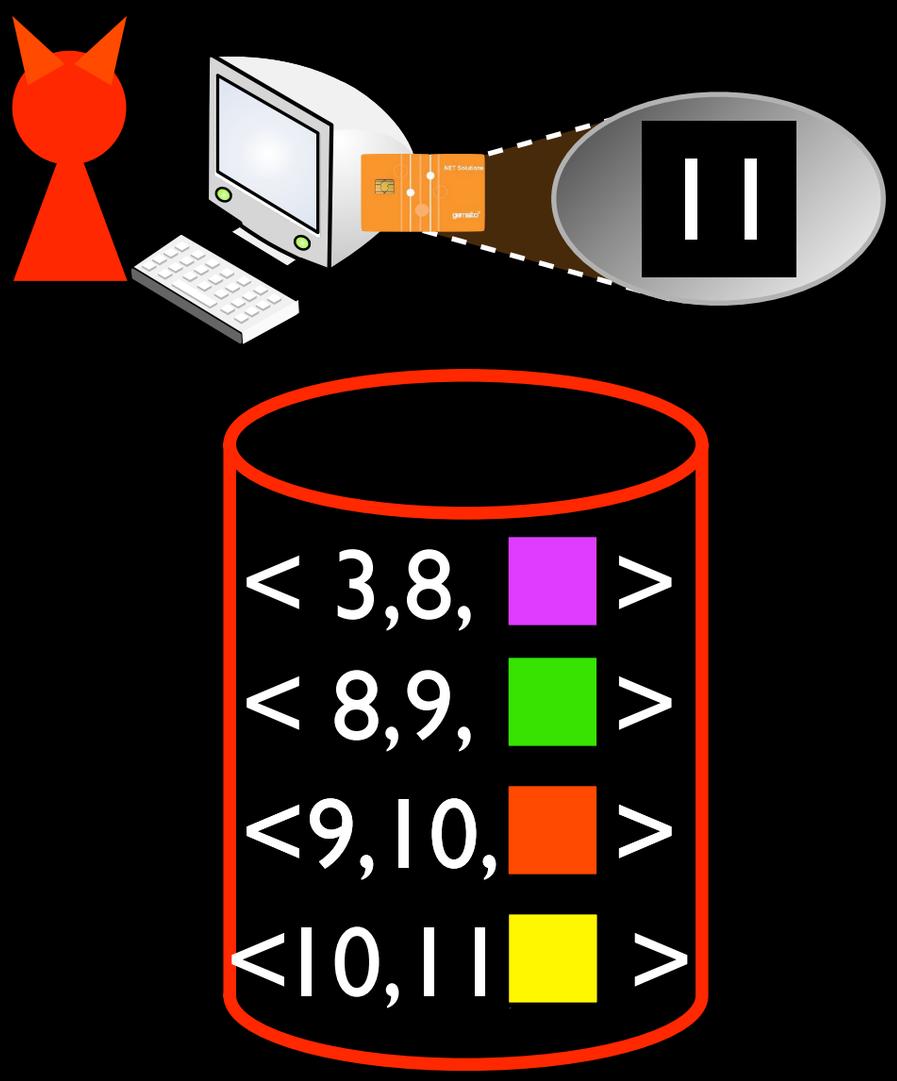
Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

Implementing a trusted log in TrInc



Untrusted storage

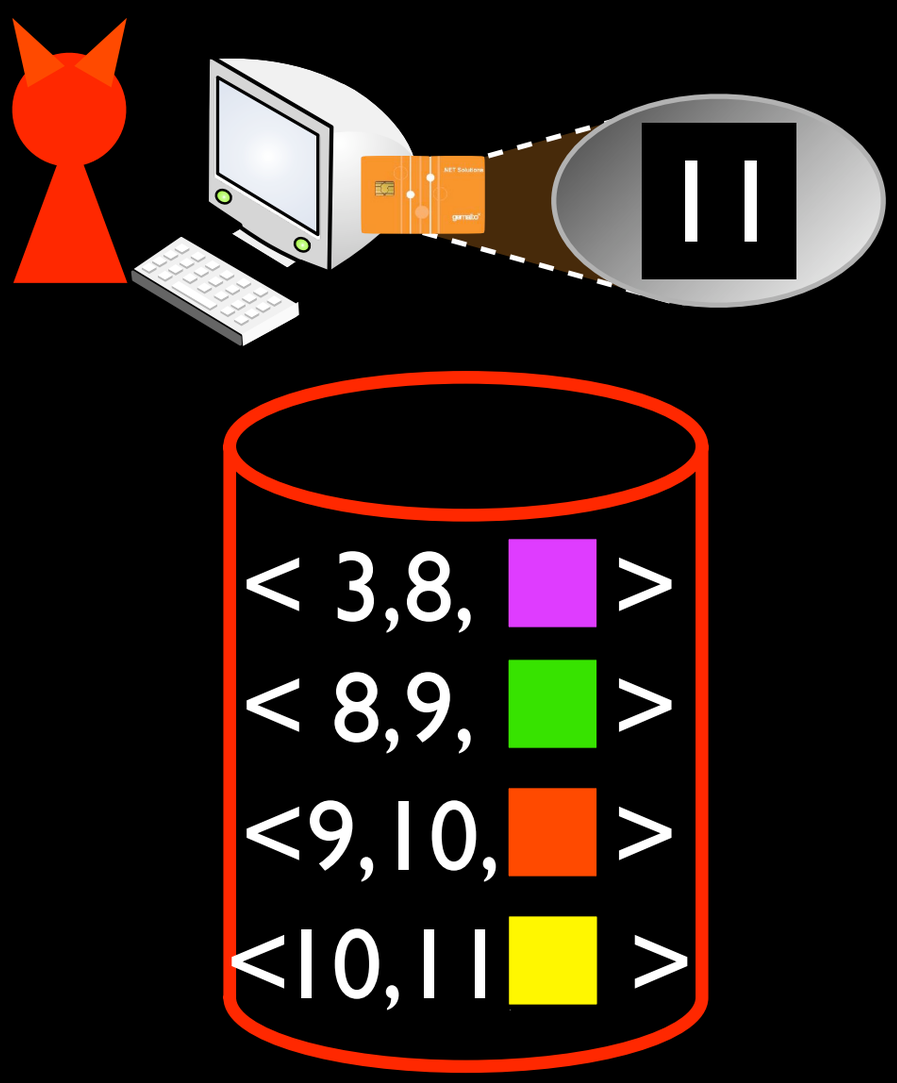
Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

Implementing a trusted log in TrInc



Untrusted storage

Append(data):

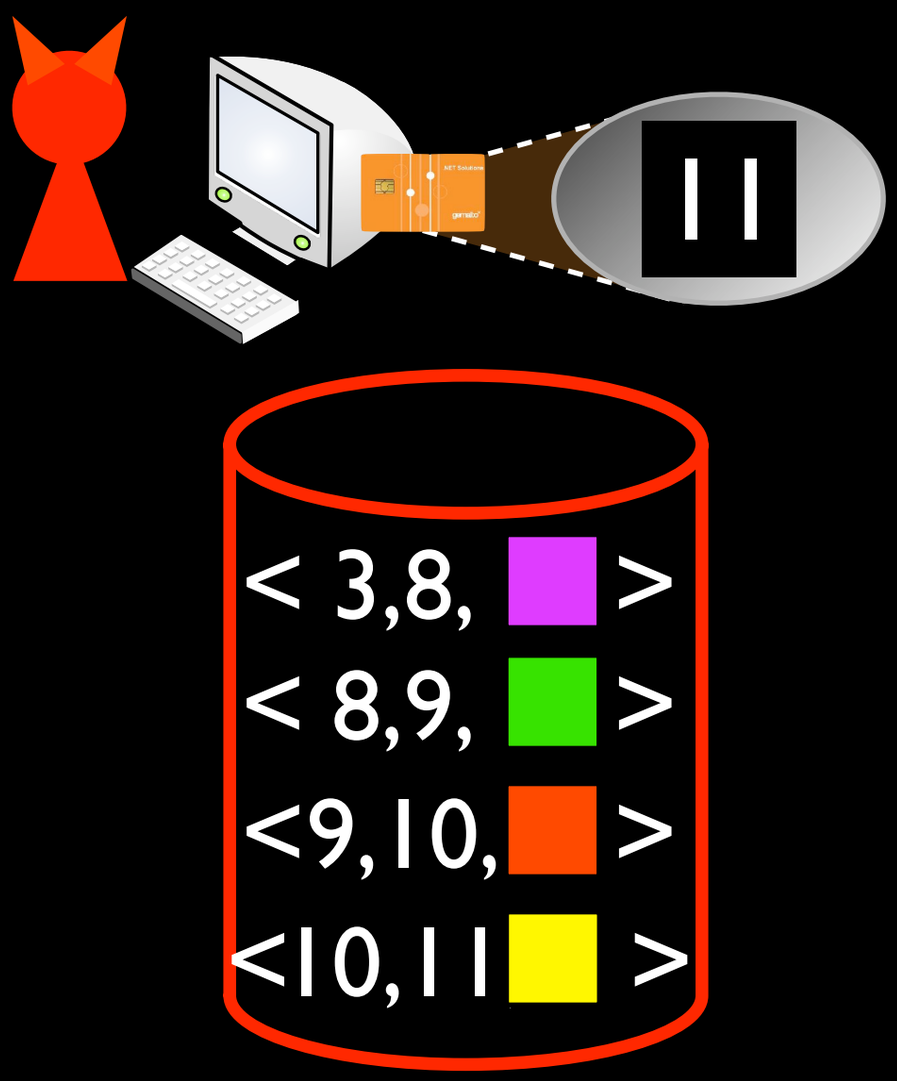
Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

<9,10, >

Implementing a trusted log in TrInc



Untrusted storage

Append(data):

Bind new data to the end of the log

Lookup(sequence num):

No equivocating on what is or is not stored

<9,10, > Fast lookups

Few hardware accesses

TrInc-A2M

- Attested Append-only Memory (A2M)
  - Stores logs in trusted storage
  - Accesses trusted storage for all methods

- A2M shown to solve
  - Byzantine fault tolerance using fewer nodes
  - SUNDR file system
  - Quorum/Update protocol

- By construction, TrInc solves these systems, too

What can TrInc do?

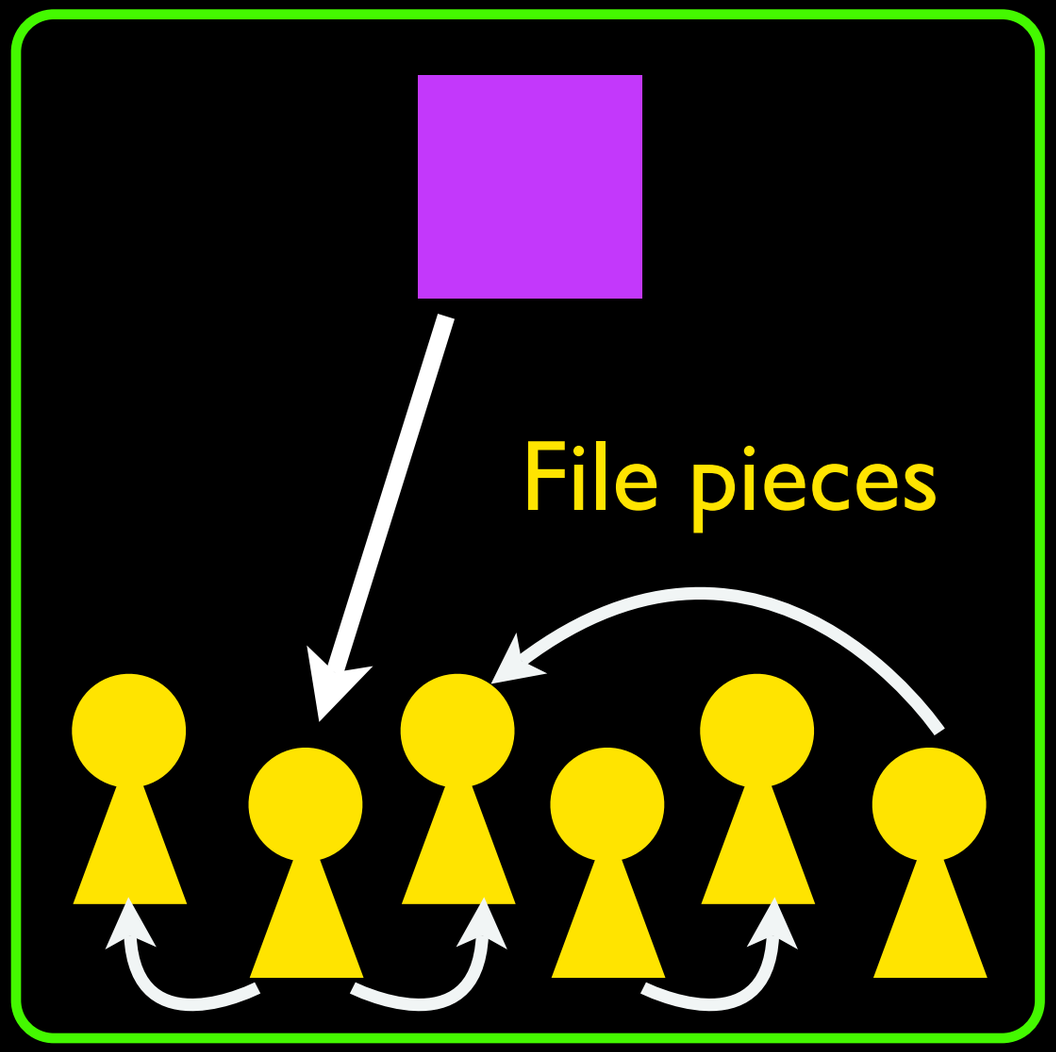
- Trusted append-only logs
- Prevent under-reporting in BitTorrent
- Reduces communication in PeerReview
- BFT with fewer nodes and messages
- Ensure fresh data in DHTs
- Prevent Sybil attacks

What can TrInc do?

- Trusted append-only logs
- Prevent under-reporting in BitTorrent
- Reduces communication in PeerReview
- BFT with fewer nodes and messages
- Ensure fresh data in DHTs
- Prevent Sybil attacks

