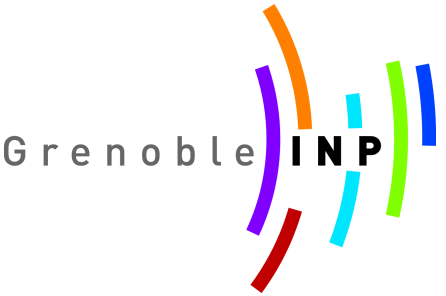
AcTinG:
Accurate Freerider Tracking in Gossip

Sonia Ben Mokhtar, Jérémie Decouchant, Vivien Quéma



Peer-To-Peer Content Sharing

Content sharing applications account for a large portion of traffic over the Internet.

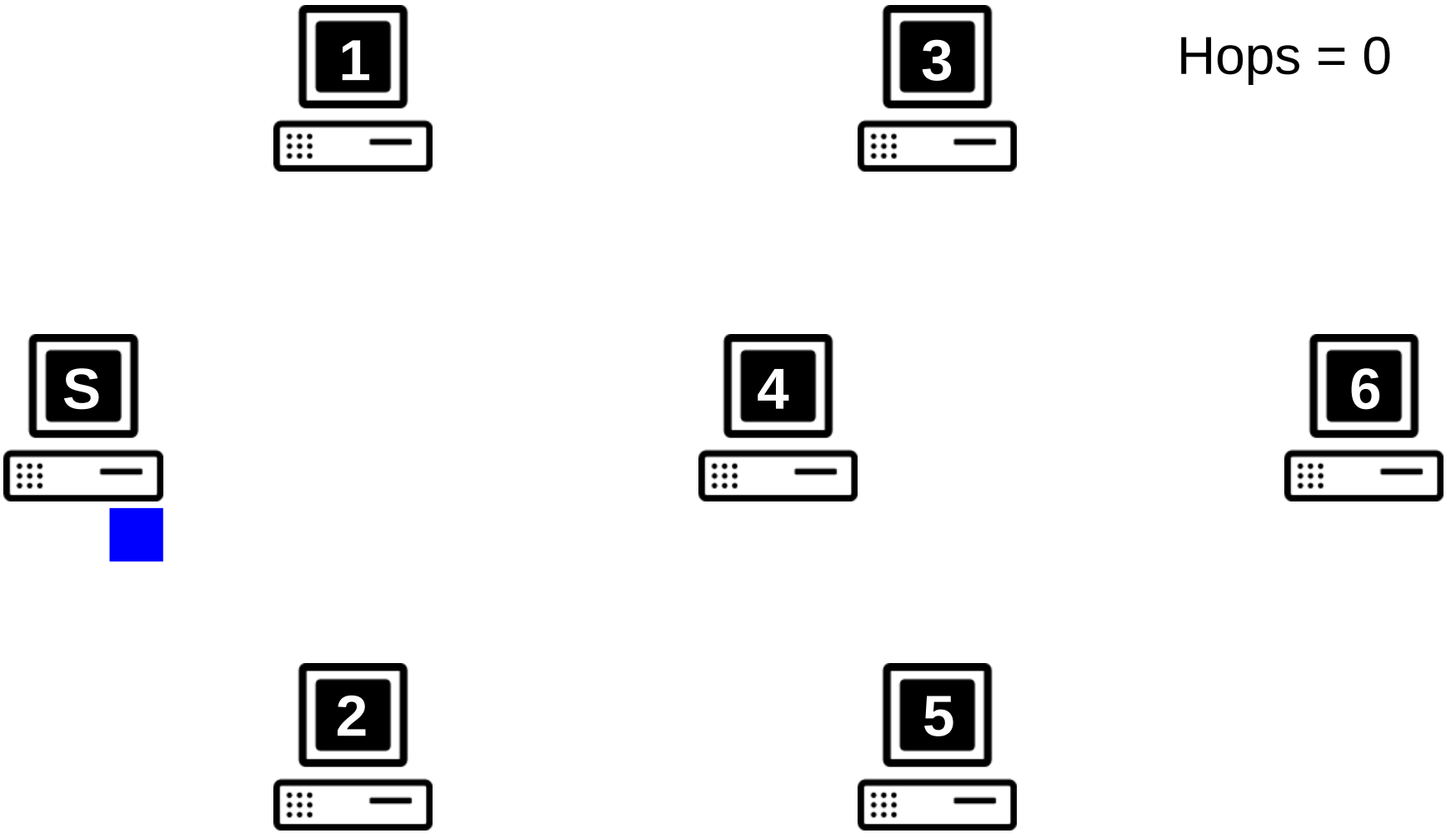
Relying on the P2P paradigm provides the following advantages :