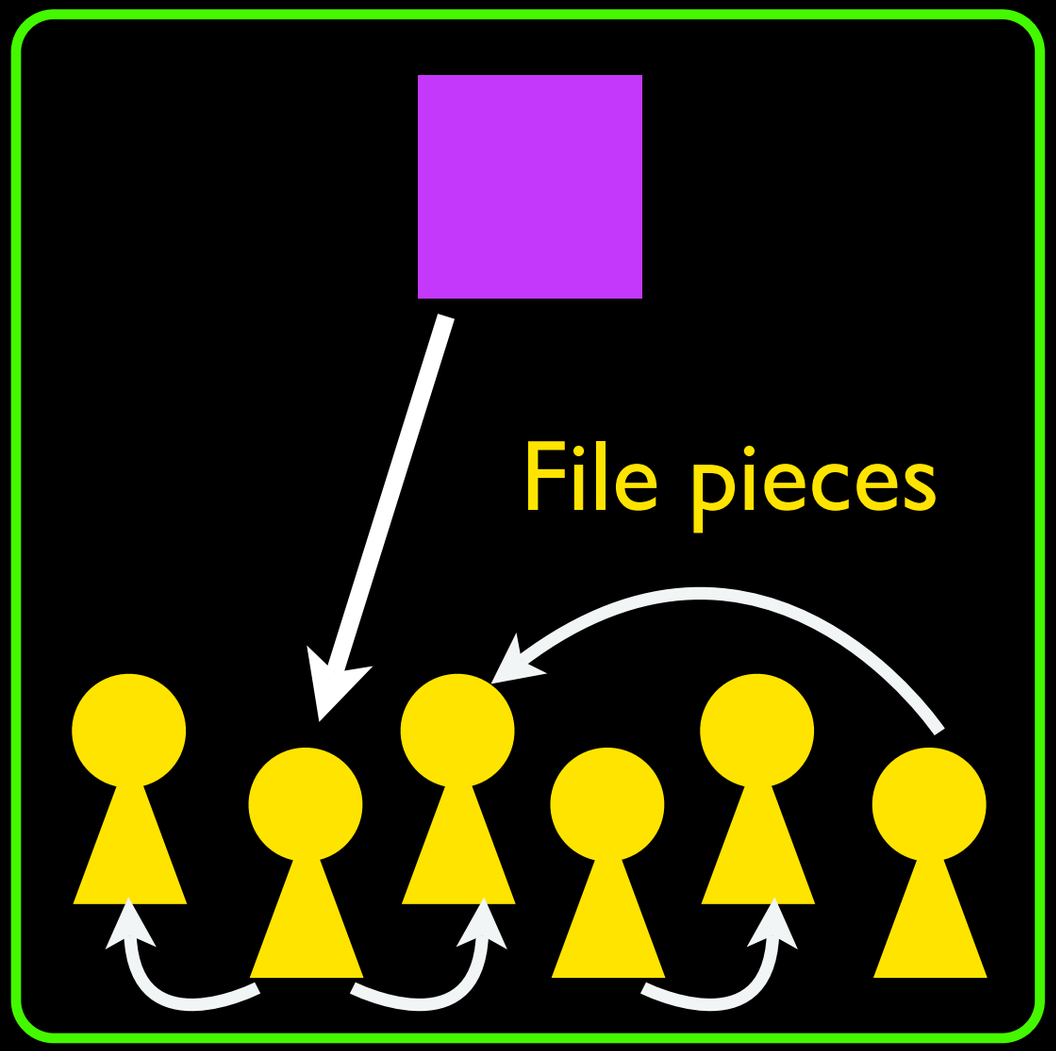
BitTorrent primer

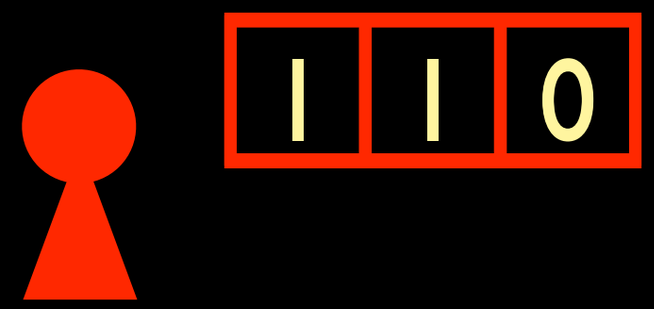
BitTorrent primer



Fast, users share the work

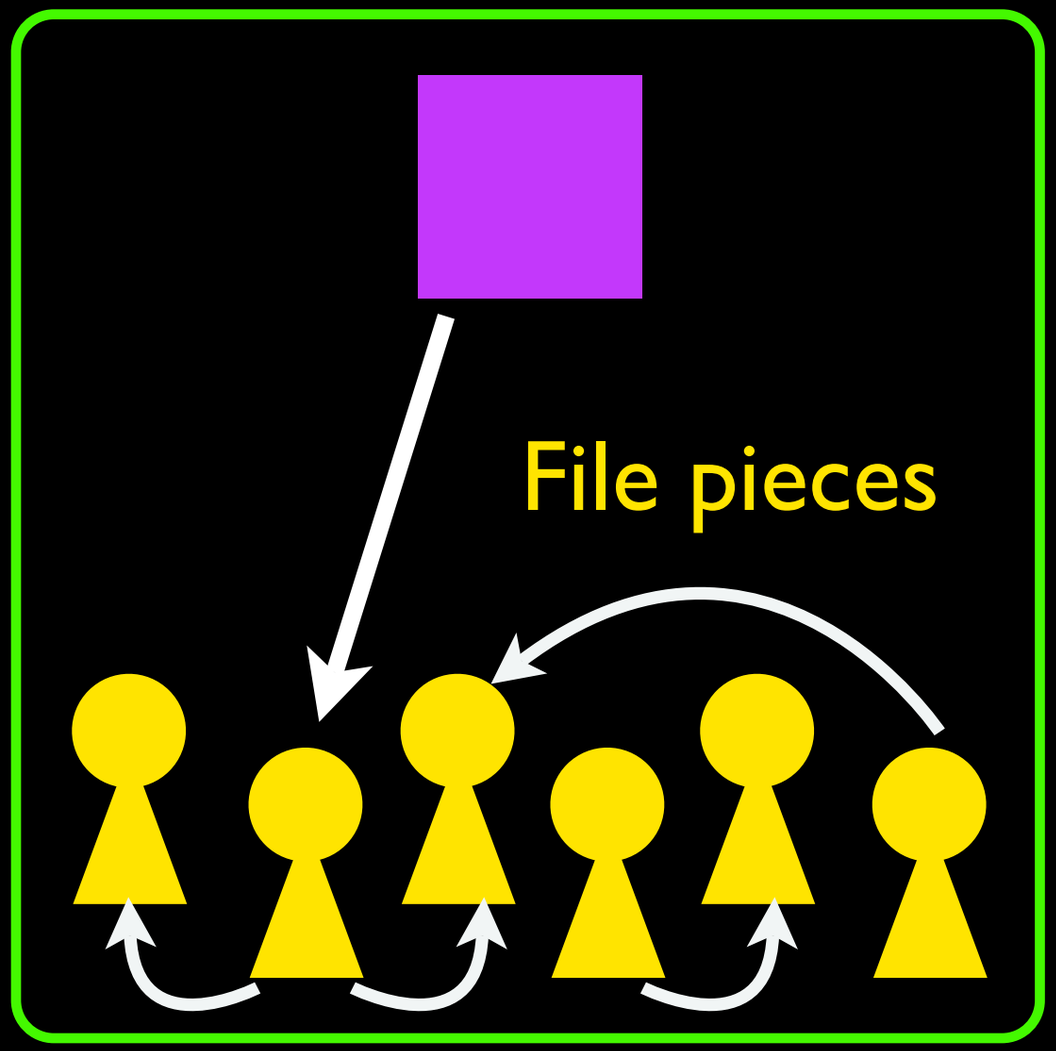
BitTorrent primer



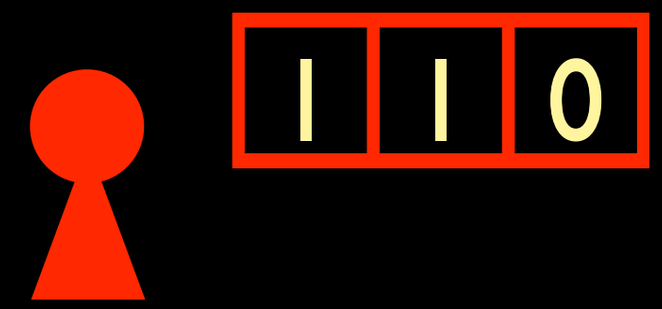


Fast, users share the work

BitTorrent primer

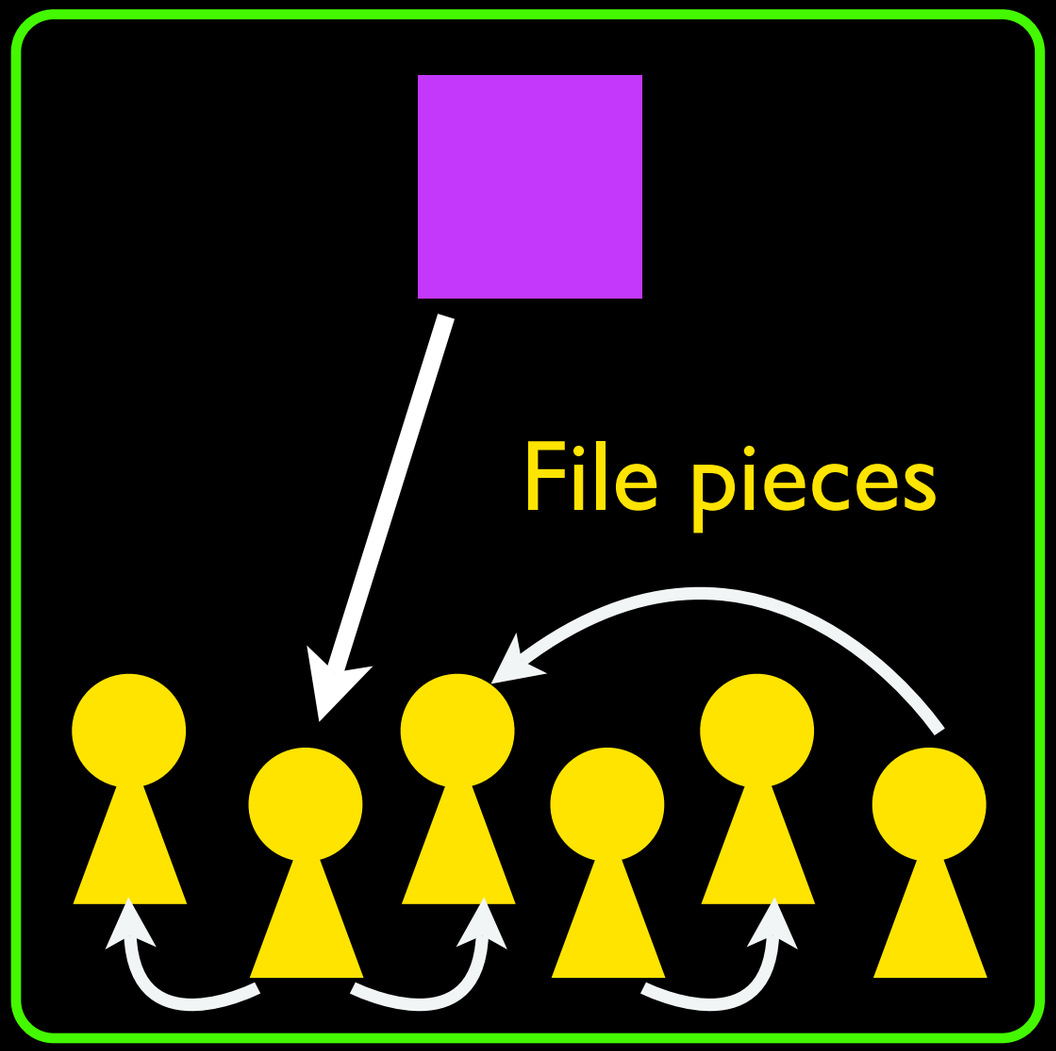


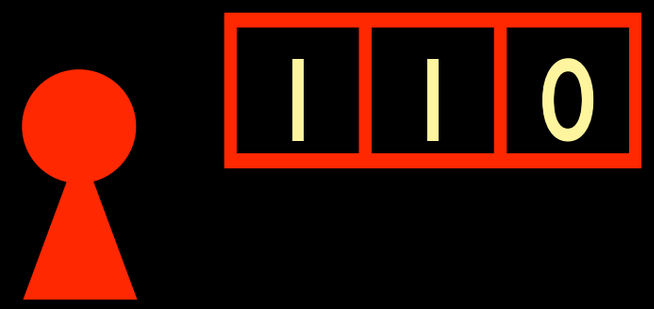
Does not have piece 2



Fast, users share the work

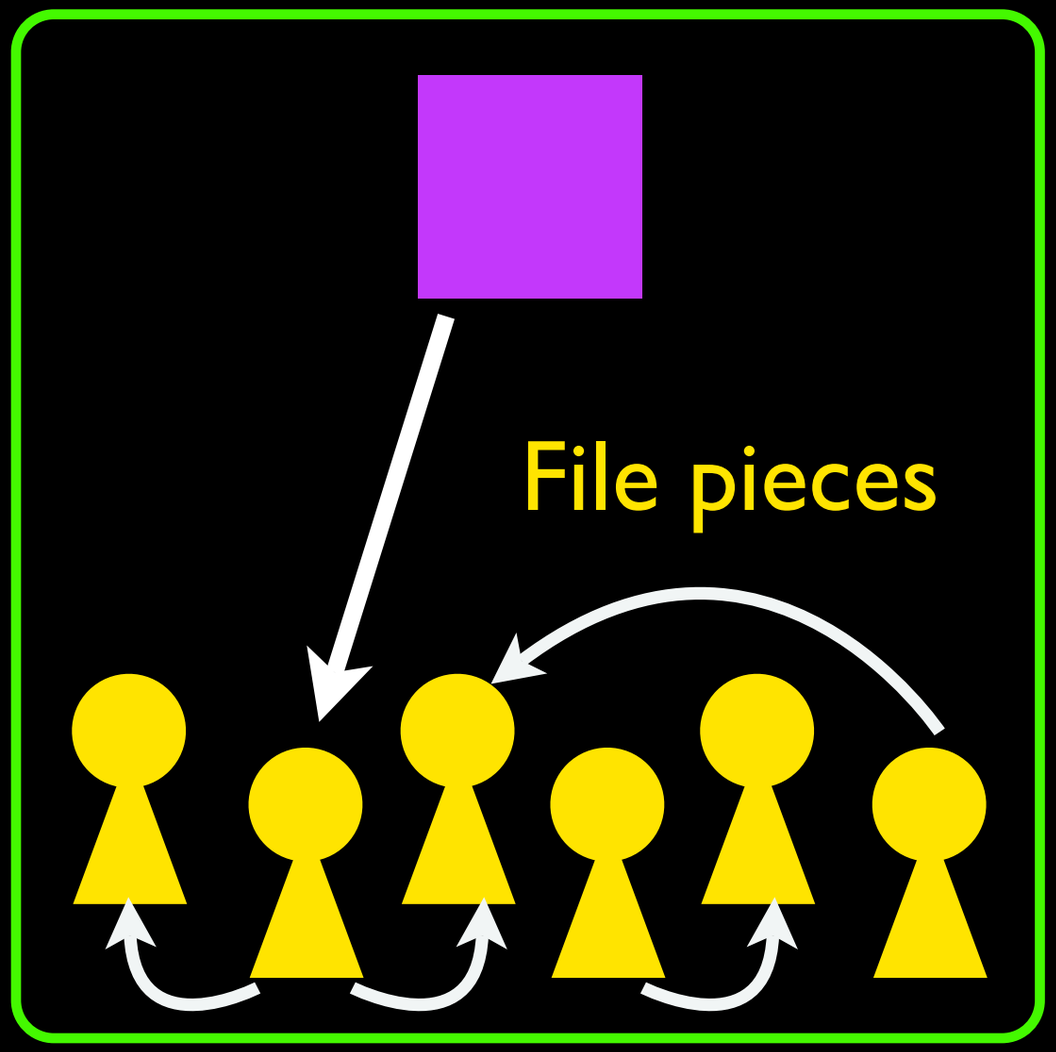
BitTorrent primer

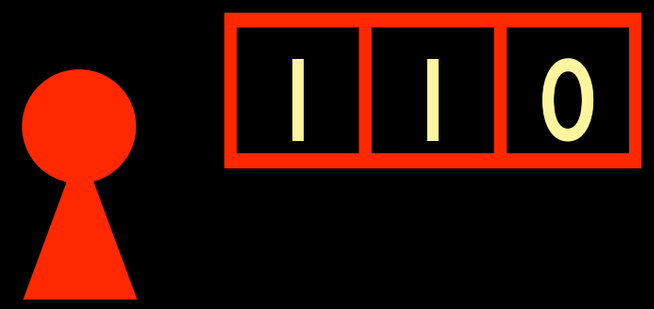


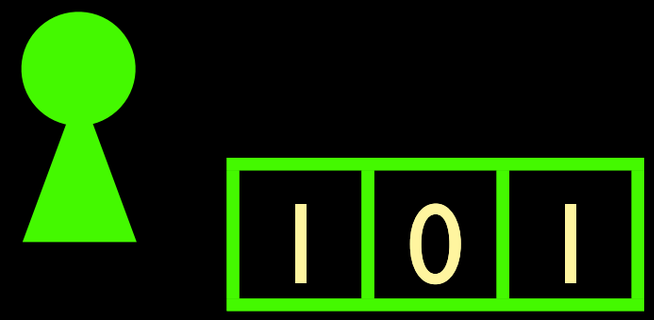


Fast, users share the work

BitTorrent primer

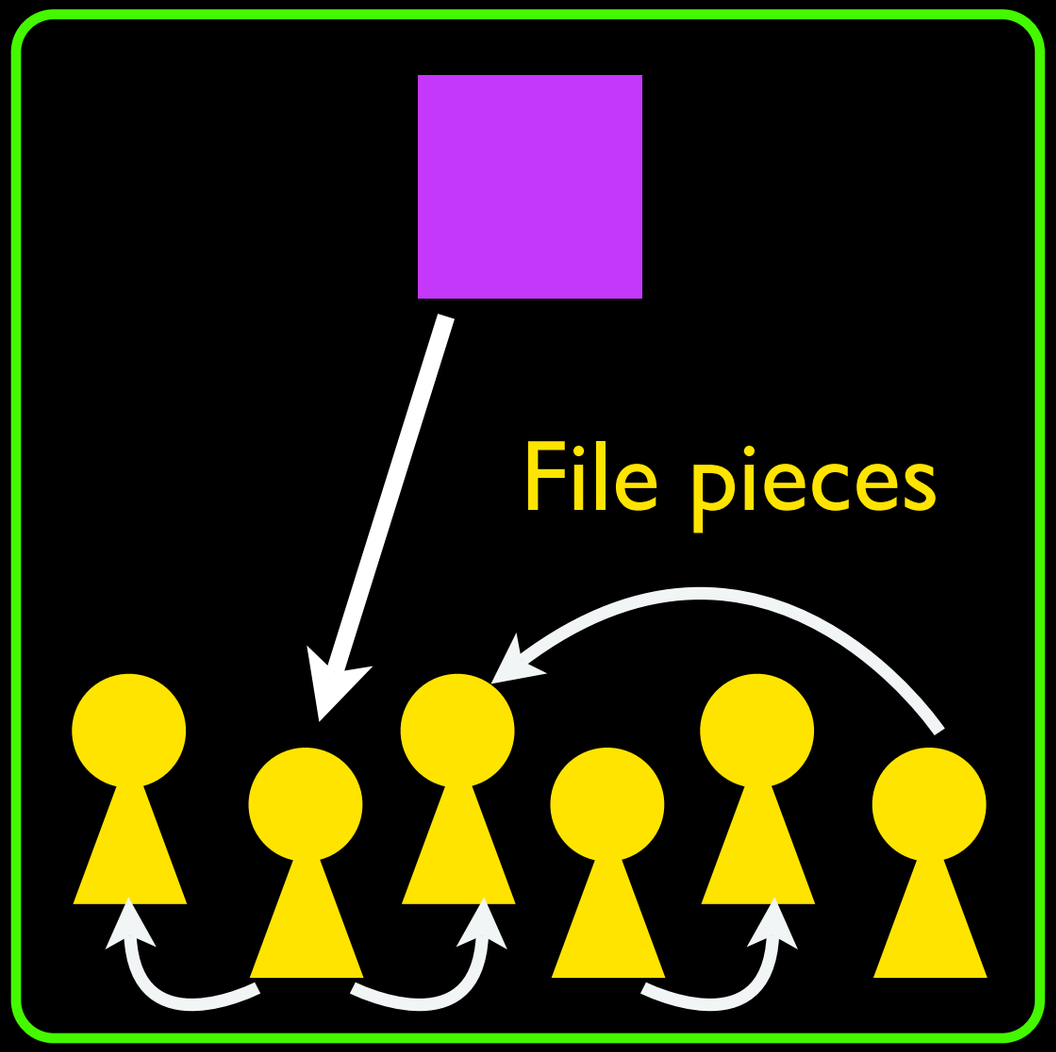


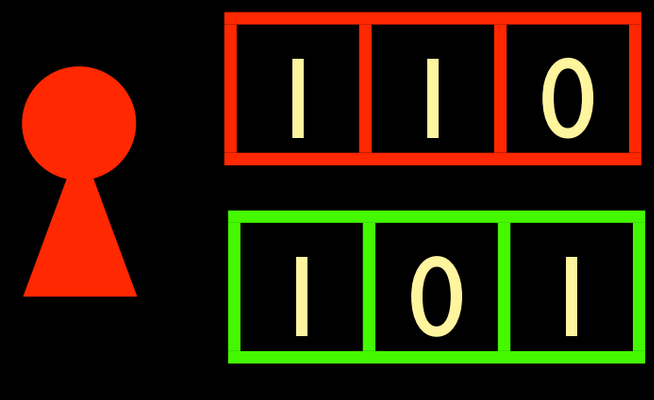


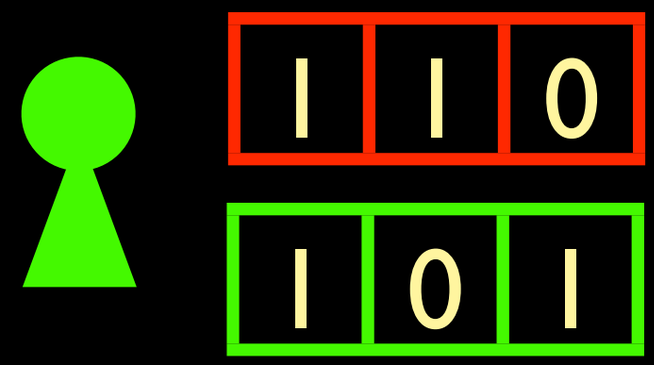


Fast, users share the work

BitTorrent primer

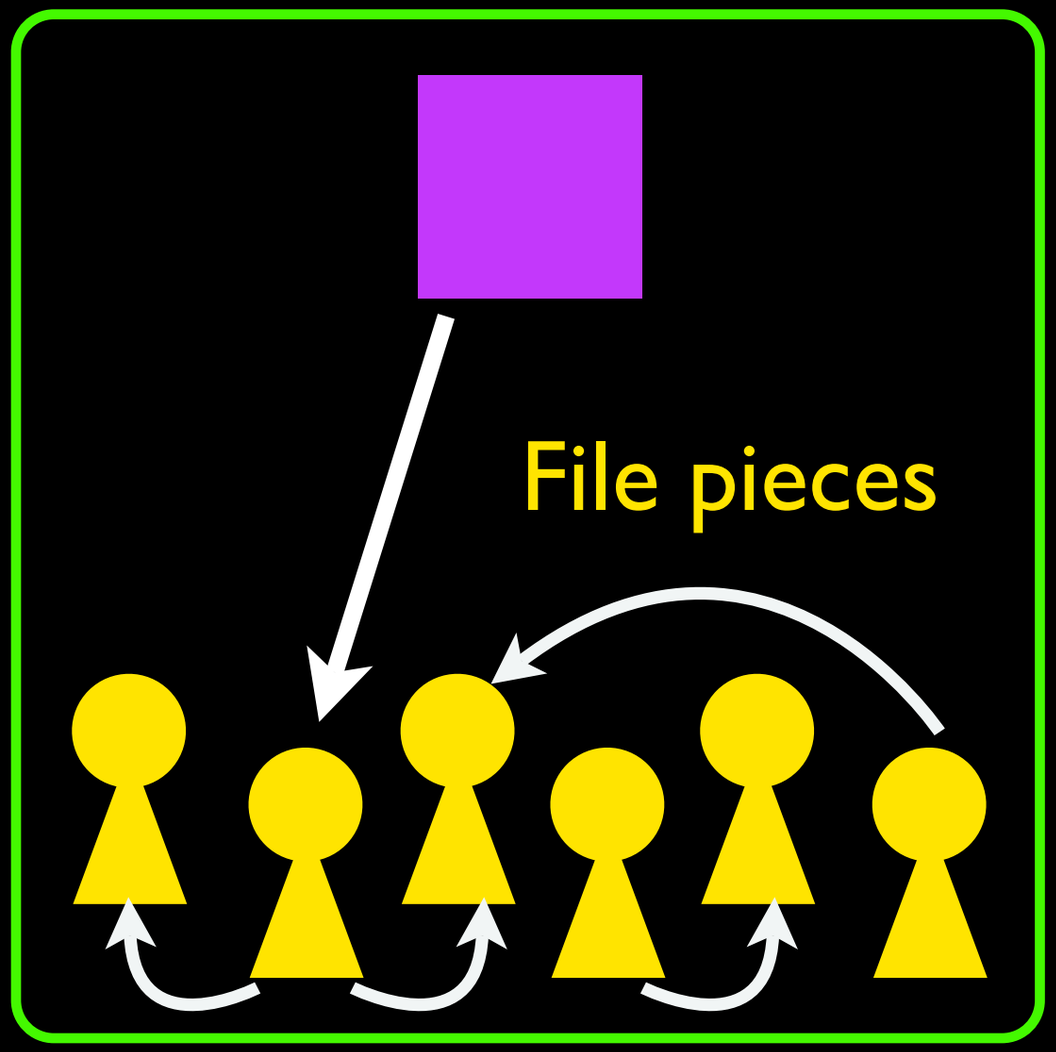


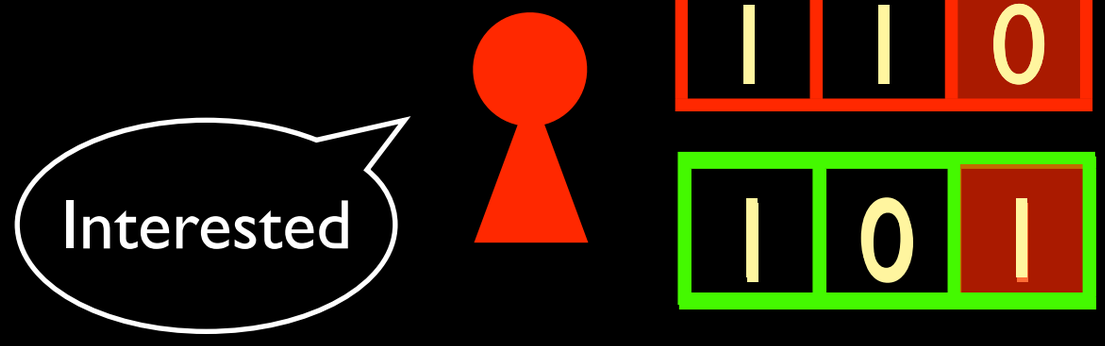


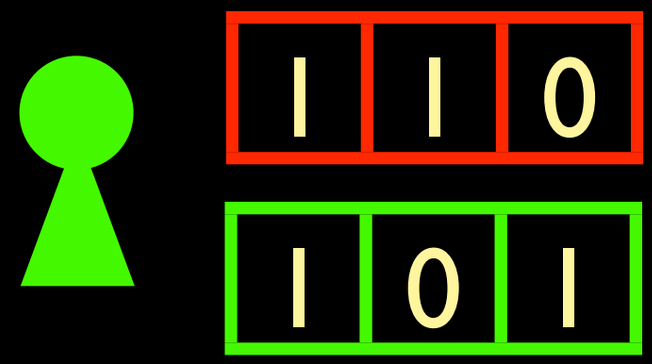


Fast, users share the work

BitTorrent primer

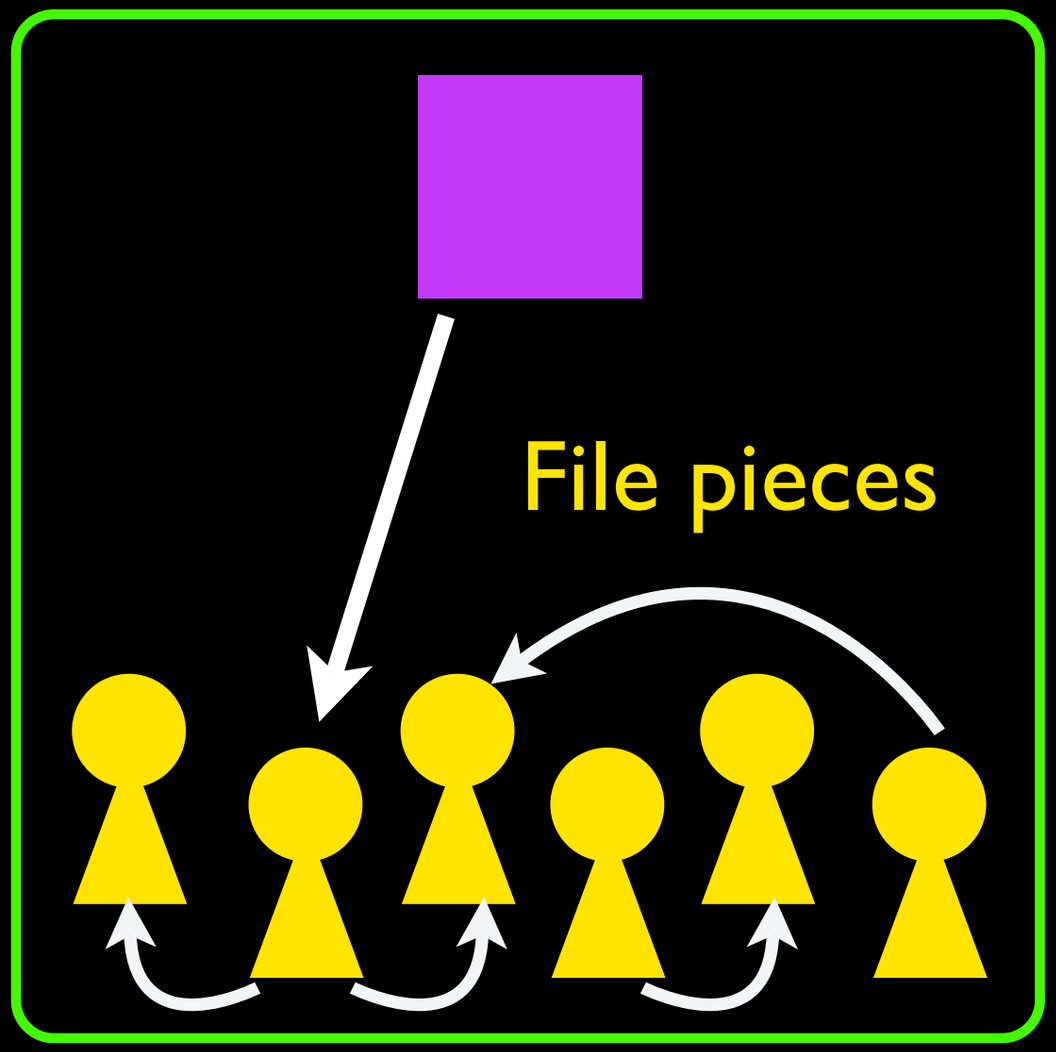


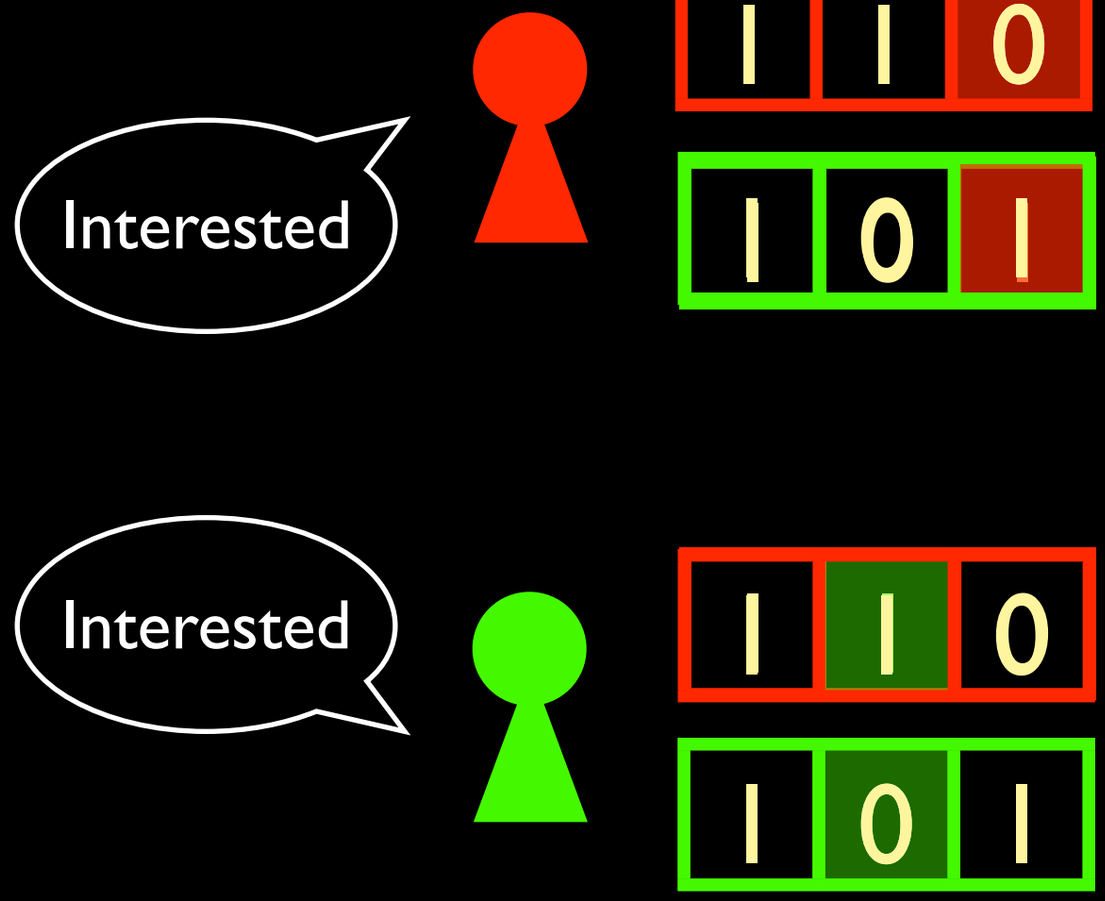




Fast, users share the work

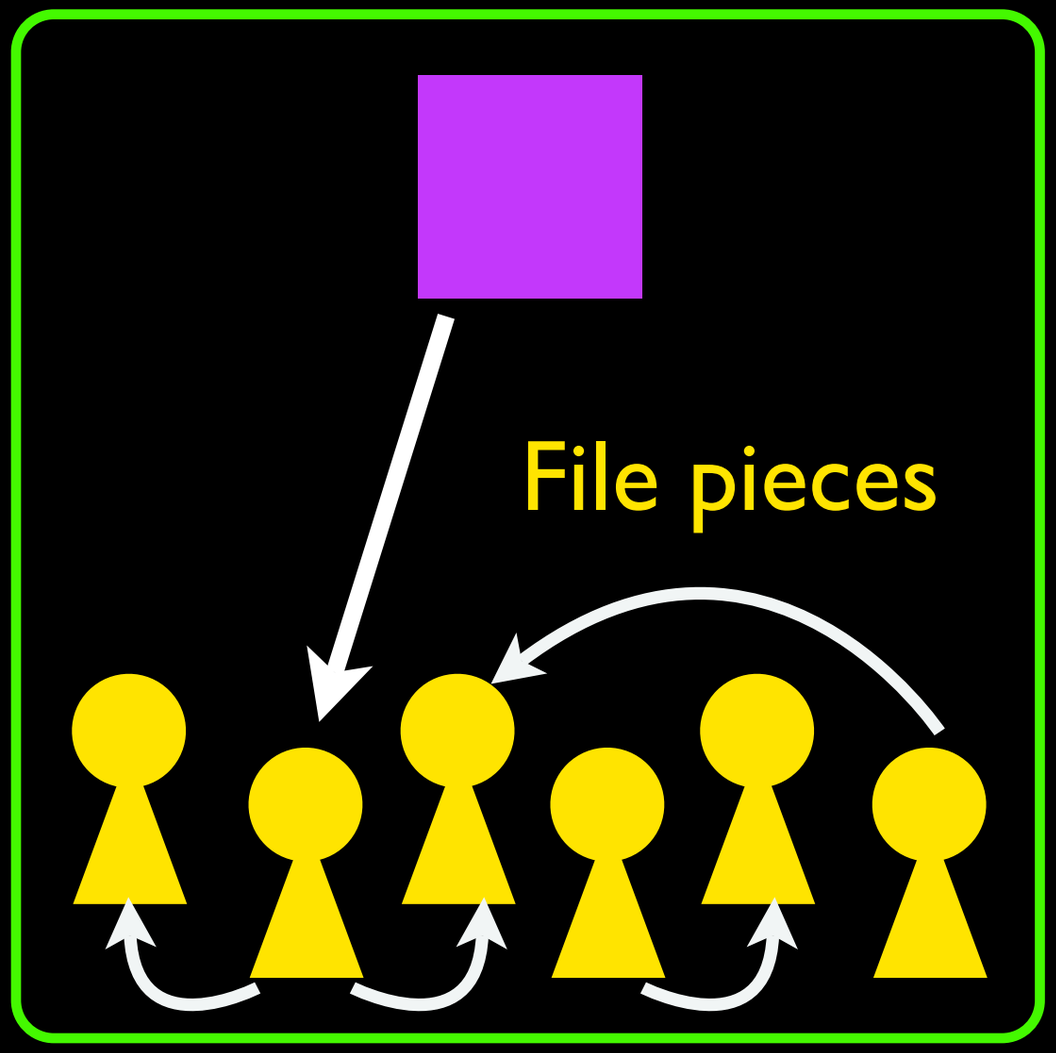
BitTorrent primer