- Robustness to failures, or churn
- Scalability
- Shifting cost (e.g., bandwidth) to clients
- No need to maintain dedicated servers

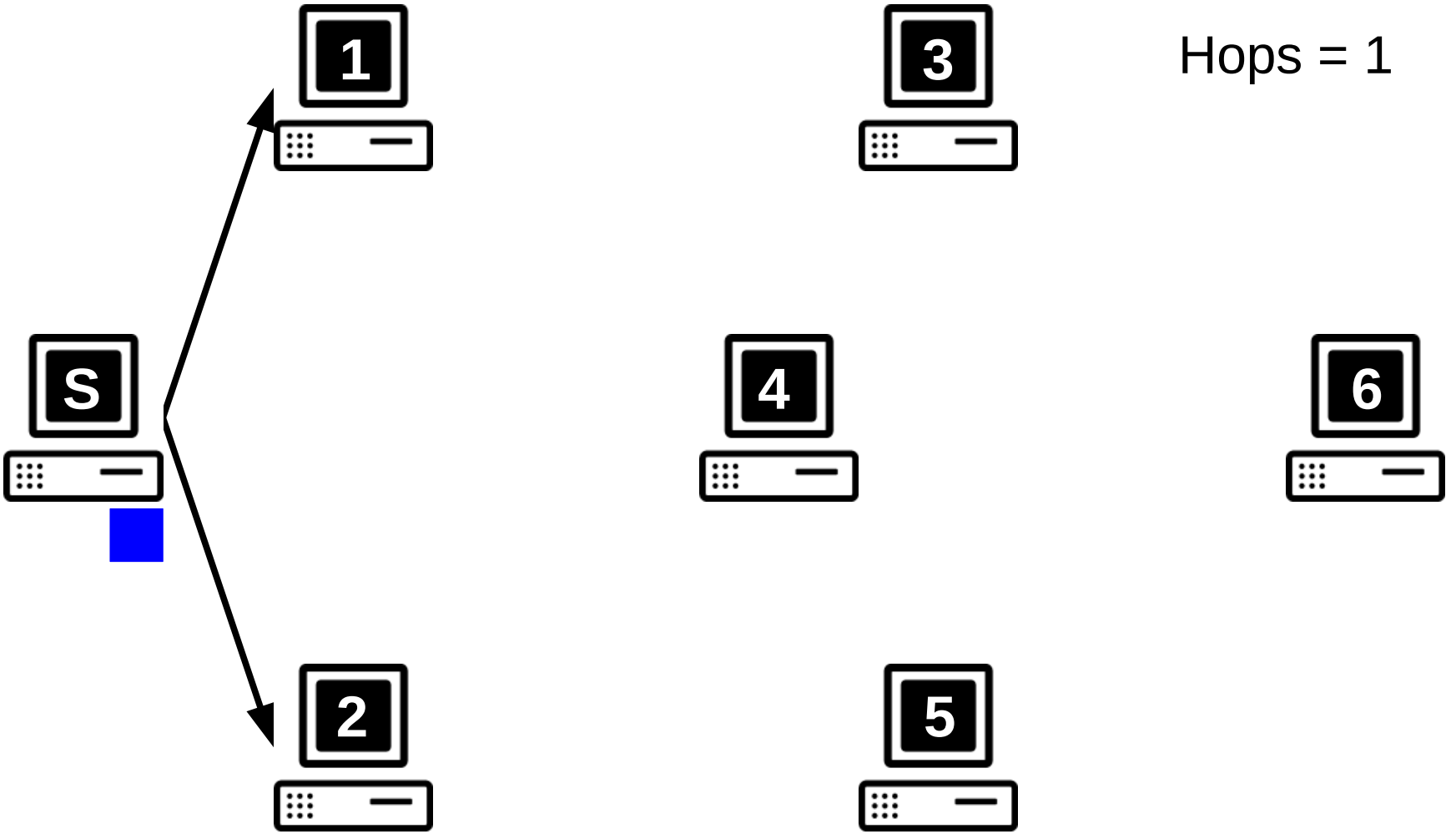
Content Dissemination: Gossip

Hops = 0

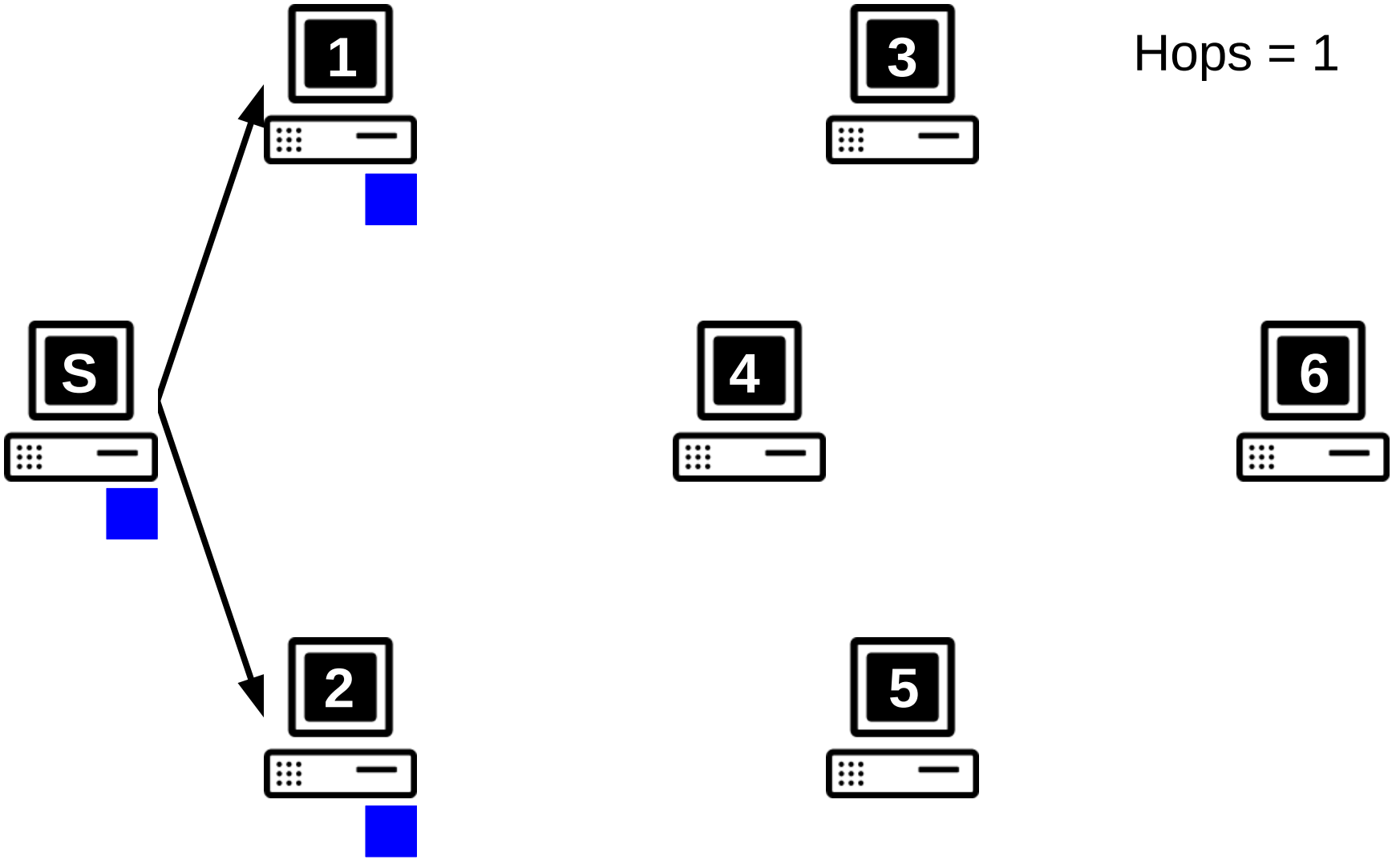