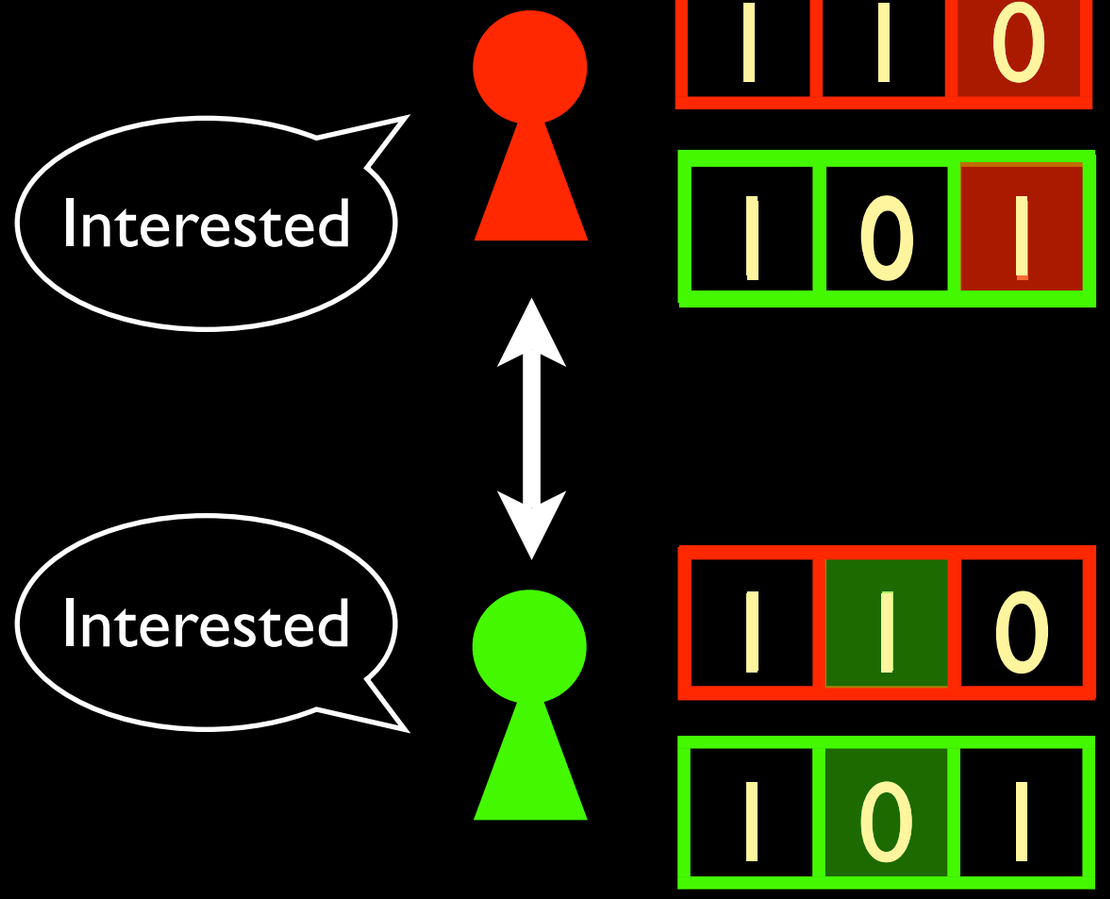




Fast, users share the work

BitTorrent primer

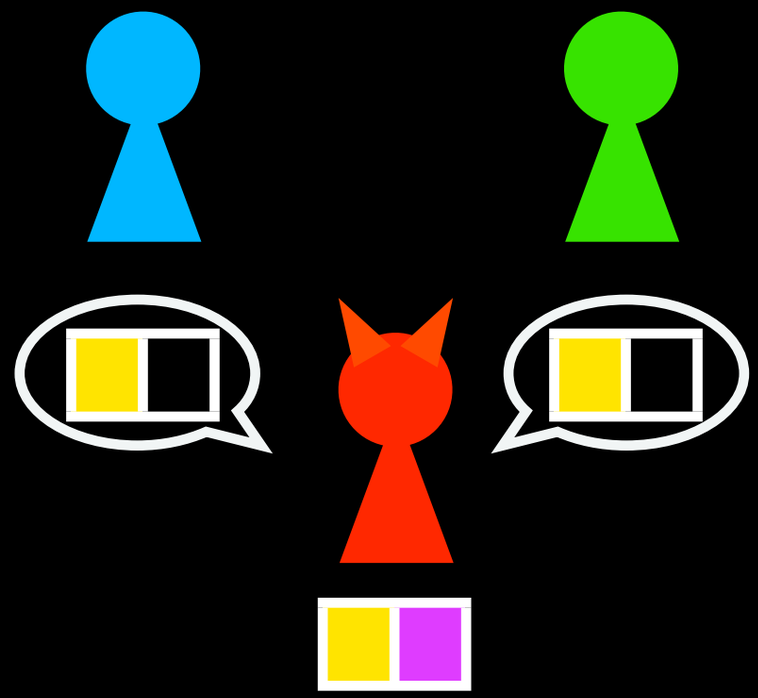




Fast, users share the work

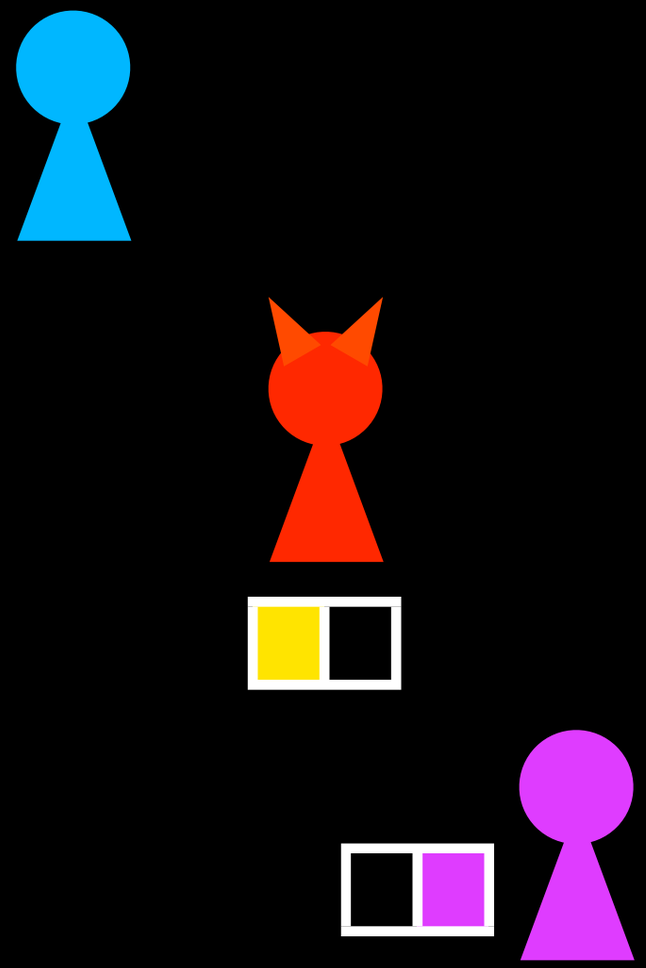
Piece under-reporting is equivocation

[SIGCOMM'08]

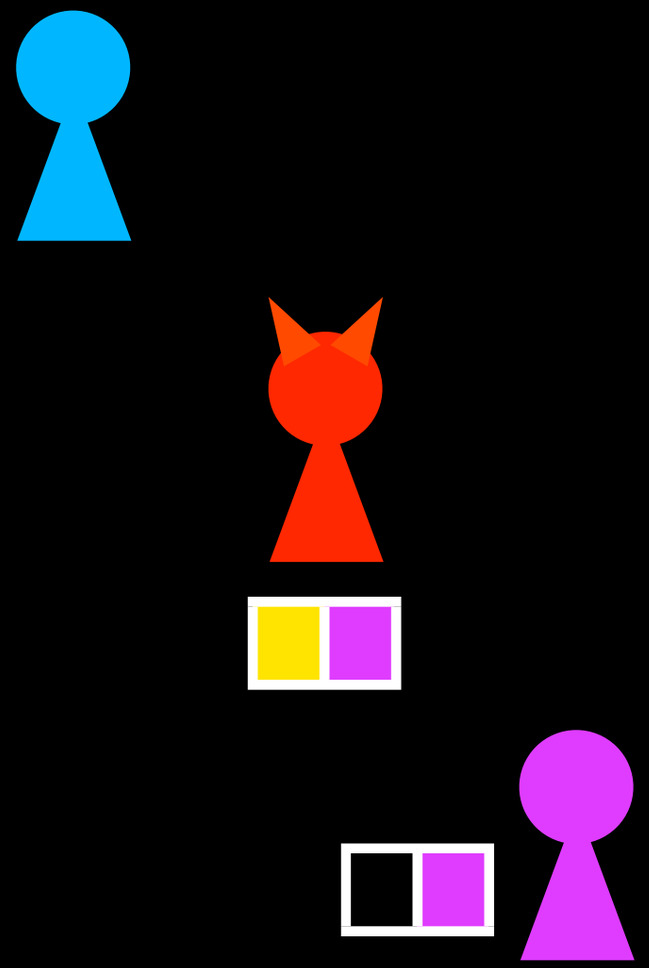


Yields prolonged interest from others
and faster download times

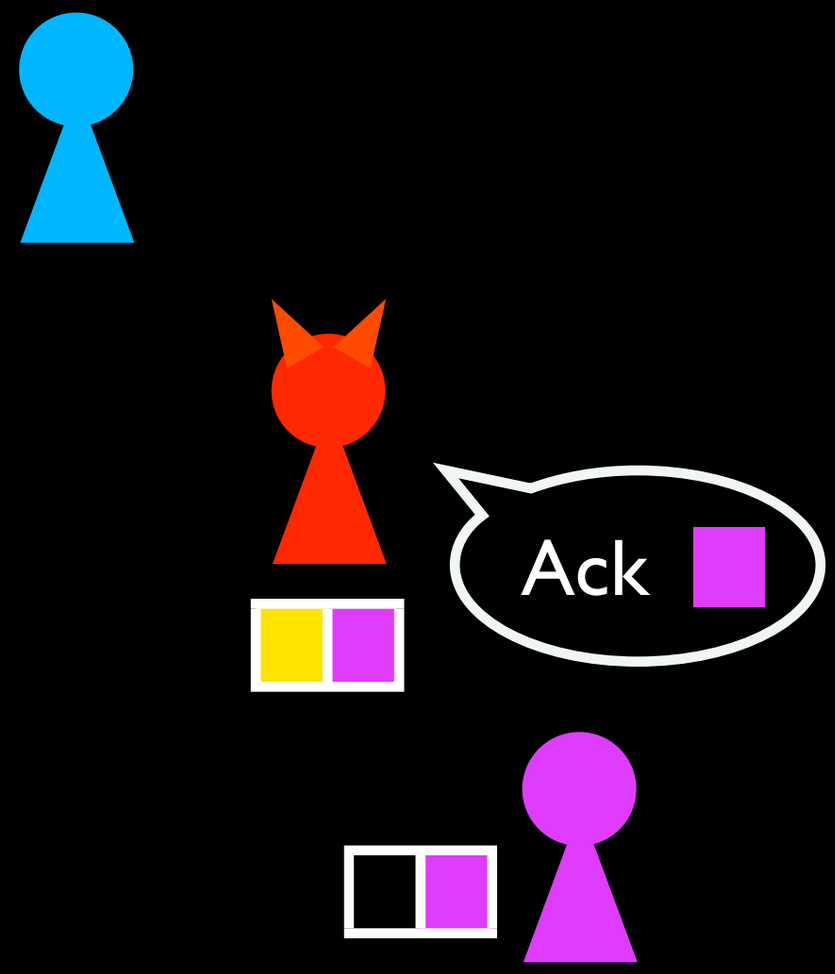
Piece under-reporting is equivocation



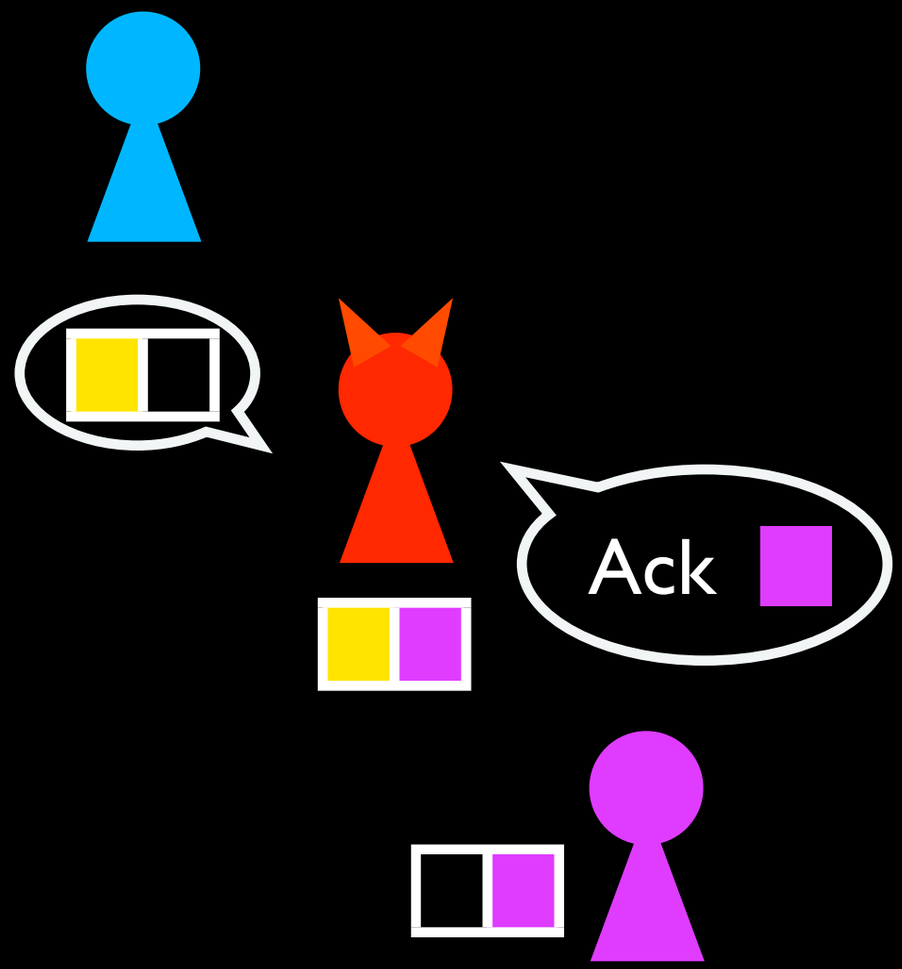
Piece under-reporting is equivocation



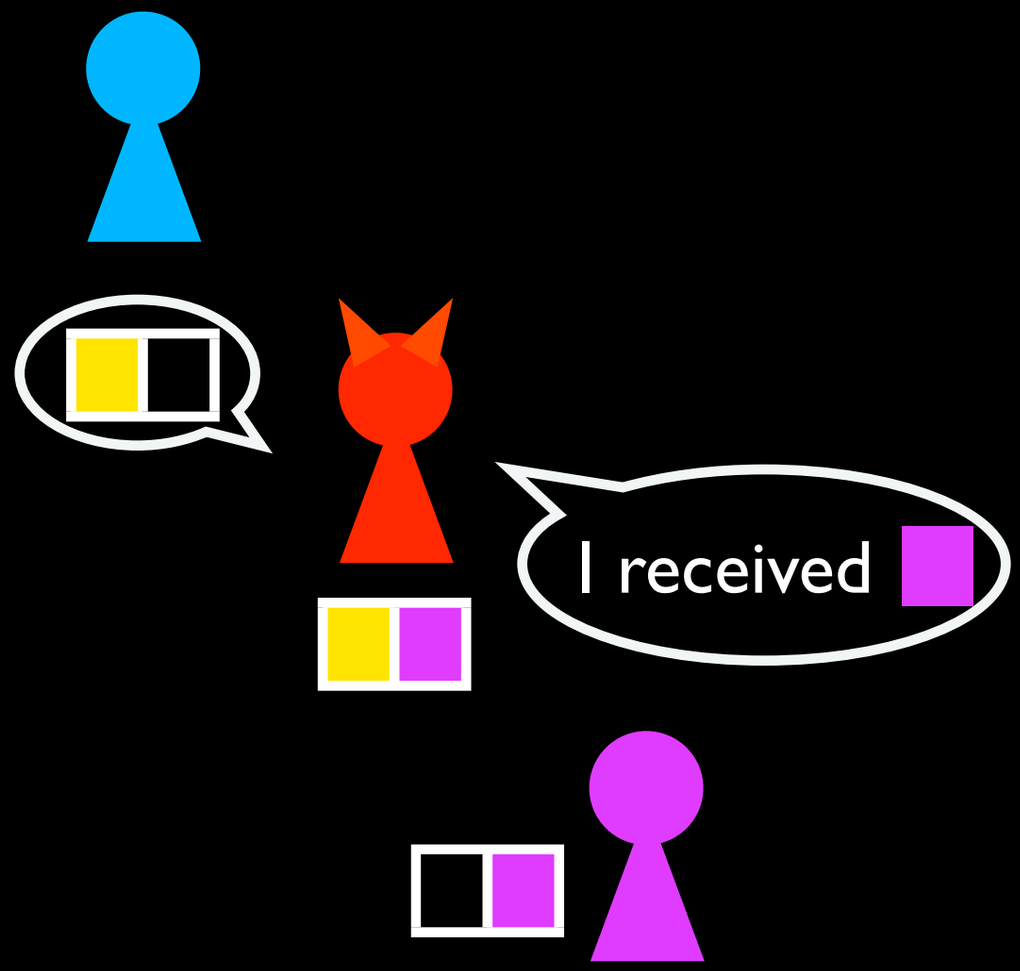
Piece under-reporting is equivocation



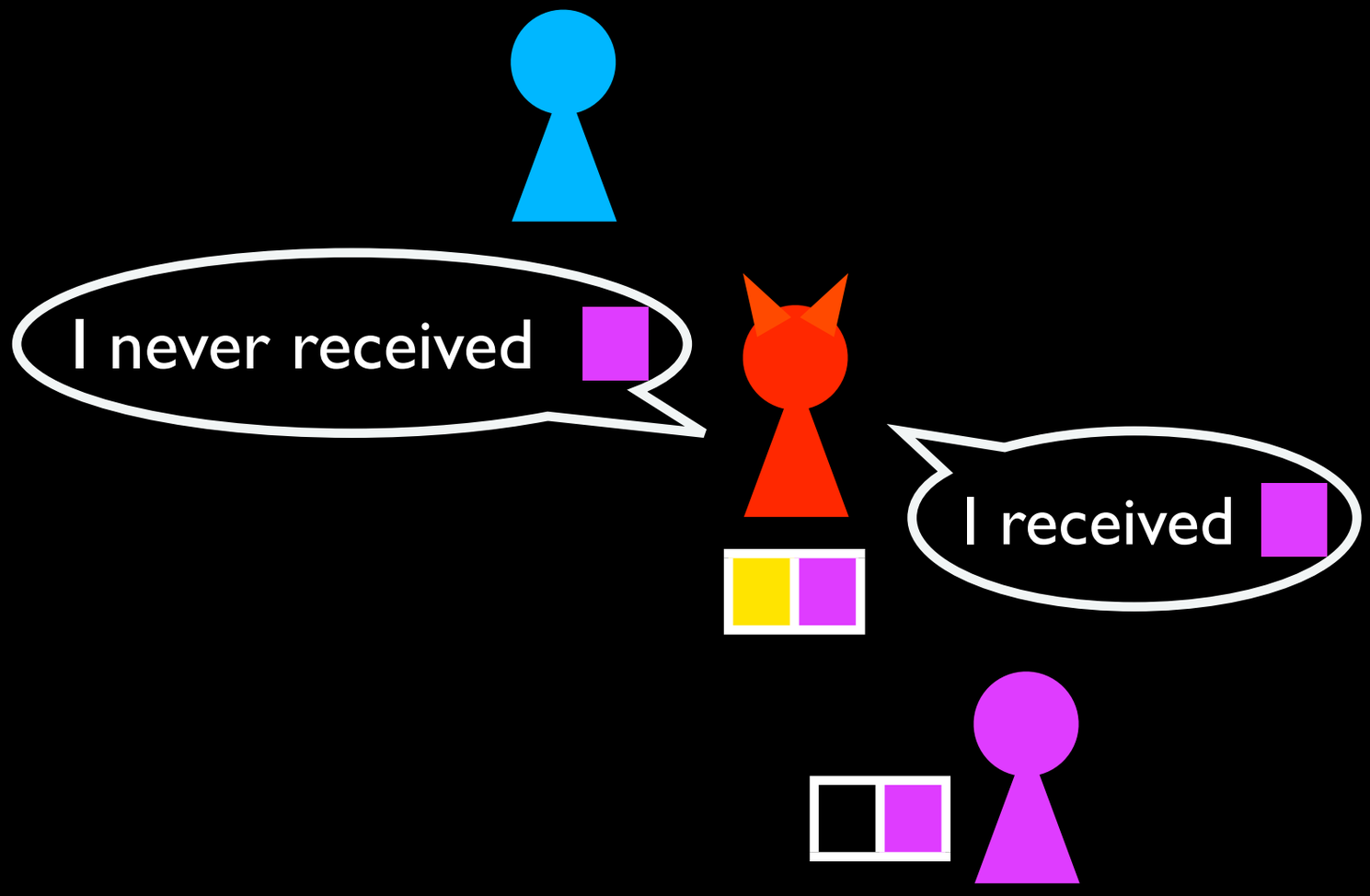
Piece under-reporting is equivocation



Piece under-reporting is equivocation



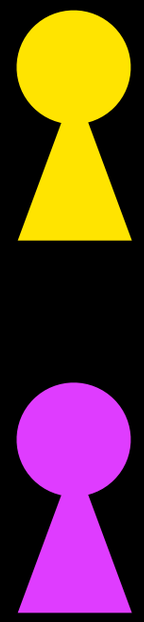
Piece under-reporting is equivocation



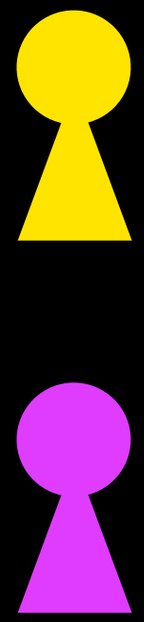
Applying TrInc

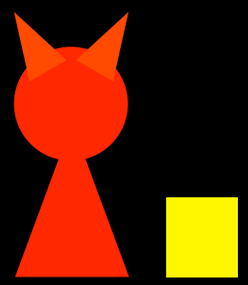
- What does the counter represent?
  - The number of pieces received
- To what do peers attest?
  - Their bitfield
  - The most recent piece received
- When do peers attest?
  - When they receive
  - When they sync their counters

TrInc-BitTorrent



TrInc-BitTorrent

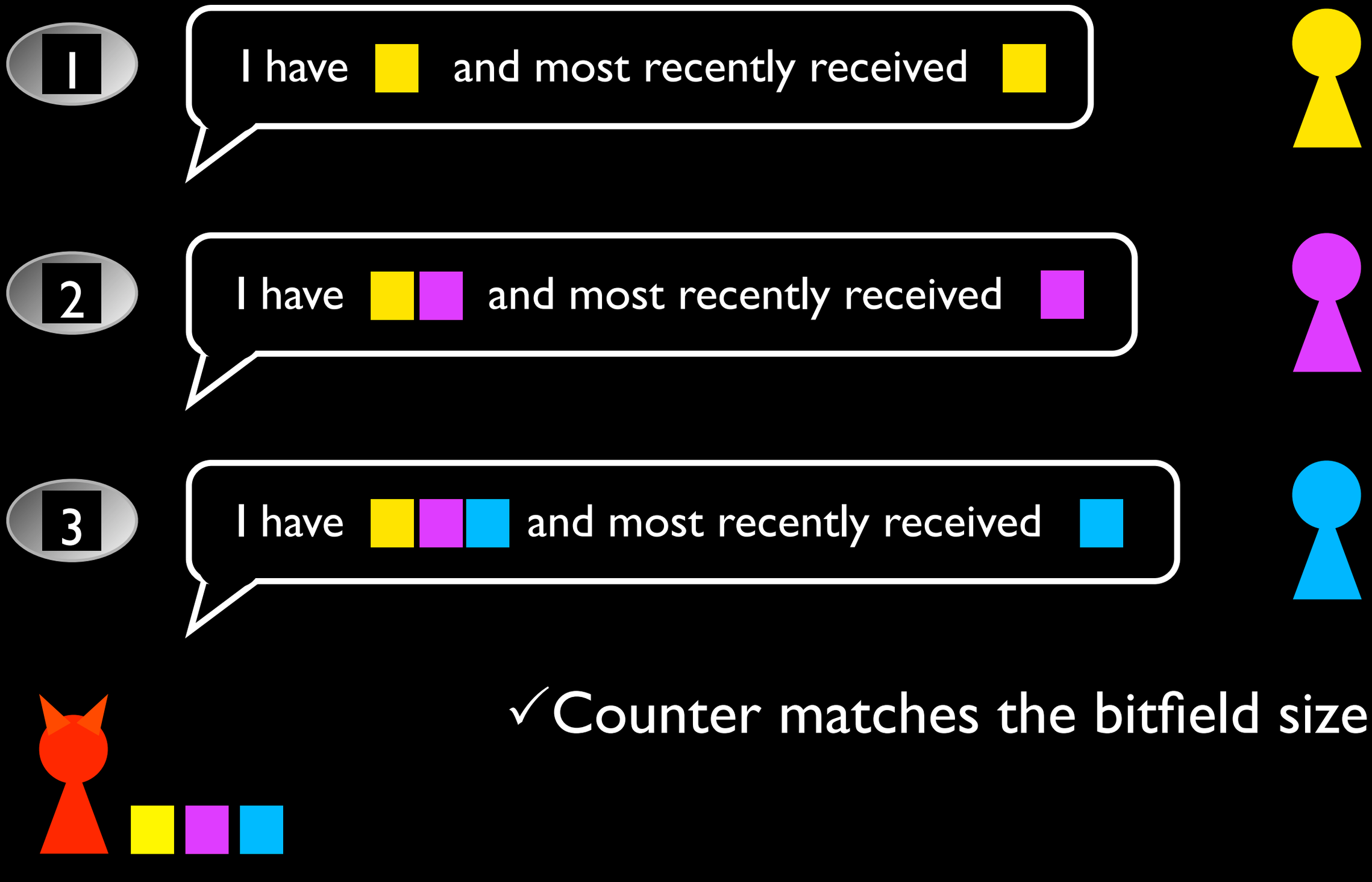




TrInc-BitTorrent

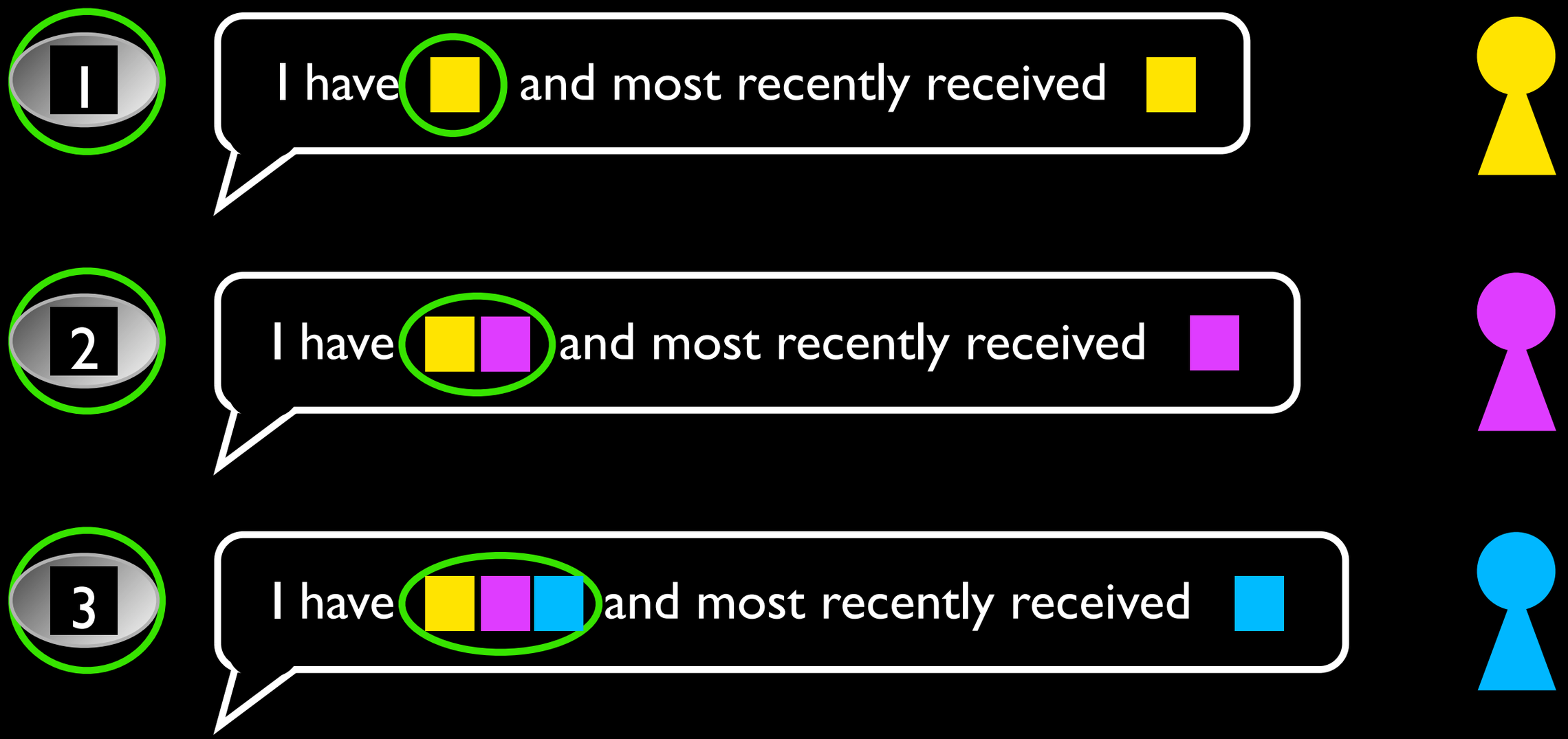
TrInc-BitTorrent

TrInc-BitTorrent



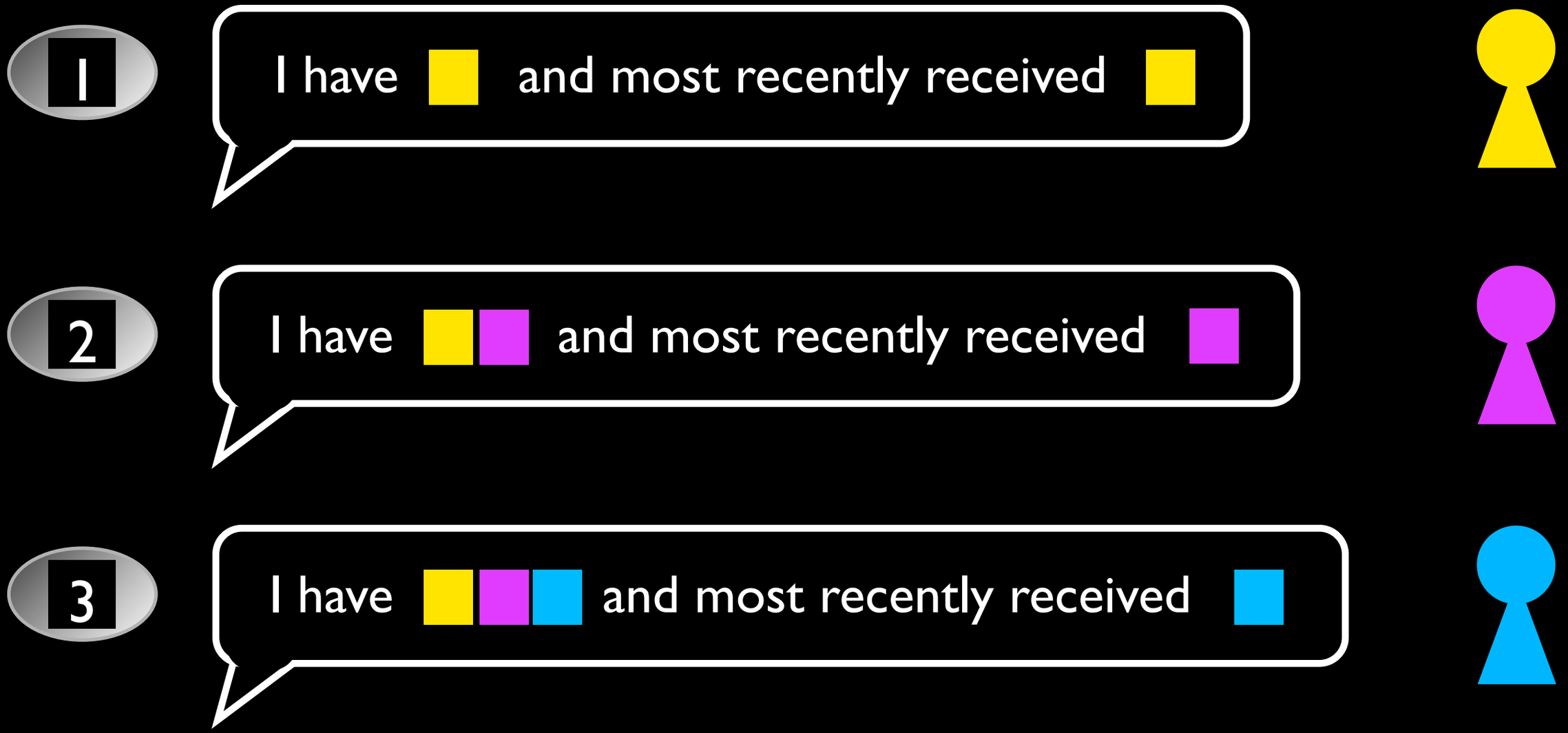
✓Counter matches the bitfield size

TrInc-BitTorrent



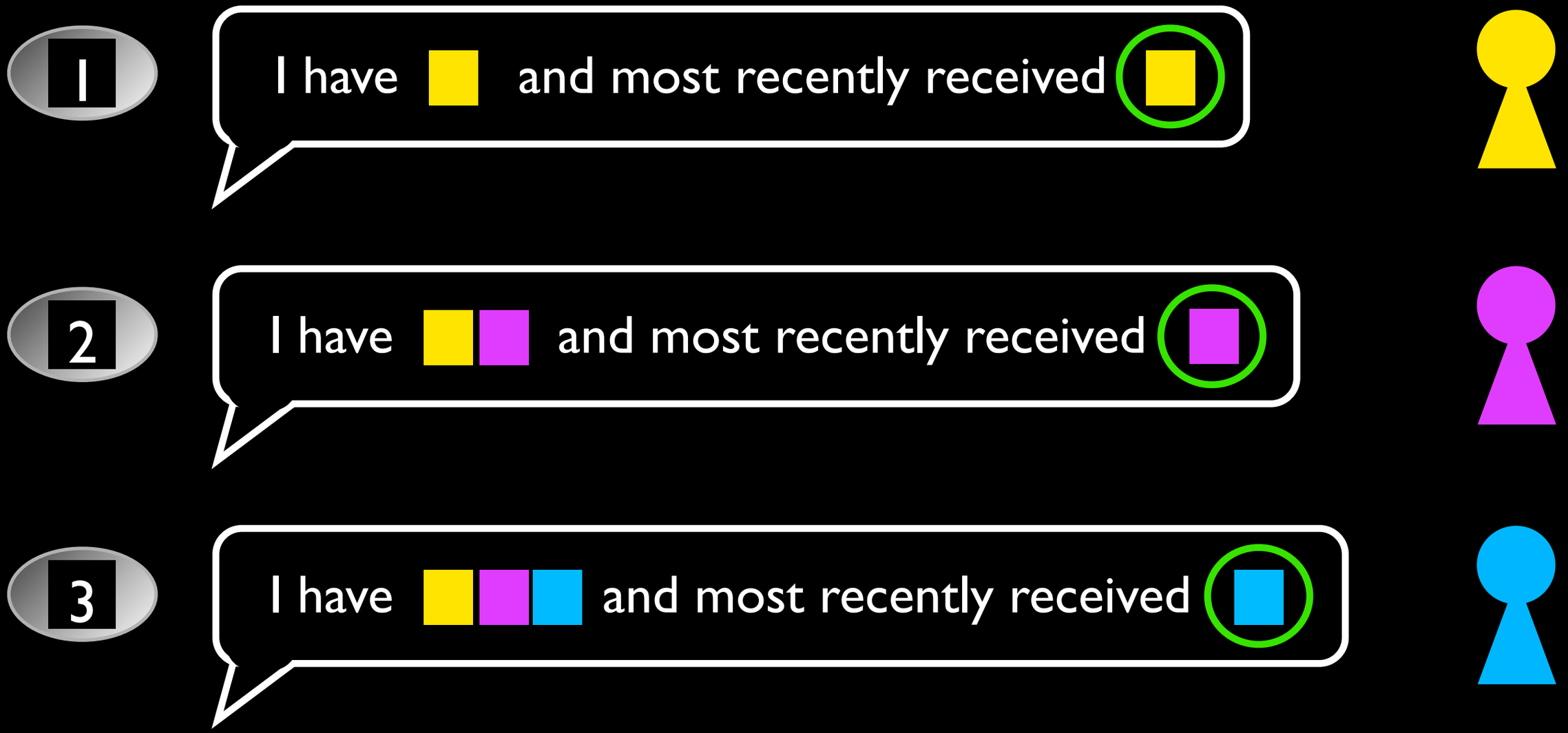
✓Counter matches the bitfield size

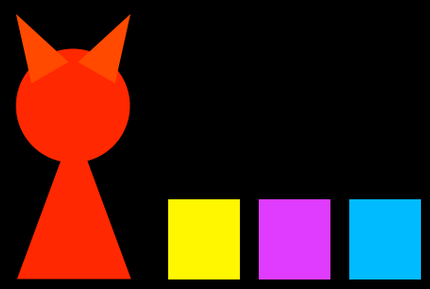
TrInc-BitTorrent



- ✓Counter matches the bitfield size
- ✓Attests to most recent piece

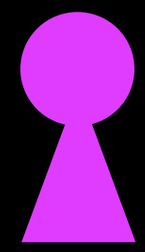
TrInc-BitTorrent



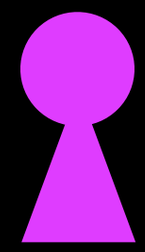


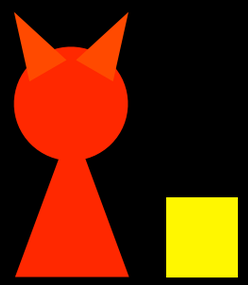
- ✓ Counter matches the bitfield size
- ✓ Attests to most recent piece

Why attest to the latest piece?

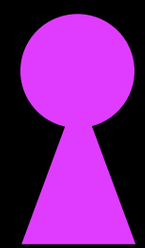


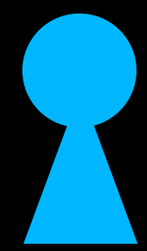
Why attest to the latest piece?

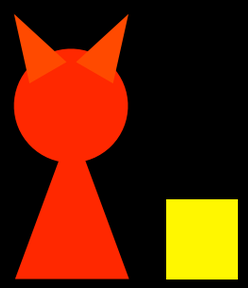




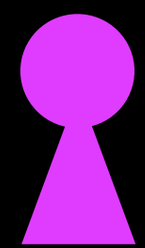
Why attest to the latest piece?

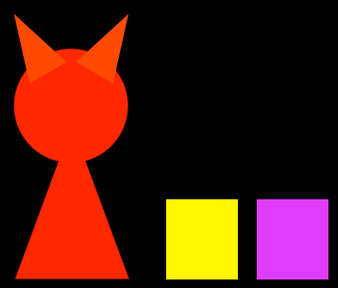




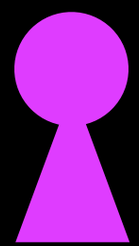


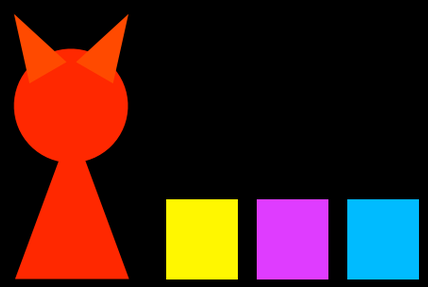
Why attest to the latest piece?



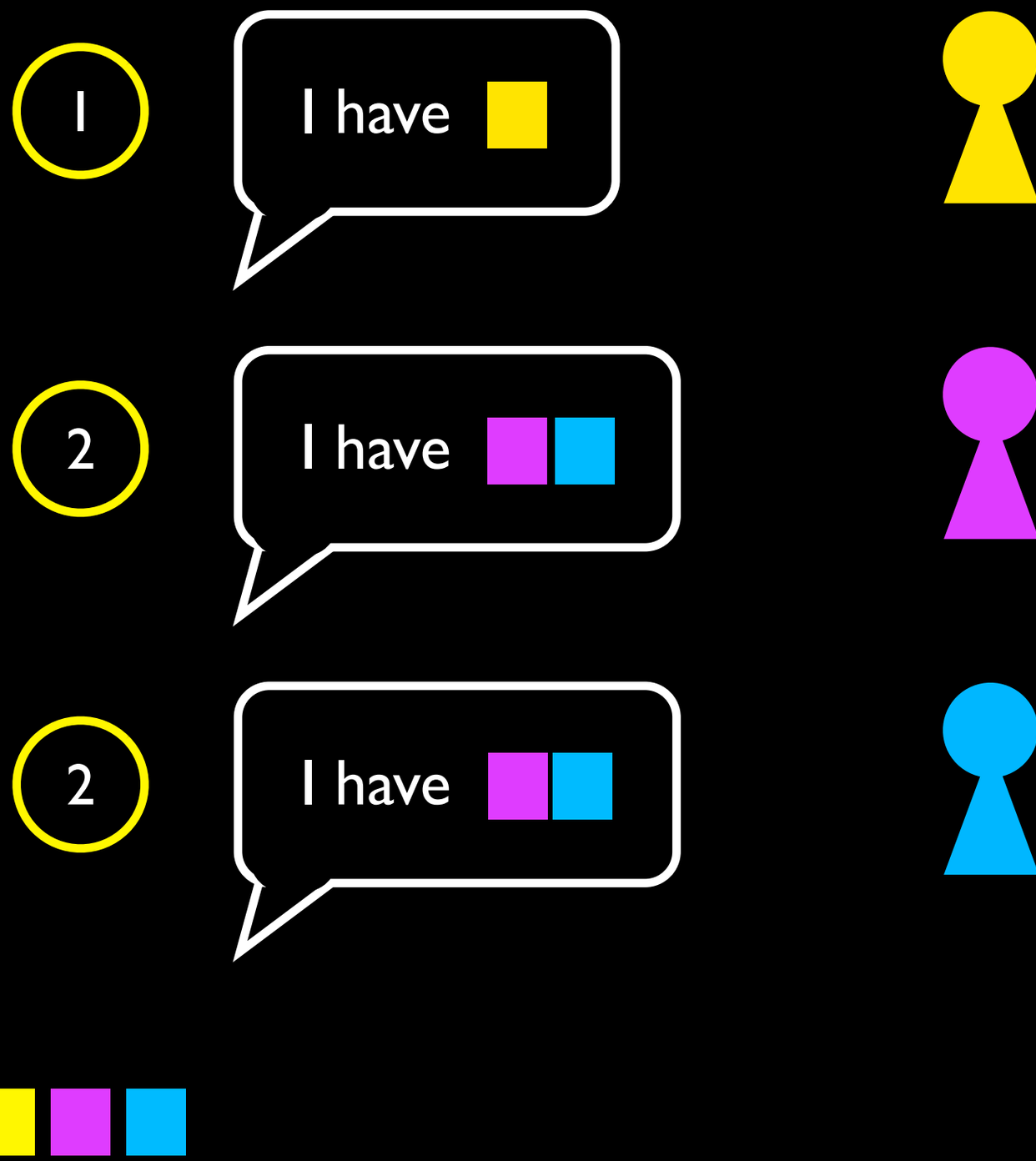


Why attest to the latest piece?

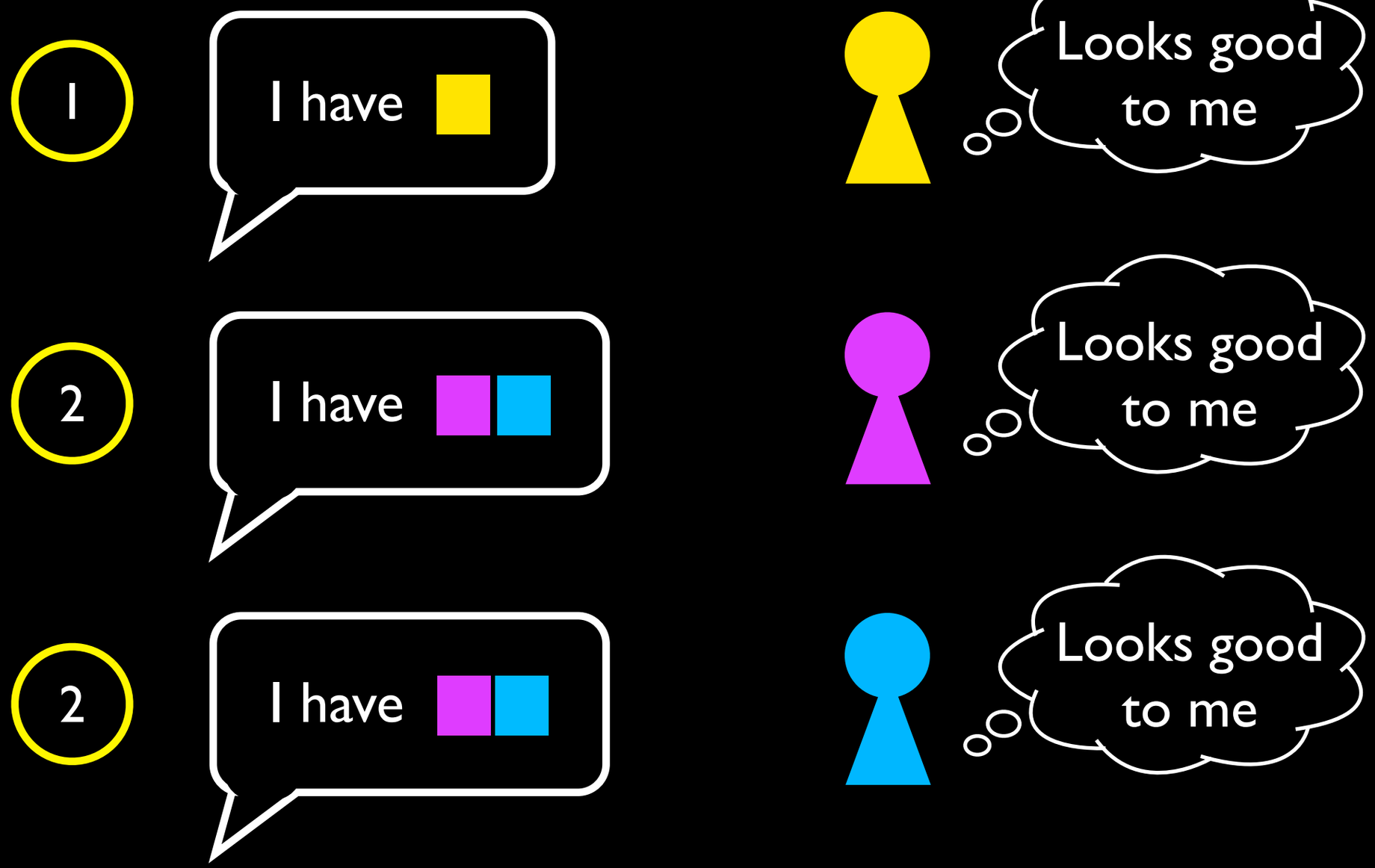




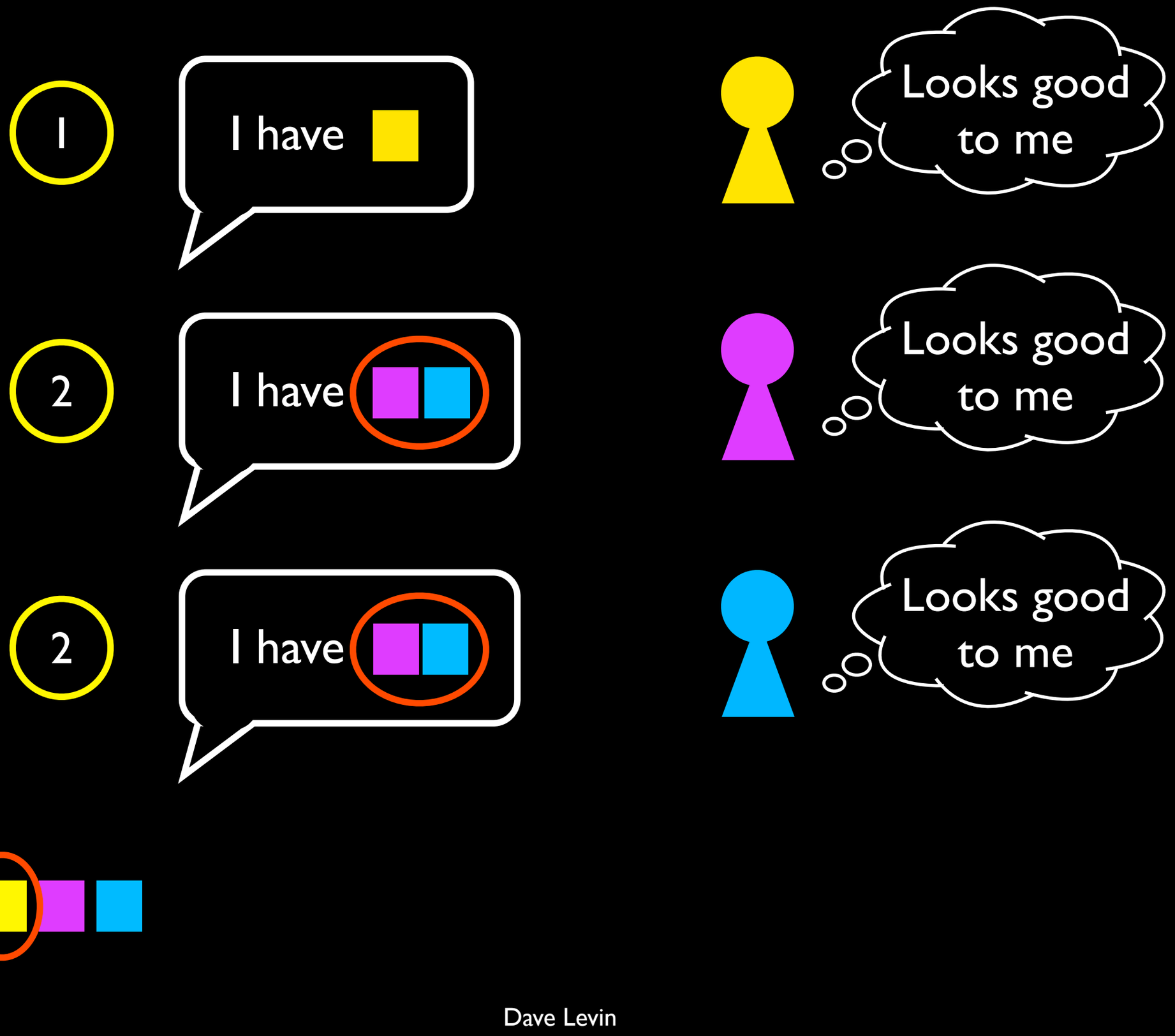
Why attest to the latest piece?



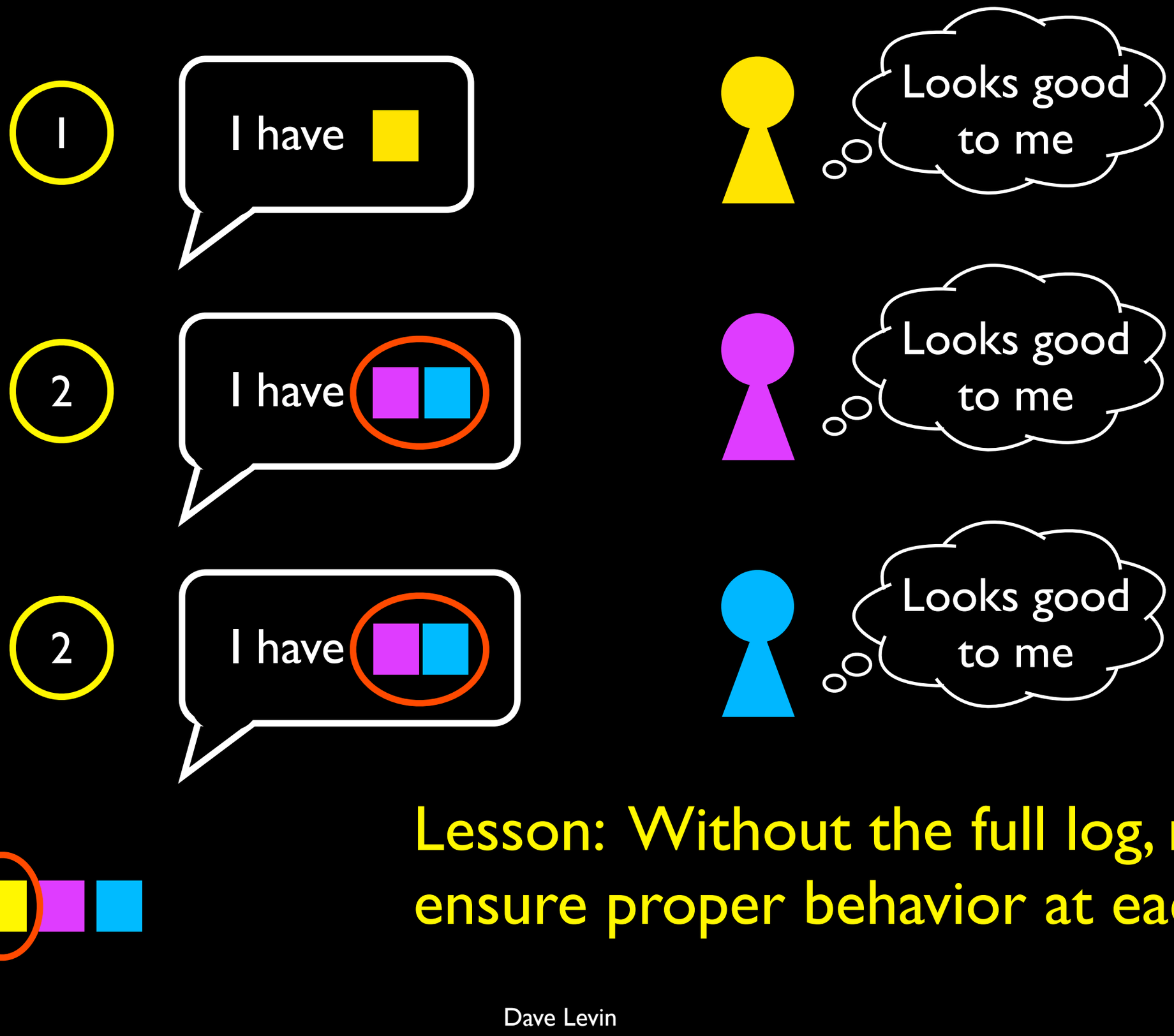
Why attest to the latest piece?



Why attest to the latest piece?



Why attest to the latest piece?



Lesson: Without the full log, must ensure proper behavior at each step

Macrobenchmarks

- TrInc-BitTorrent
  - Solves piece under-reporting
- TrInc-A2M
  - Reduces hardware requirements
  - Higher throughput
- TrInc-PeerReview
  - Reduces the communication necessary to achieve fault detection

Contributions

1. TrInc – A new, practical primitive for eliminating equivocation

2. Applications of TrInc

3. Implementation in currently available hardware

Contributions

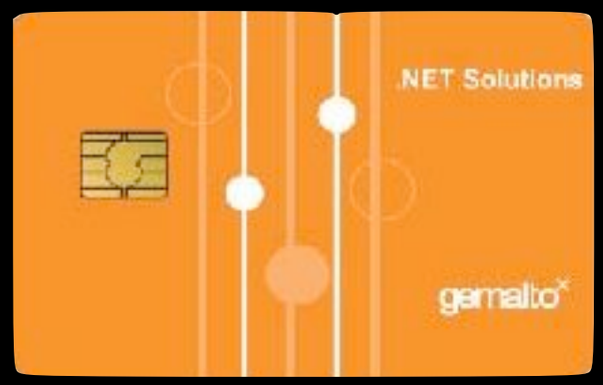
1. TrInc – A new, practical primitive for eliminating equivocation

2. Applications of TrInc

3. Implementation in currently available hardware

Implementation

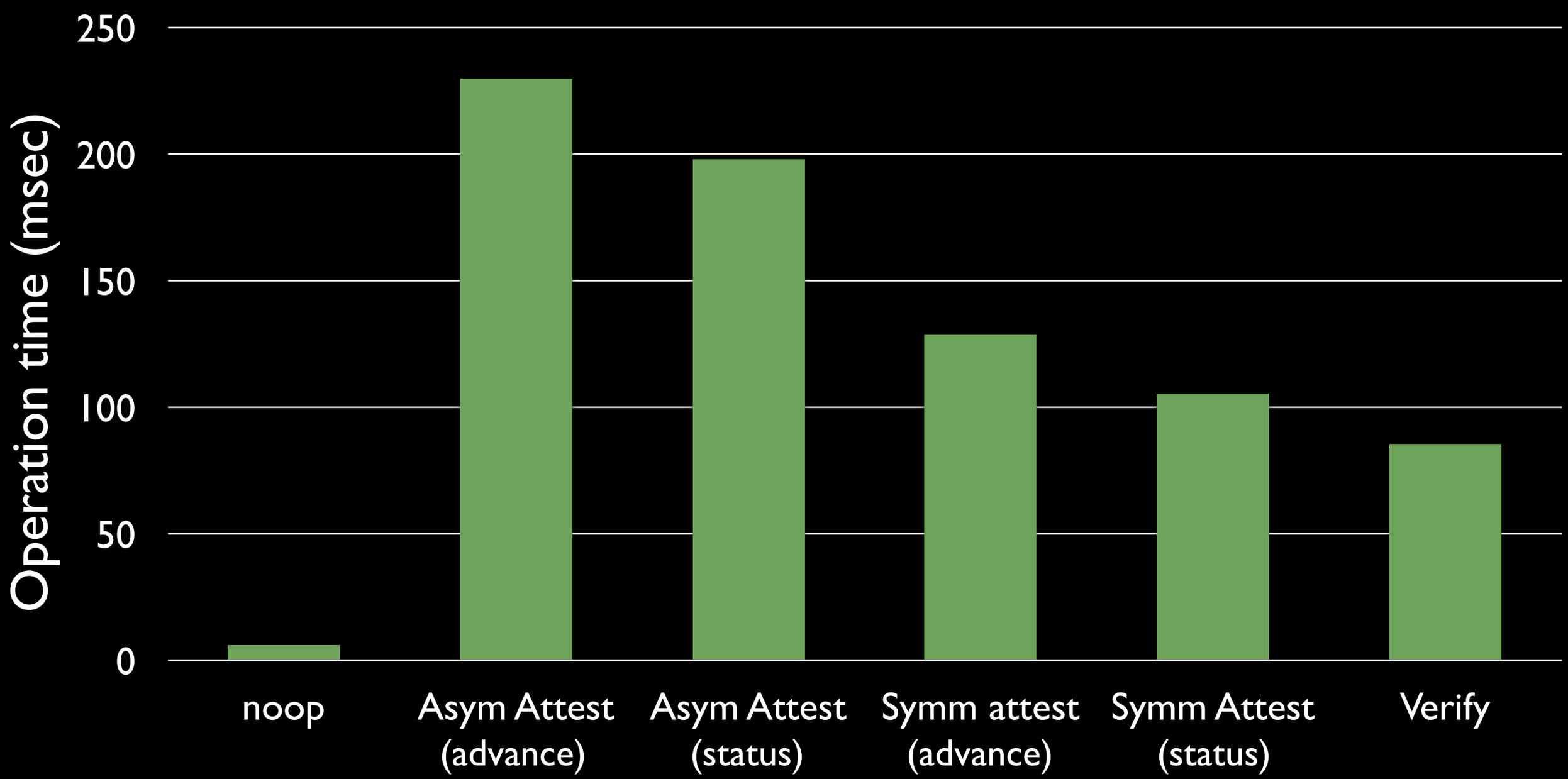
- Gemalto .NET Smartcard
  - Crypto unit (RSA & 3-DES)
  - 32-bit micro-controller
  - 80 KB persistent memory



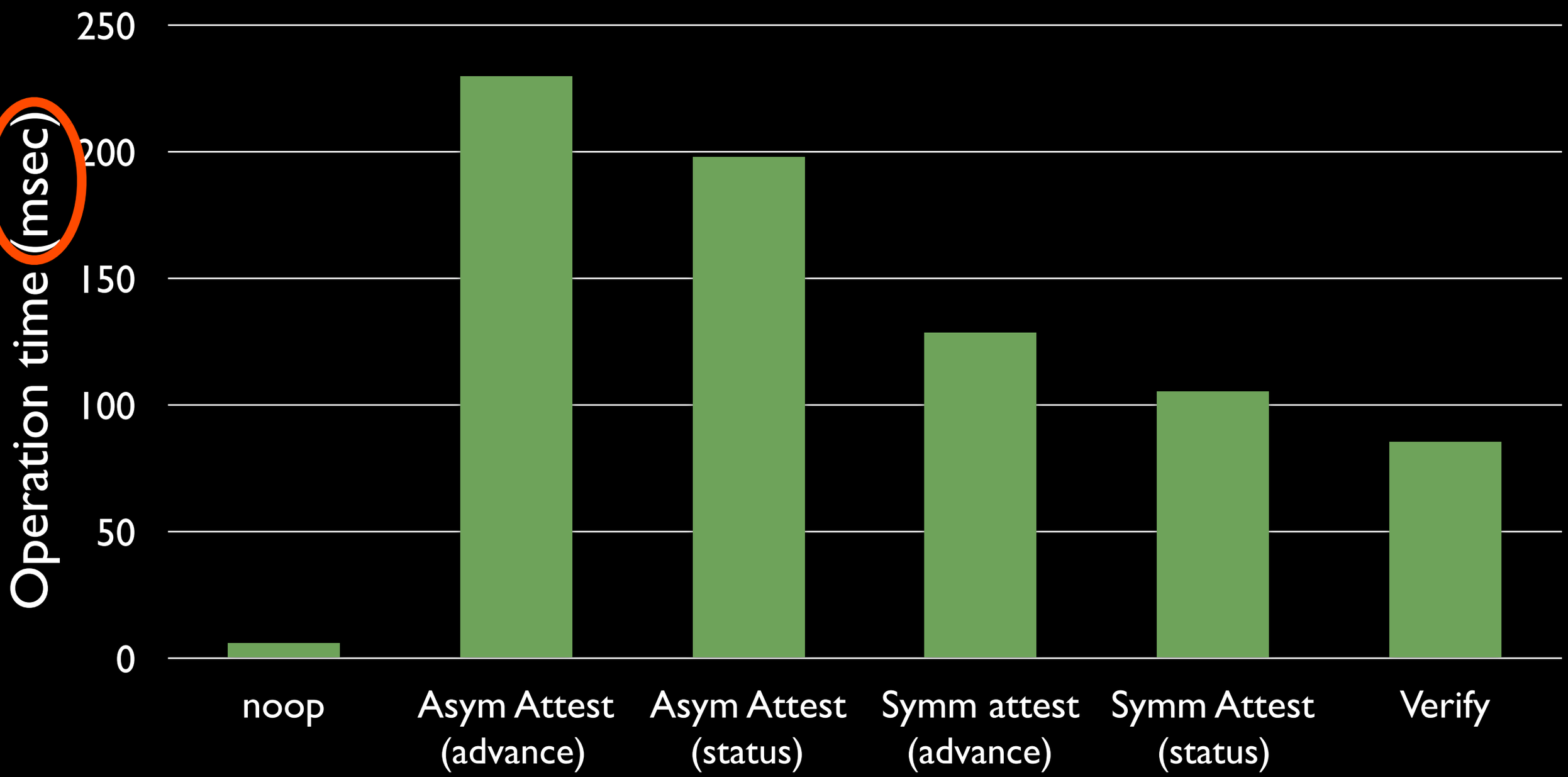
- A few dozen lines of C#

- Case studies
  - TrInc-A2M
  - TrInc-PeerReview
  - TrInc-BitTorrent

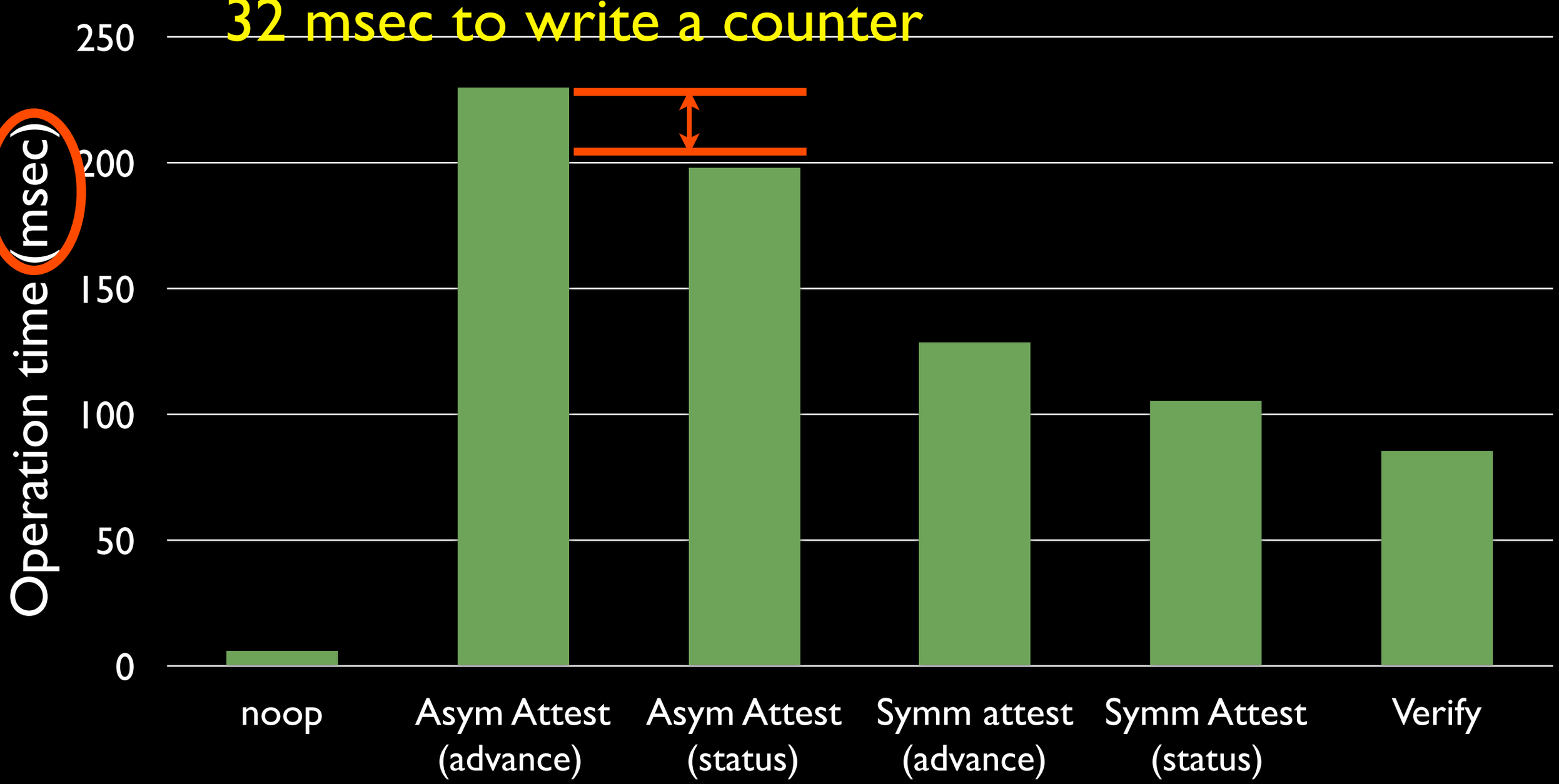
TrInc microbenchmarks



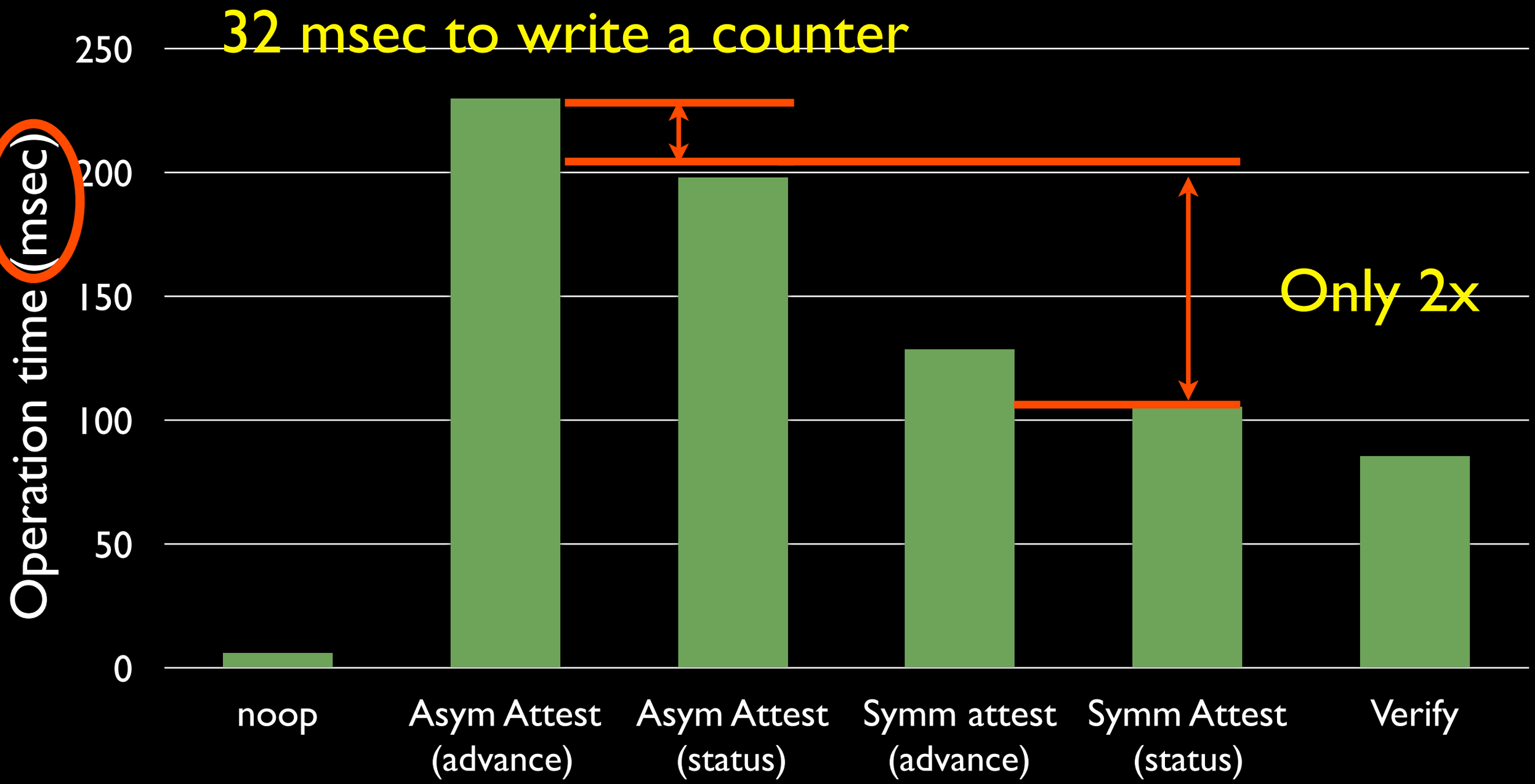
TrInc microbenchmarks



TrInc microbenchmarks



TrInc microbenchmarks



Why so slow?

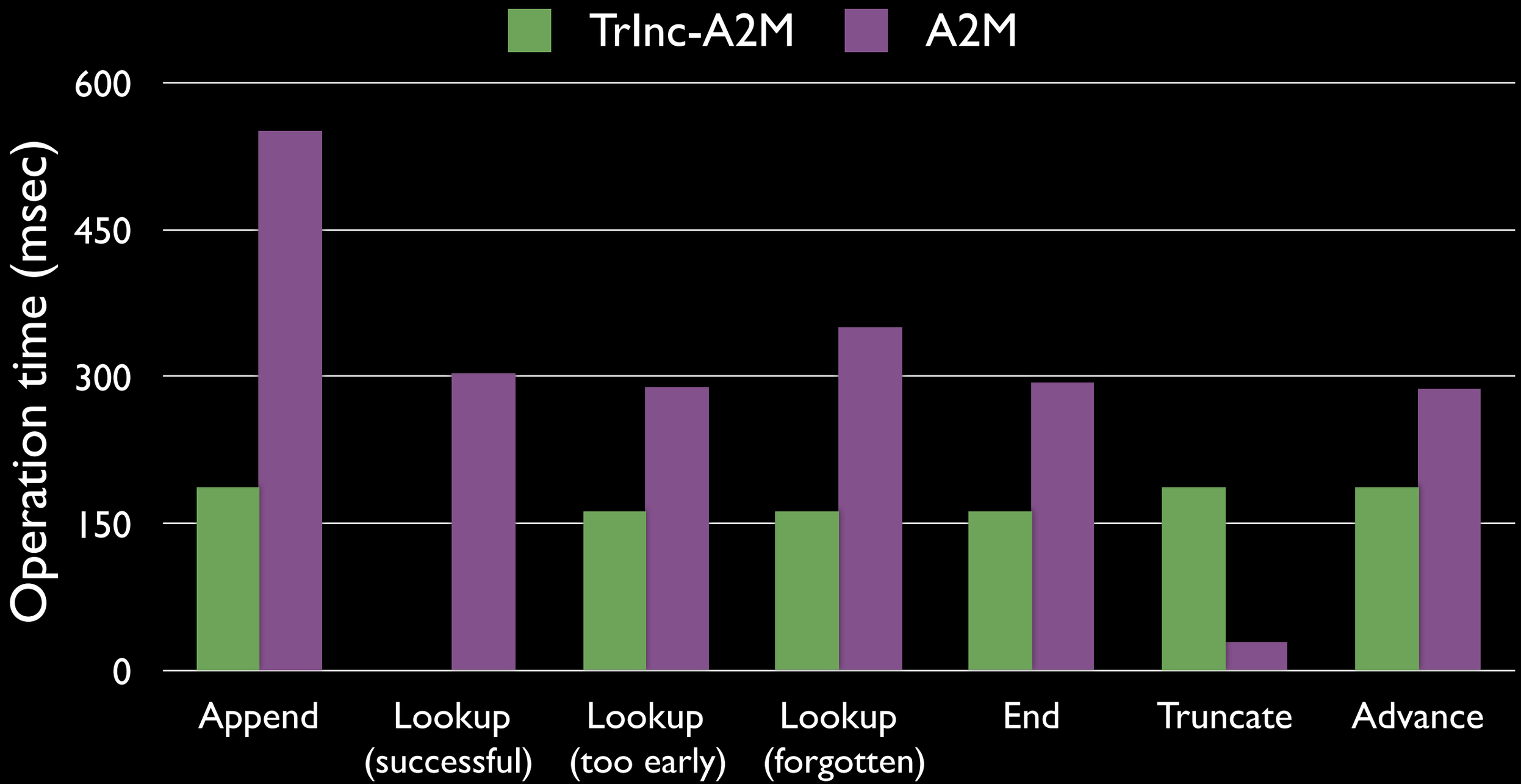
- Fundamentally new application of trusted hardware
  - Typically used for bootstrapping
  - TrInc makes it intrinsic to the protocol

- It can be faster
  - There just has not been the call for it prior to TrInc

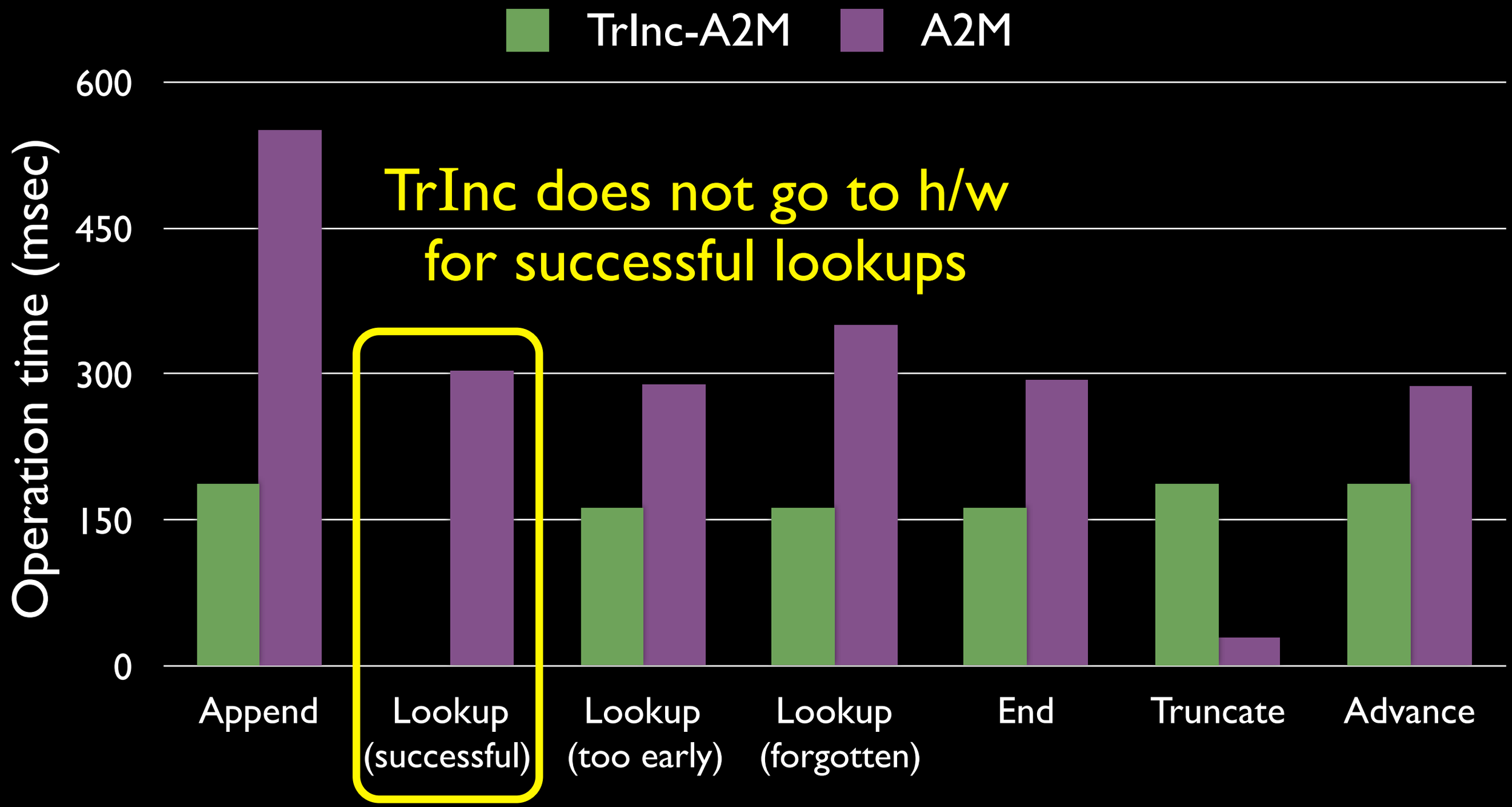
Summary

- Equivocation is a versatile and powerful
- A small amount of trust can secure a large system
- TrInc is
  - Minimal – A counter and a key
  - Versatile – Applies to a wide range of systems
  - Practical – Uses the same components available today

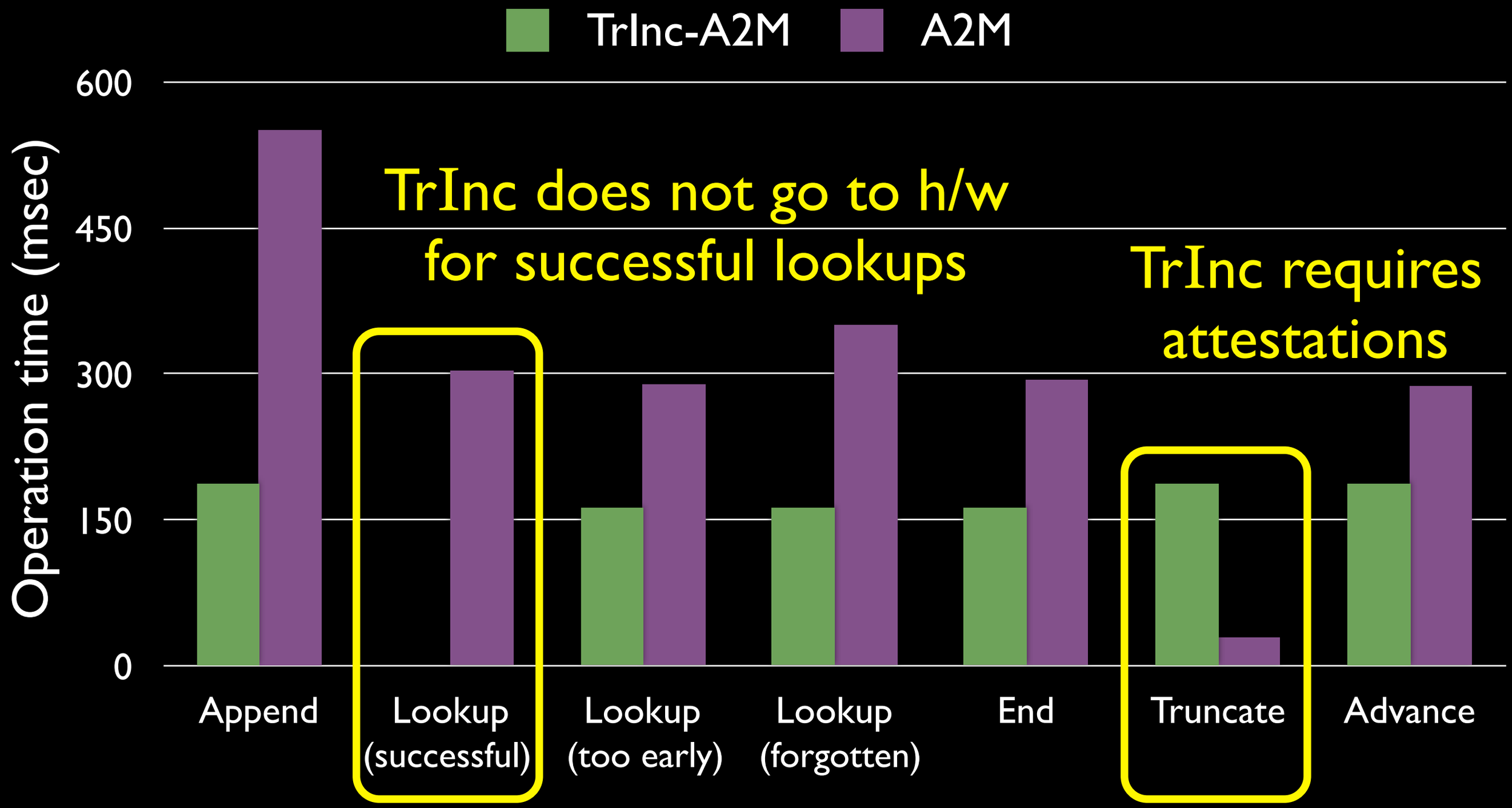
TrInc speeds up A2M



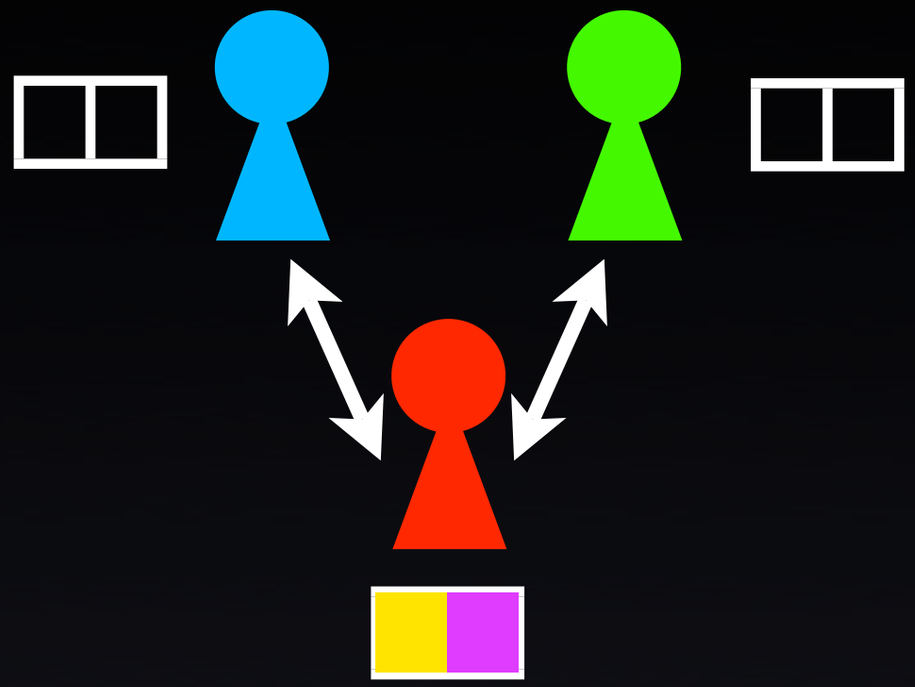
TrInc speeds up A2M



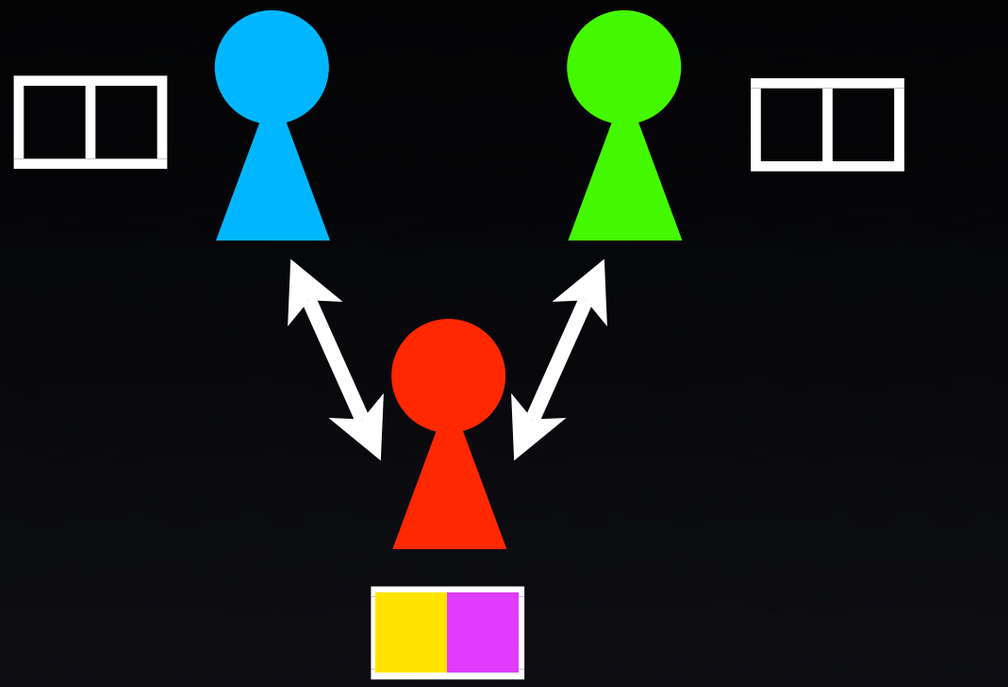
TrInc speeds up A2M

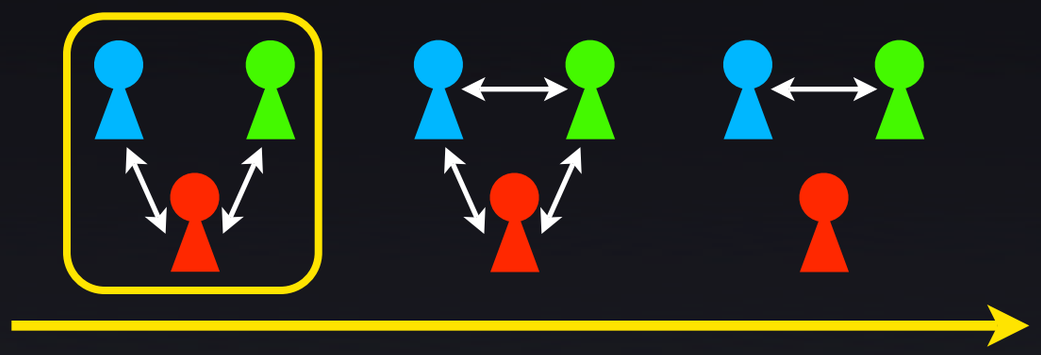


Block Revelation

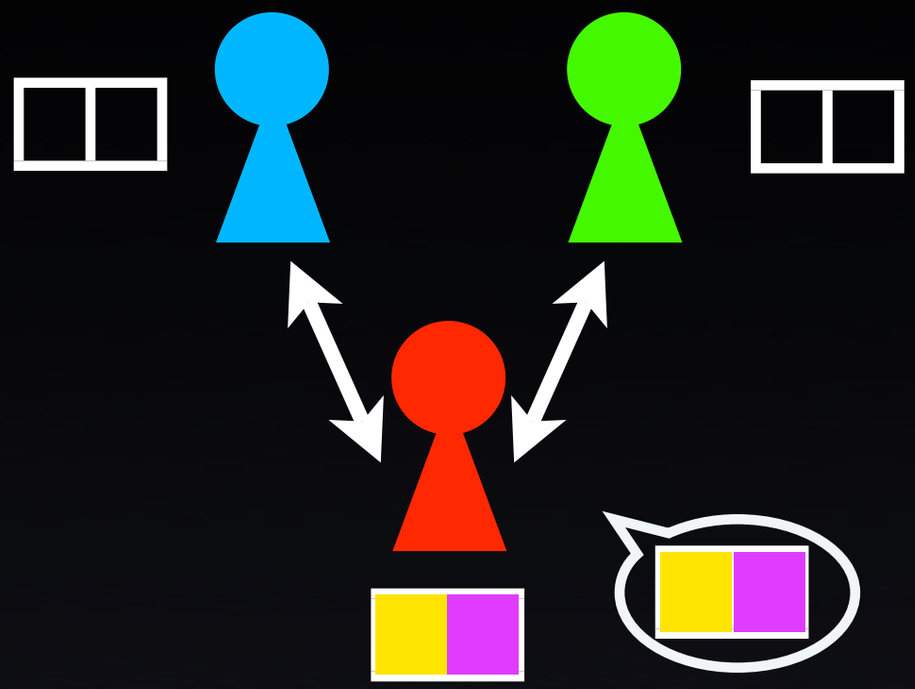


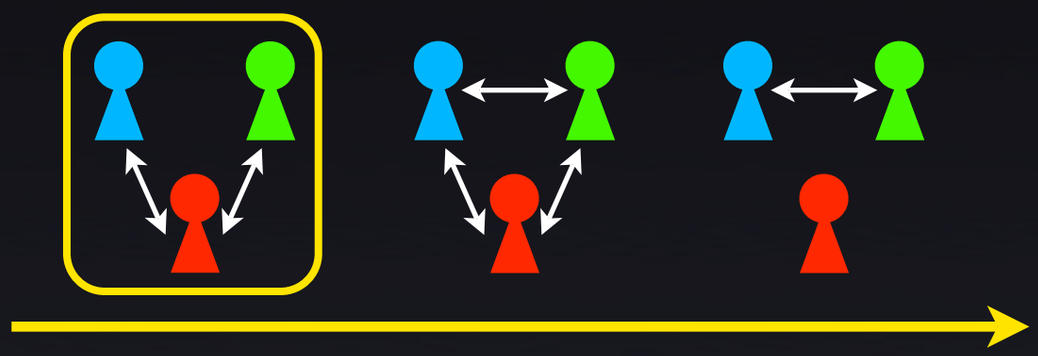
Block Revelation



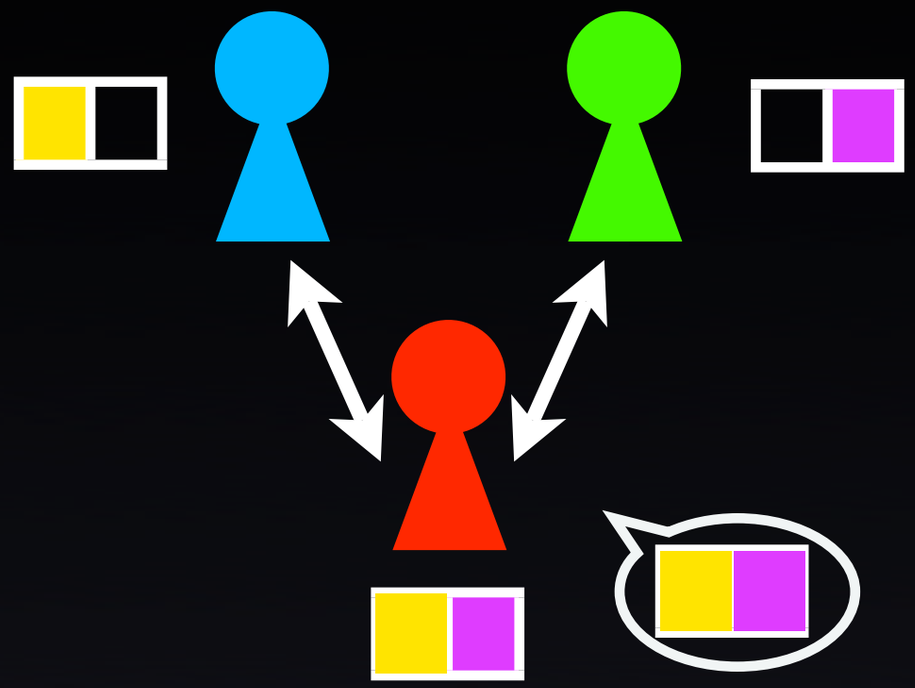


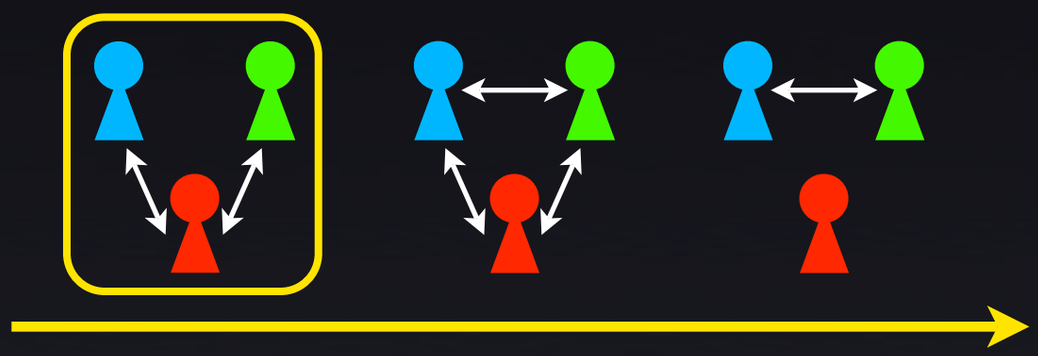
Block Revelation



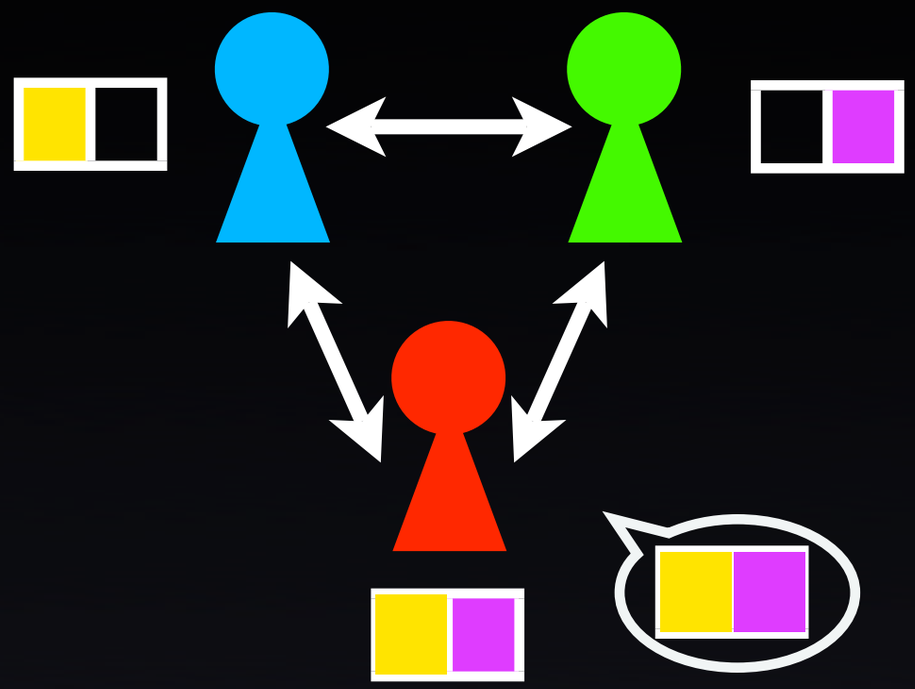


Block Revelation

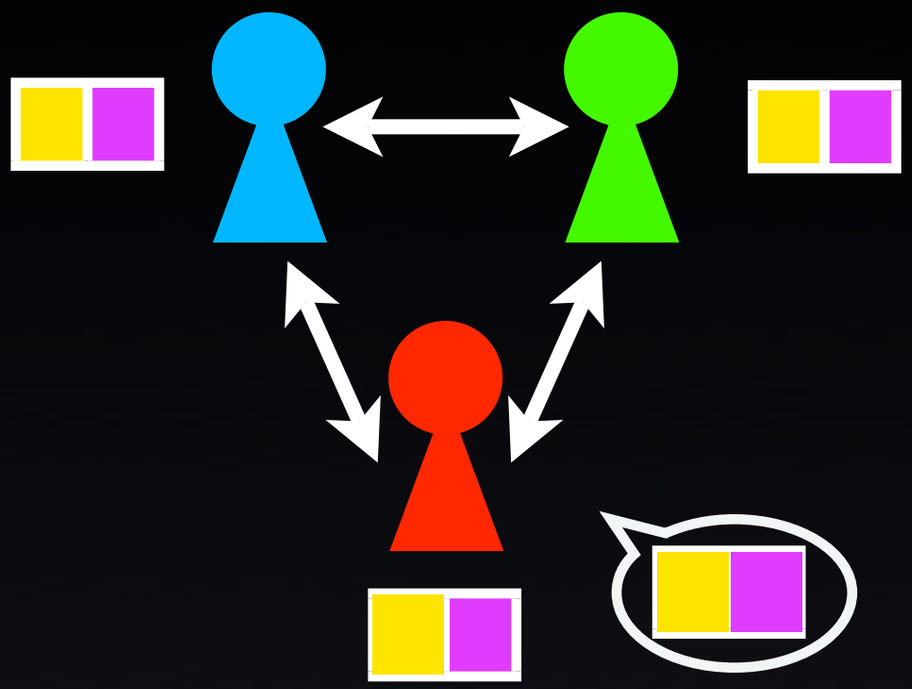




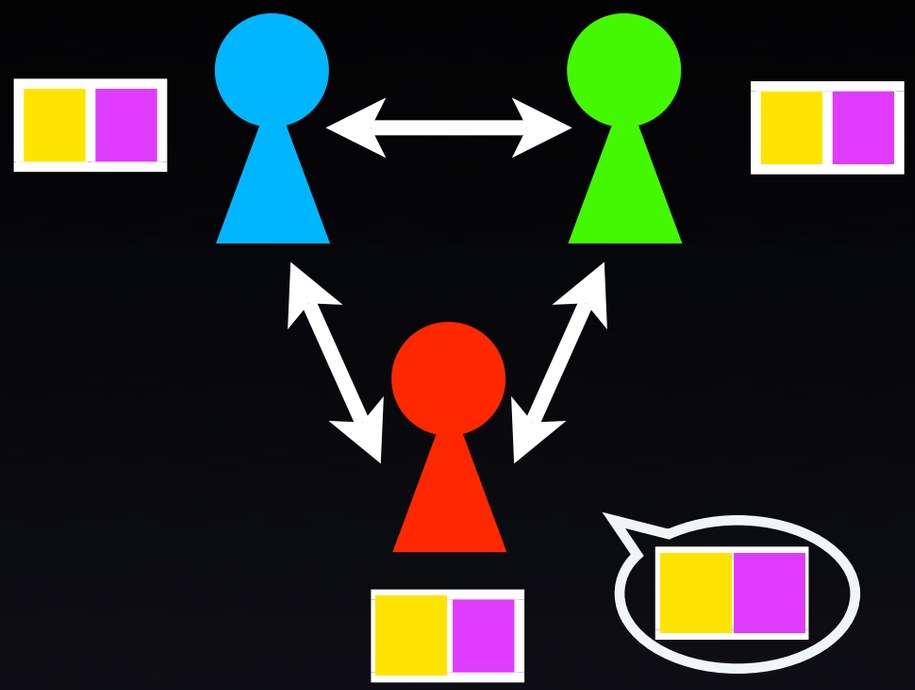
Block Revelation

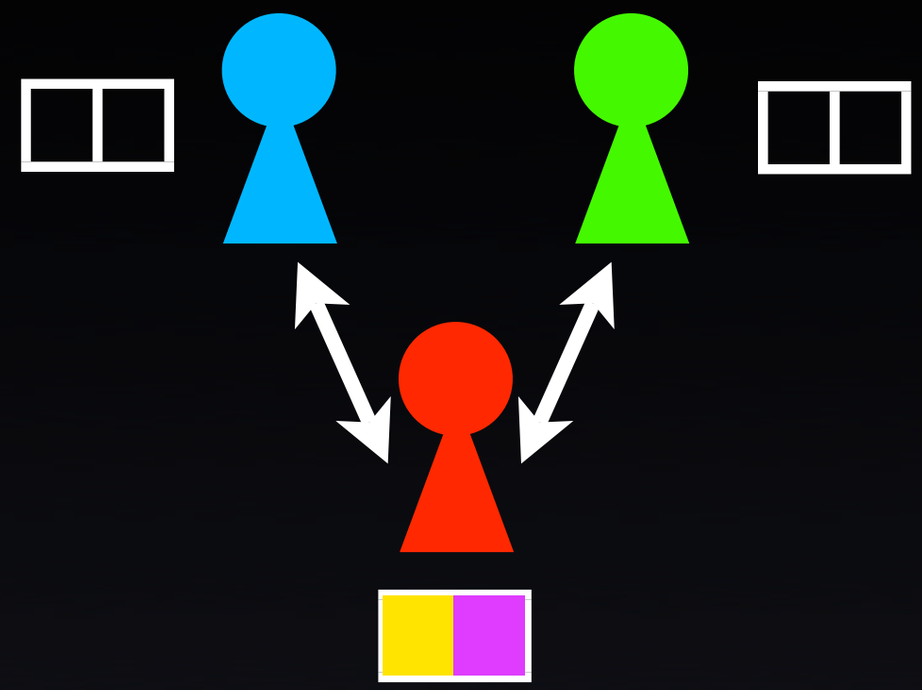


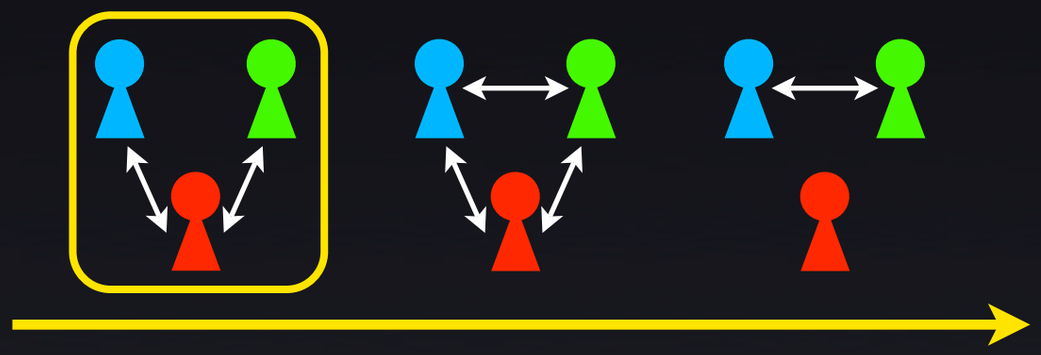
Block Revelation



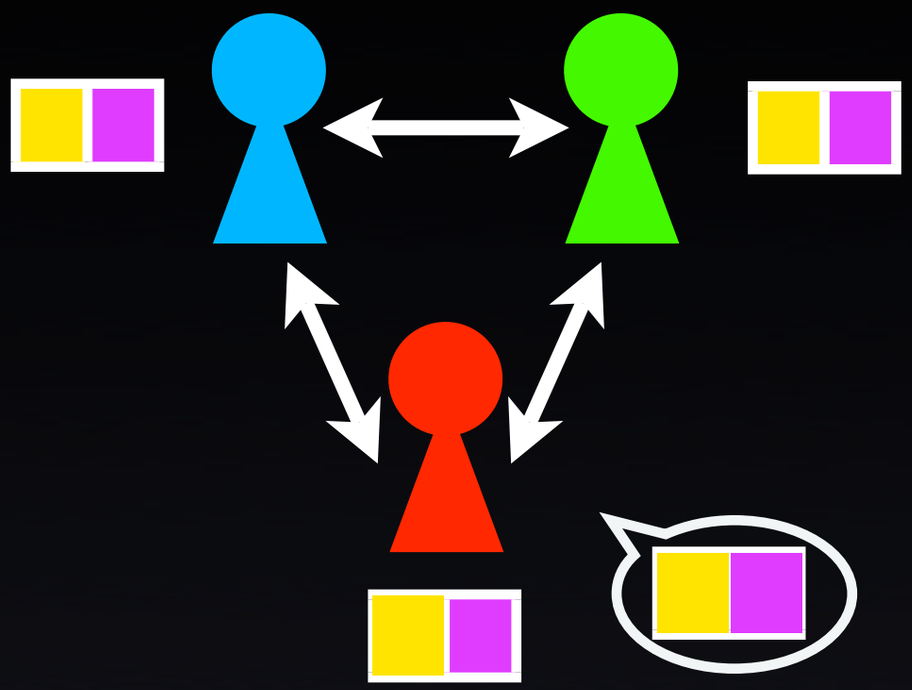
Block Revelation

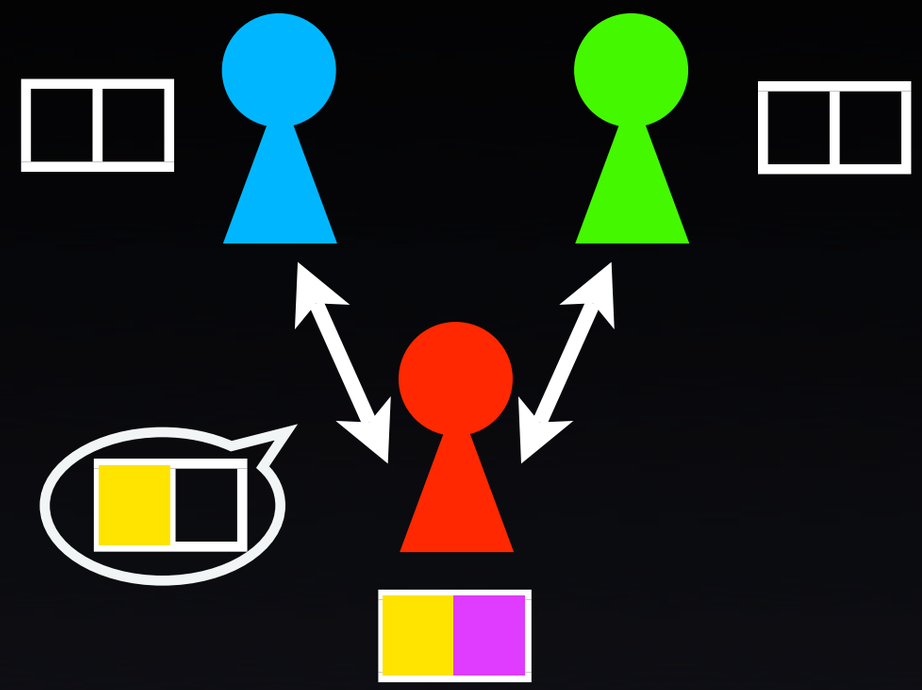


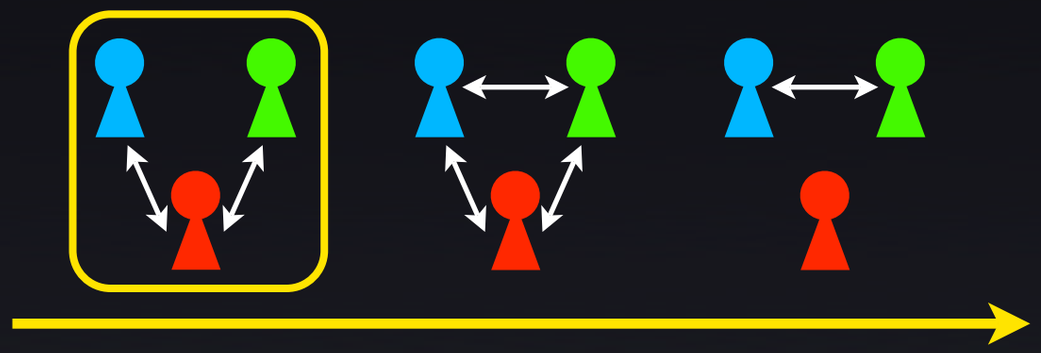




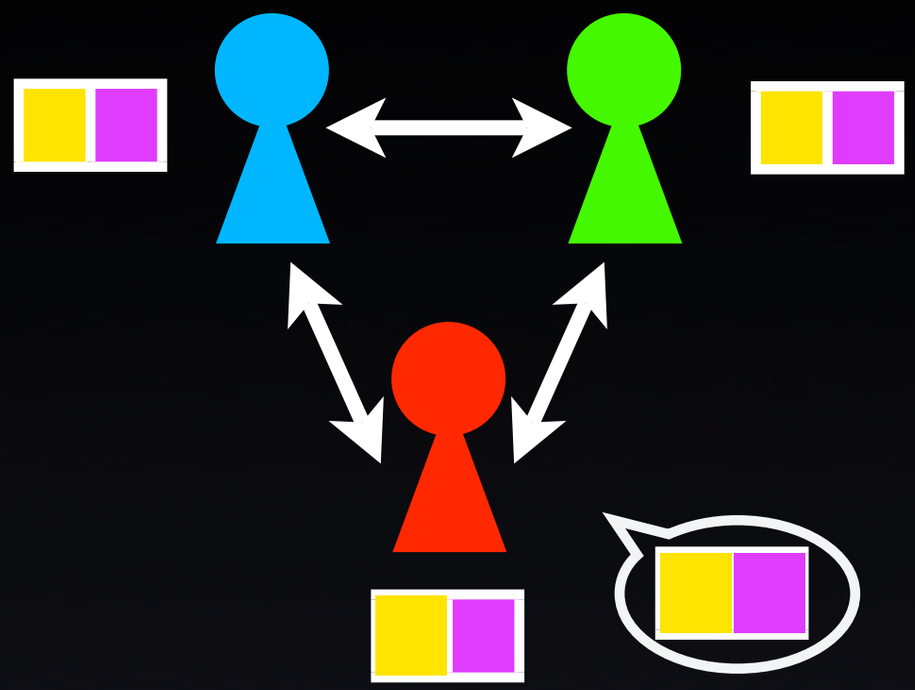
Block Revelation

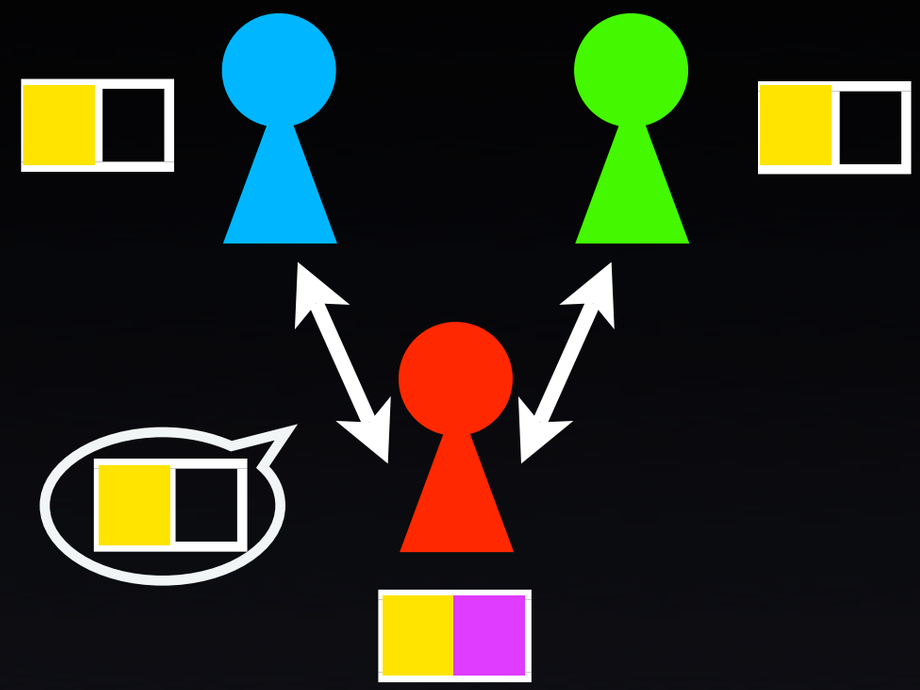


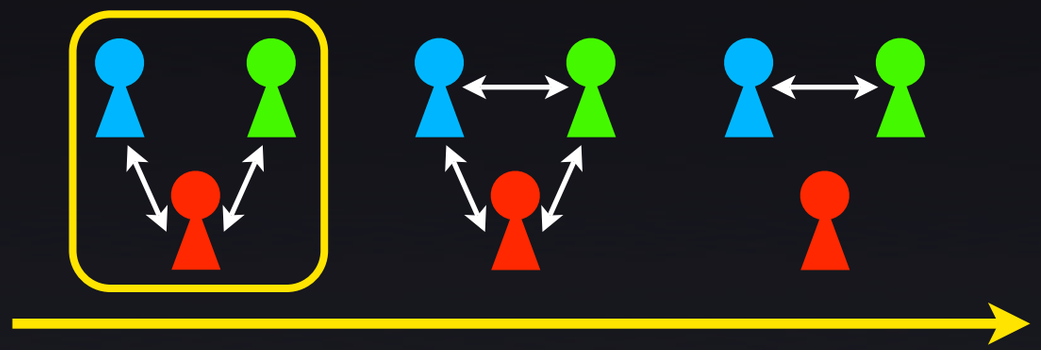




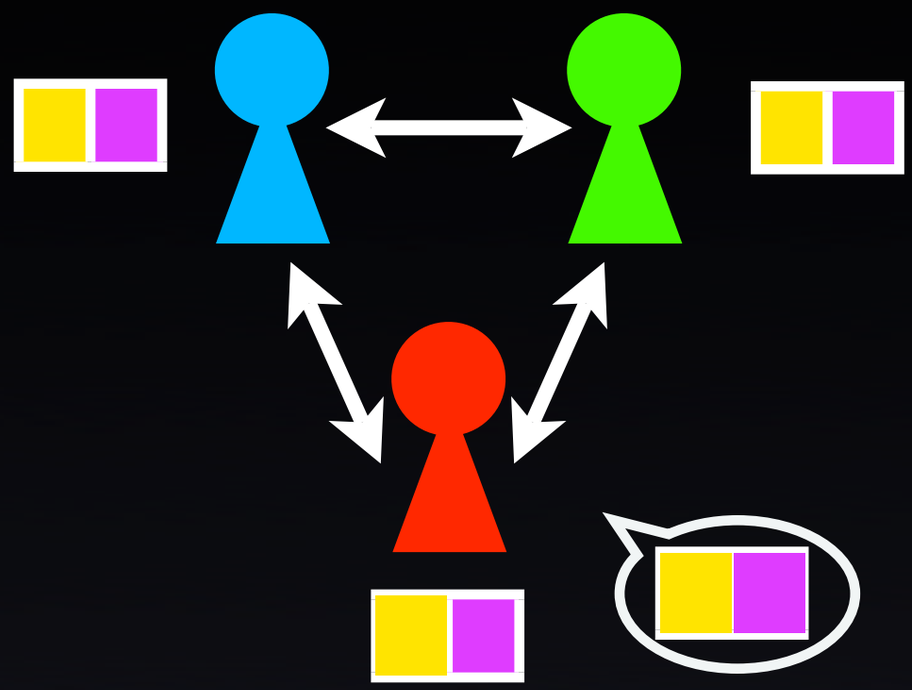
Block Revelation

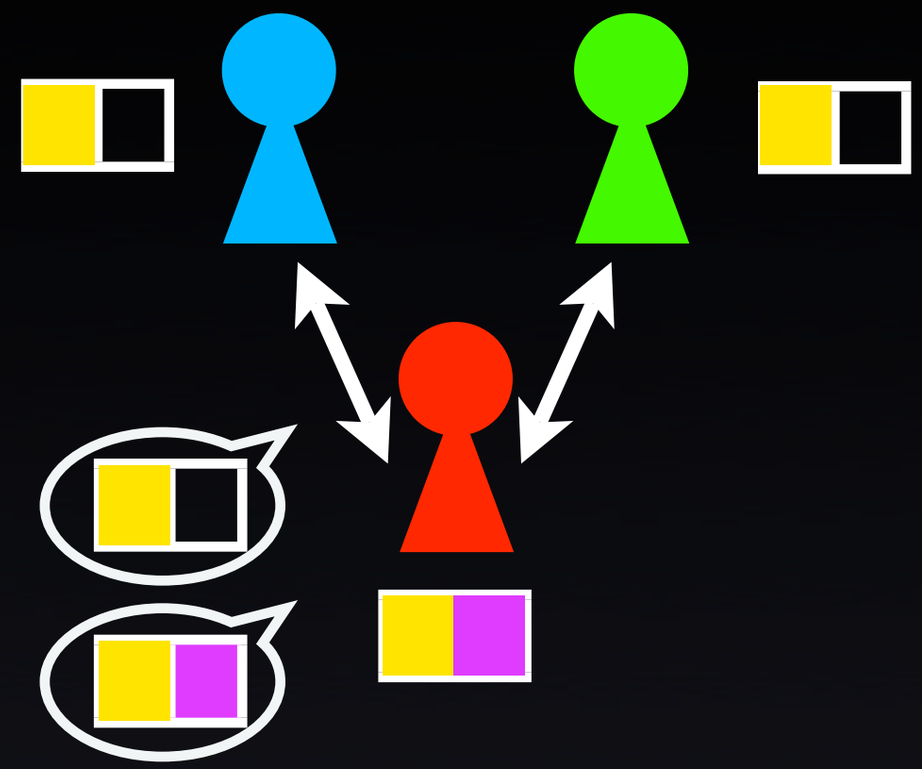


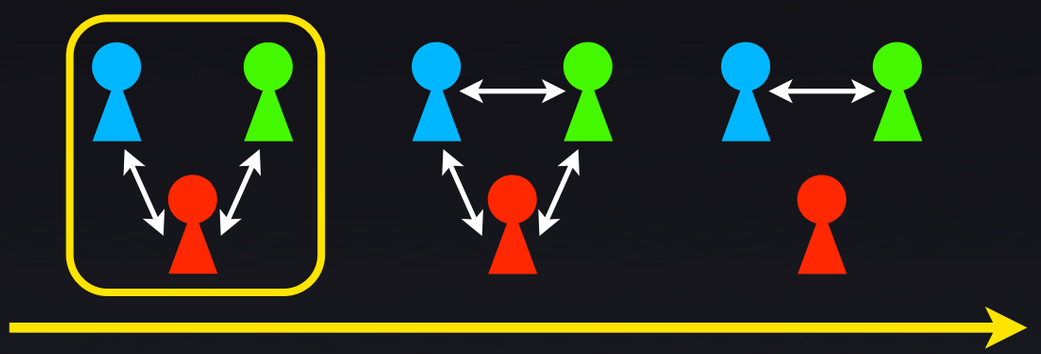




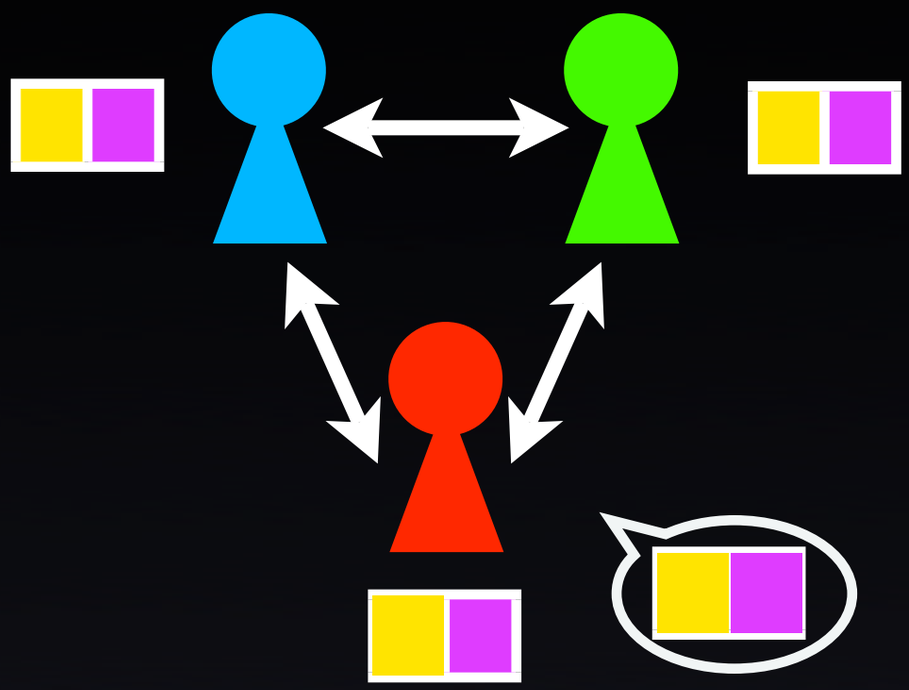
Block Revelation

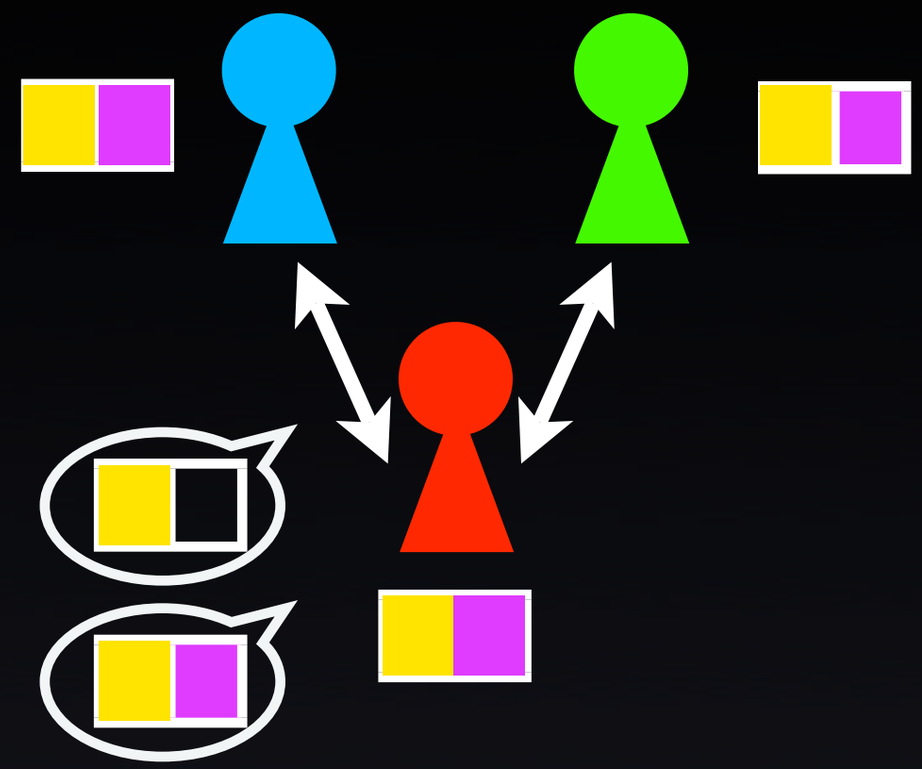


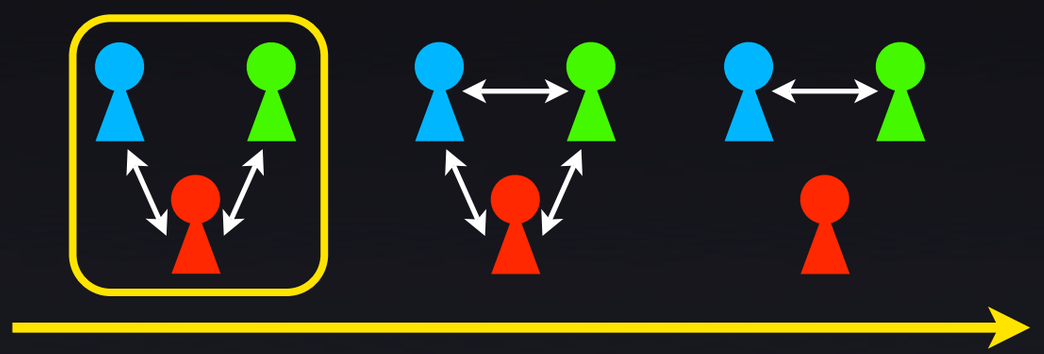




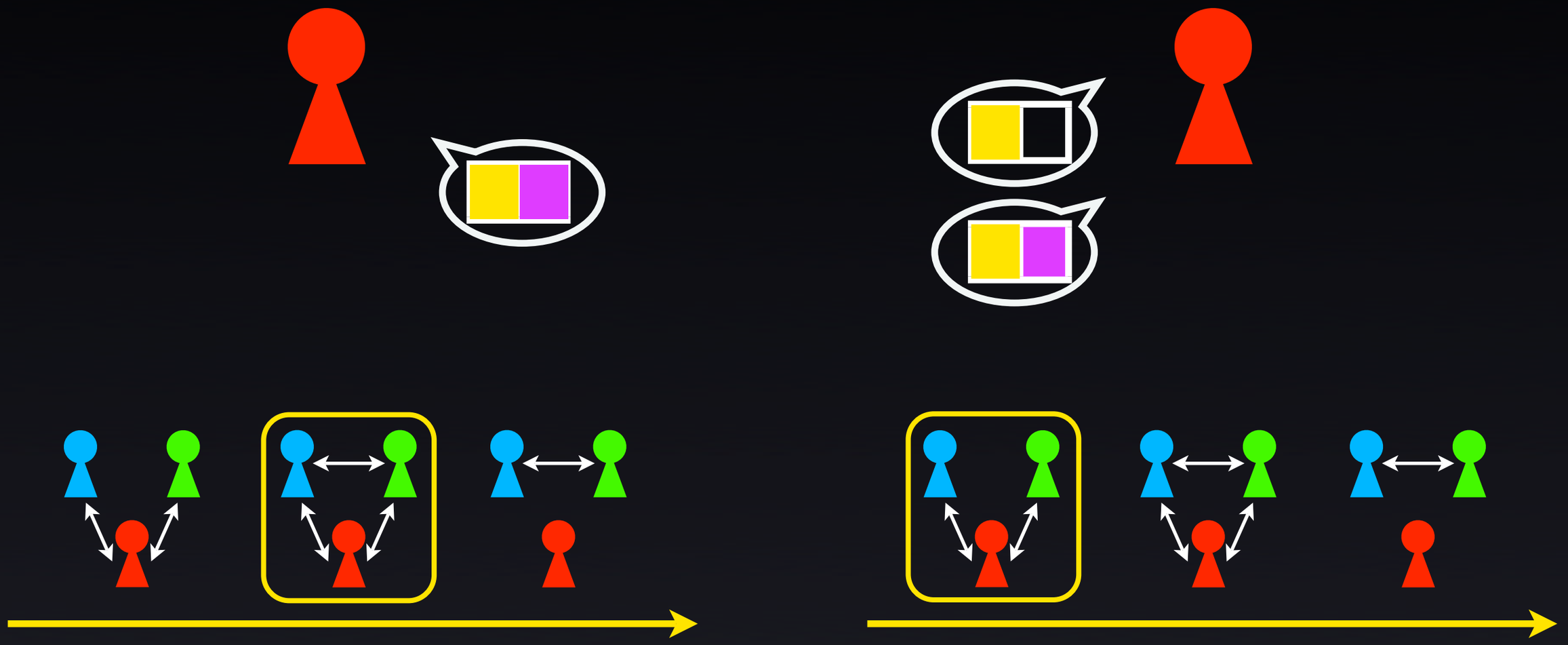
Block Revelation





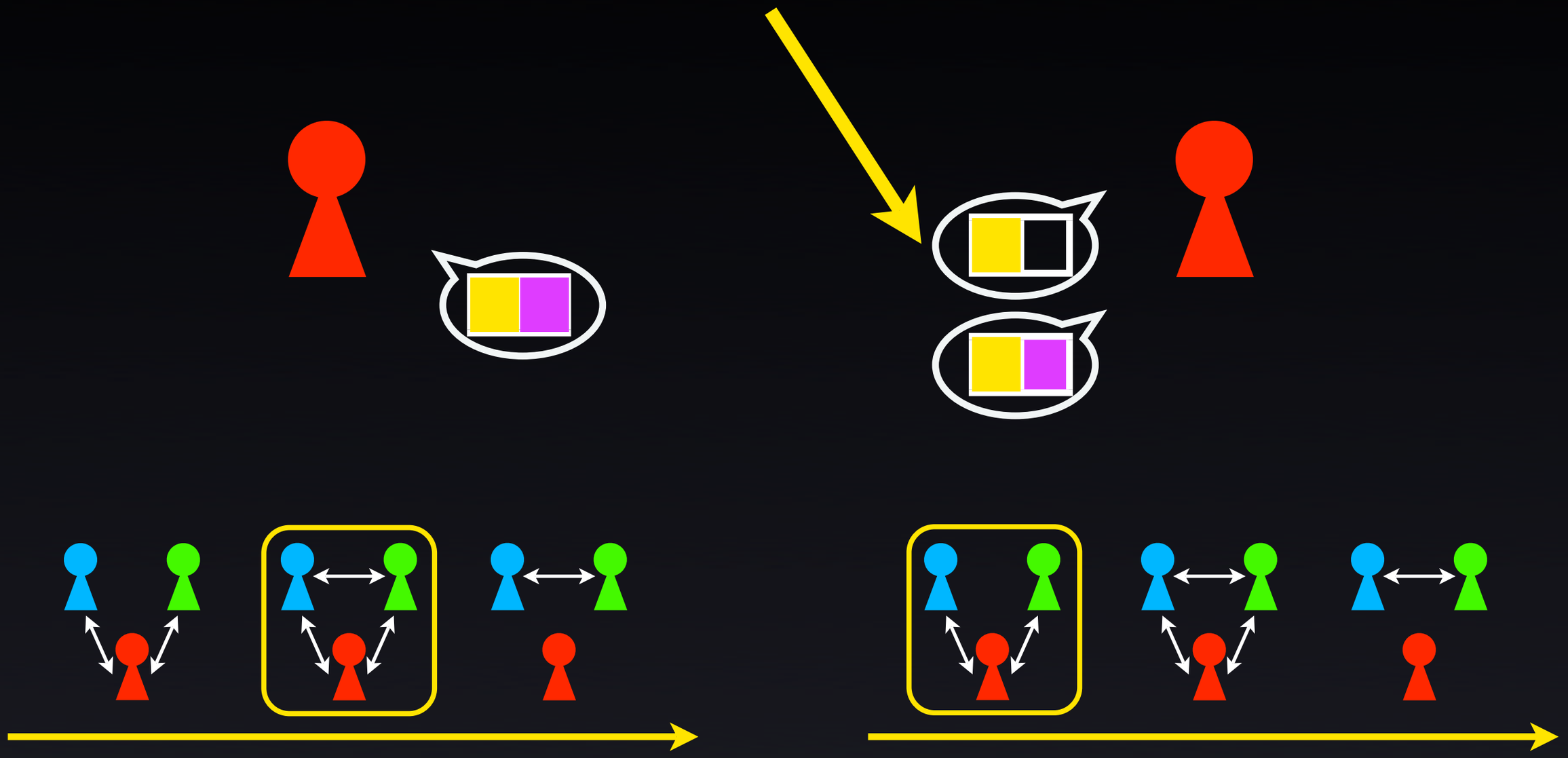


Block Revelation

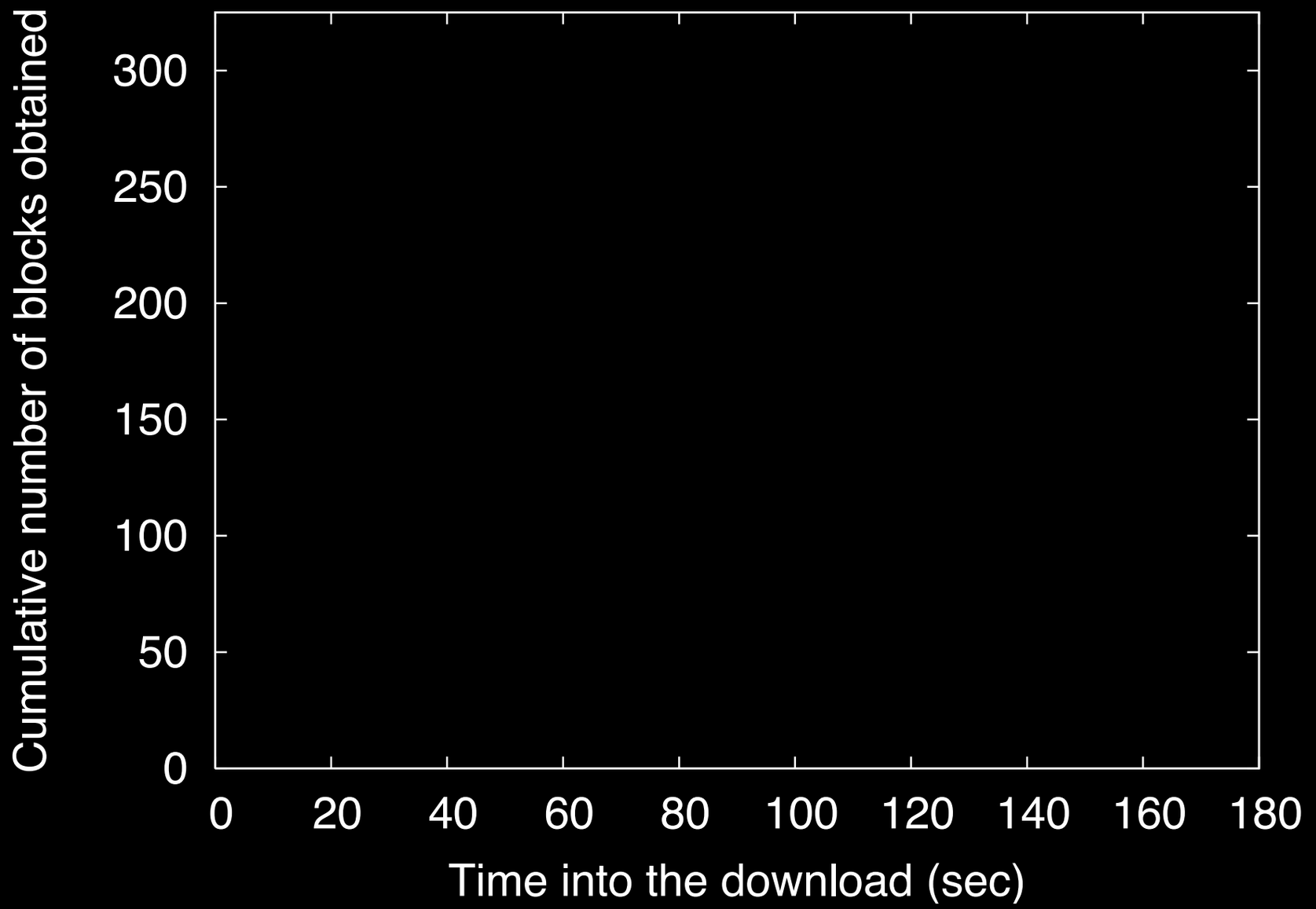


Block Revelation

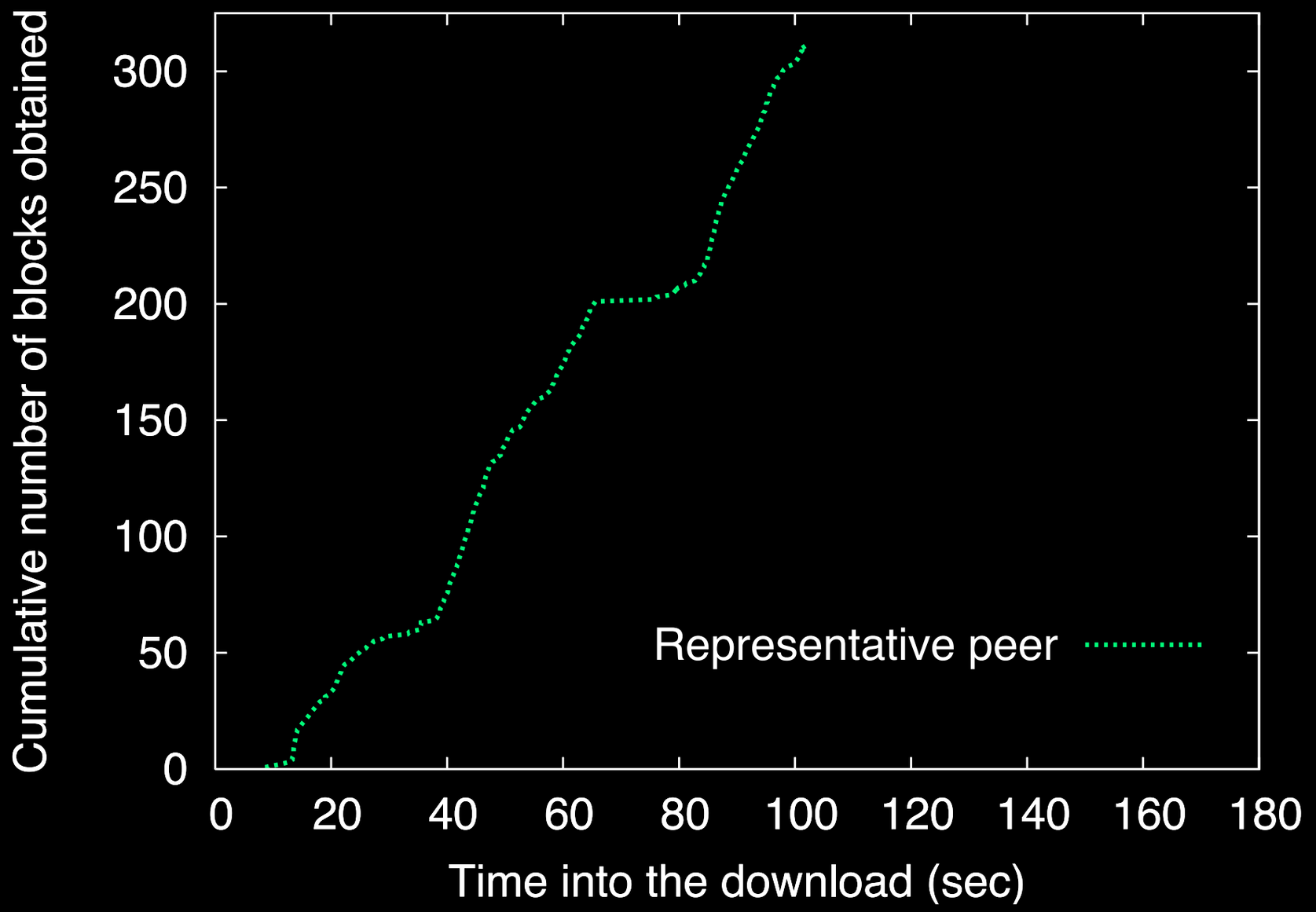
Strategically *under*-report



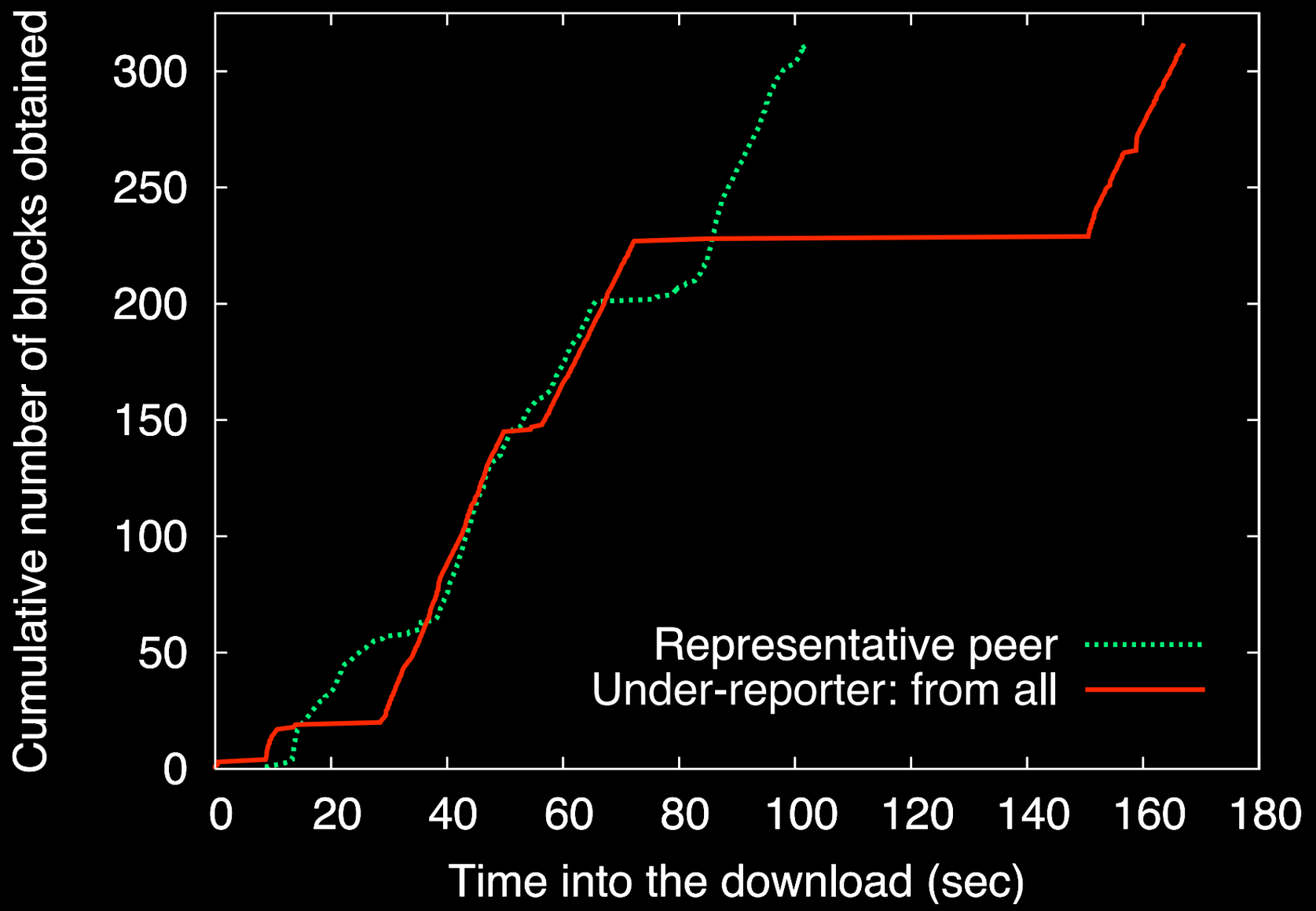
TrInc-BitTorrent Results



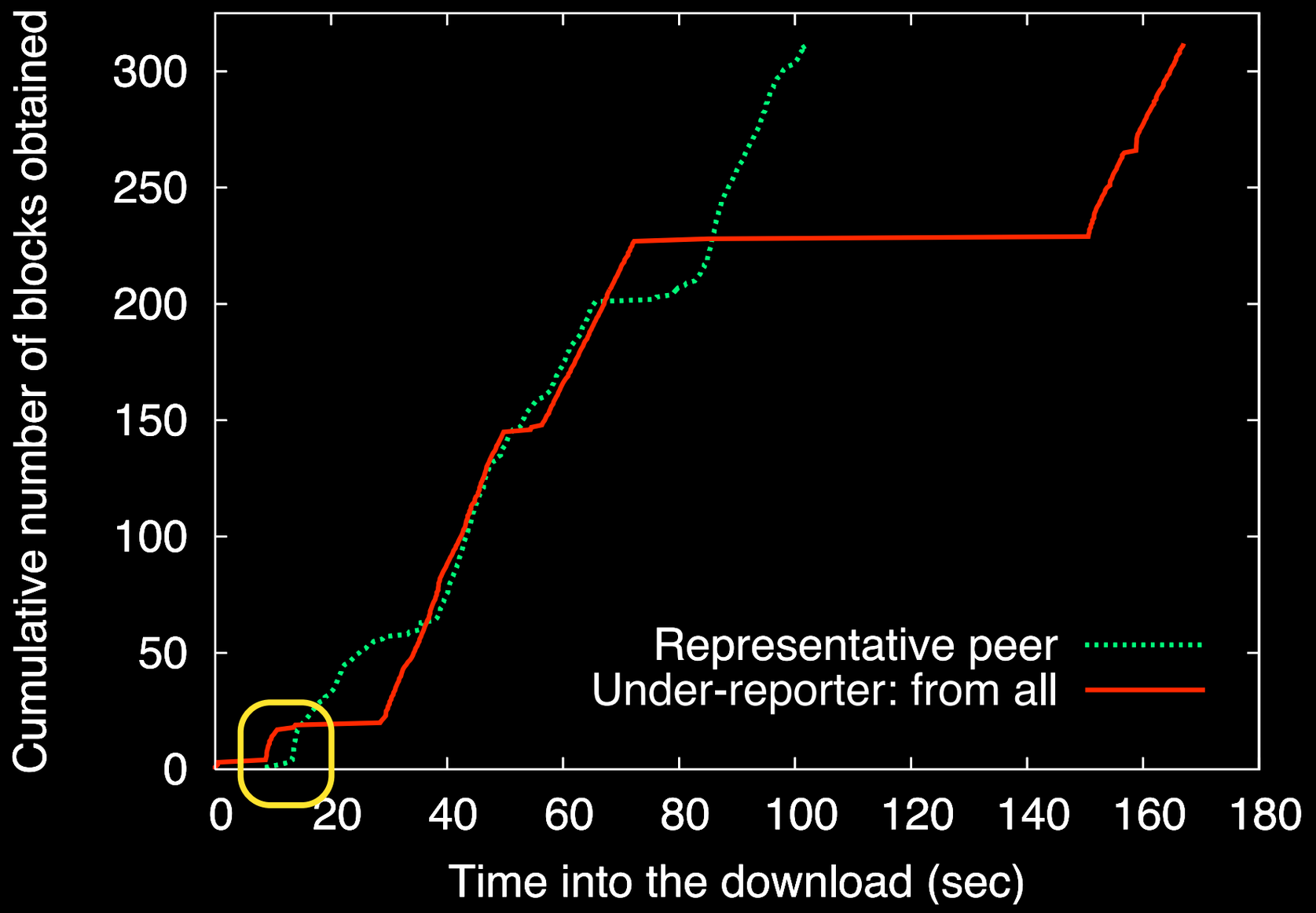
TrInc-BitTorrent Results



TrInc-BitTorrent Results



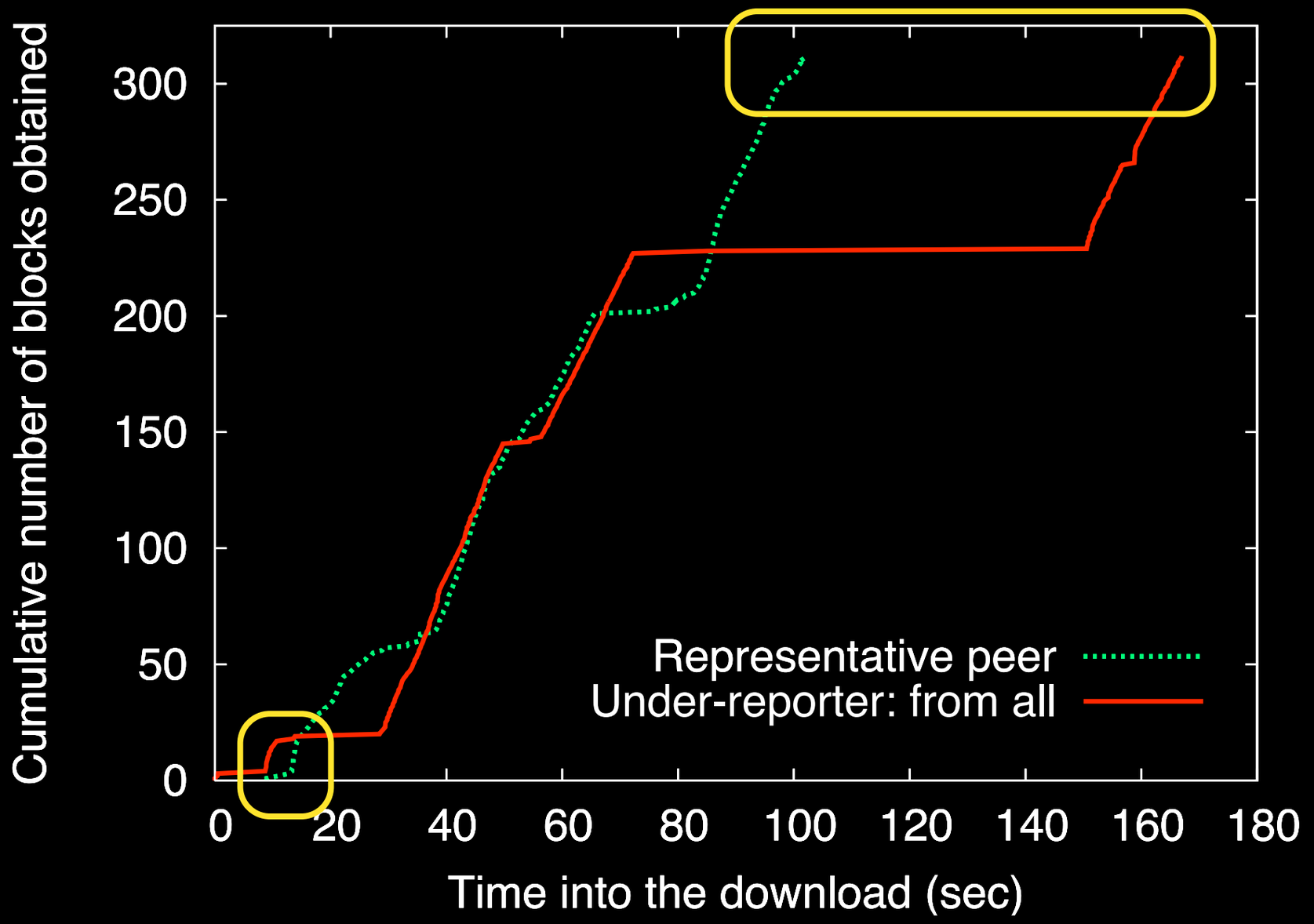
TrInc-BitTorrent Results



Under-reporter pulls ahead

TrInc-BitTorrent Results

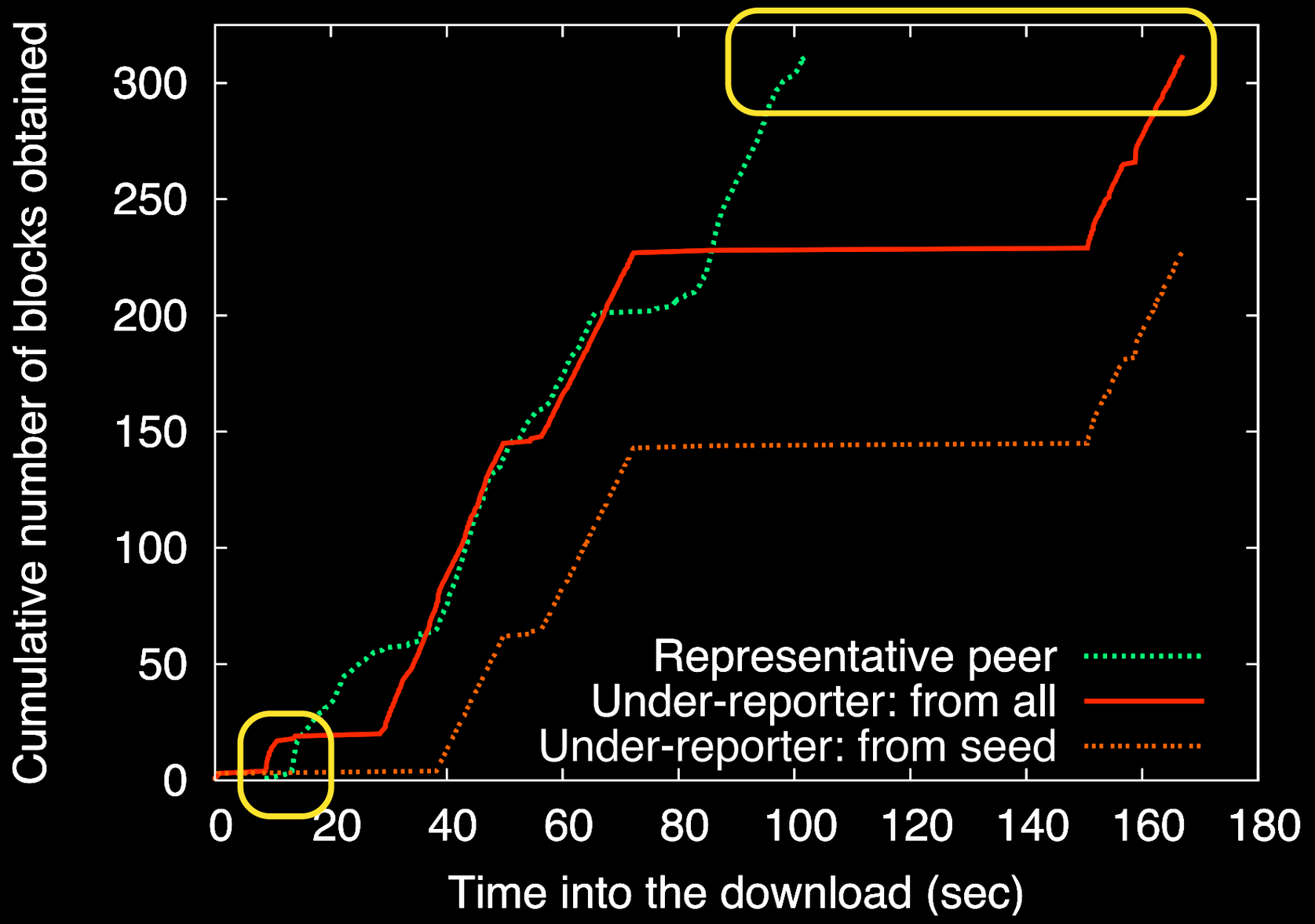
But ultimately downloads slower



Under-reporter pulls ahead

TrInc-BitTorrent Results

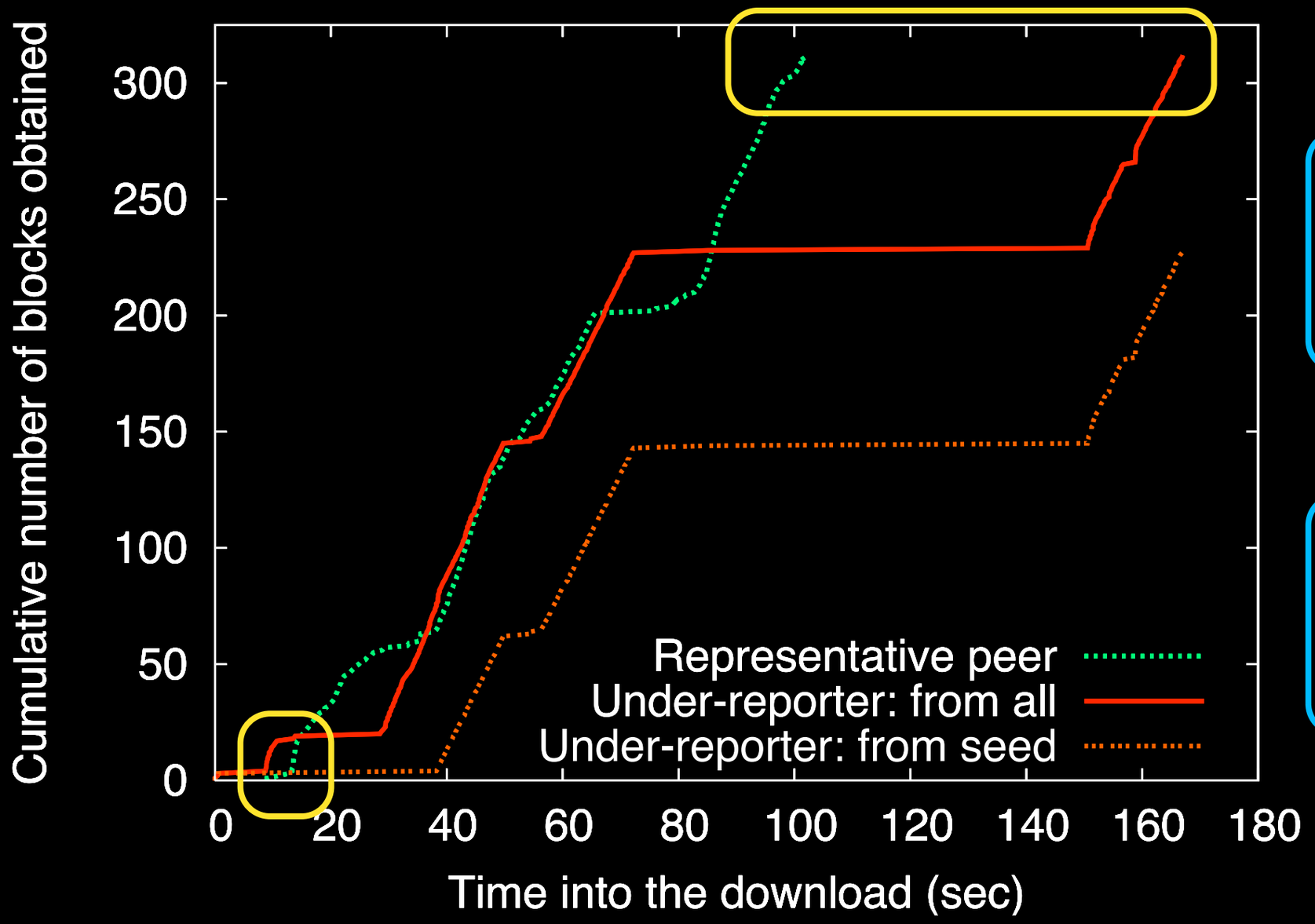
But ultimately downloads slower



Under-reporter pulls ahead

TrInc-BitTorrent Results

But ultimately downloads slower



Truth-tellers
A median of 6%
from the seeder

Under-reporter
73% of file
from the seeder

Under-reporter pulls ahead