Content Dissemination: Gossip



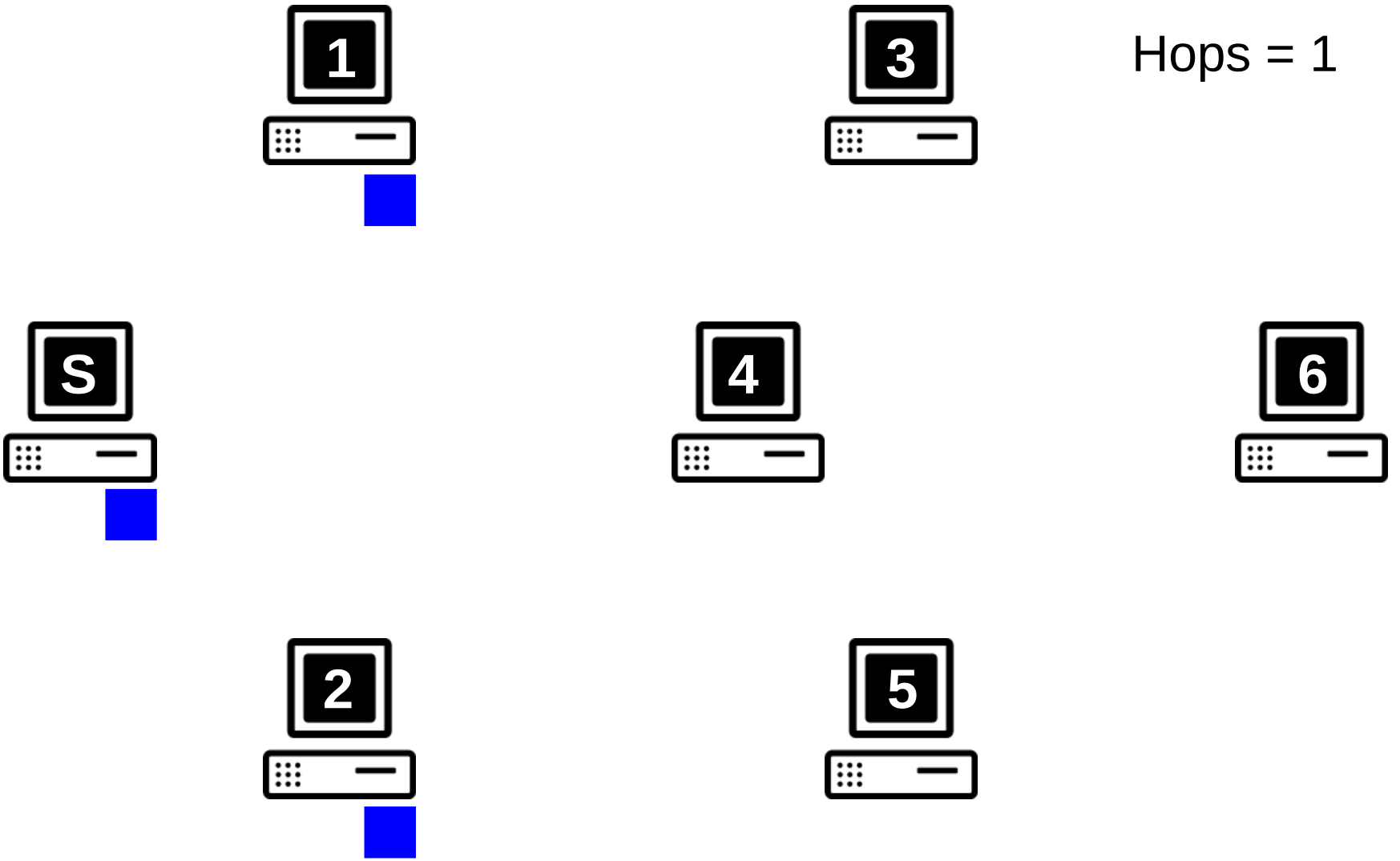
Content Dissemination: Gossip



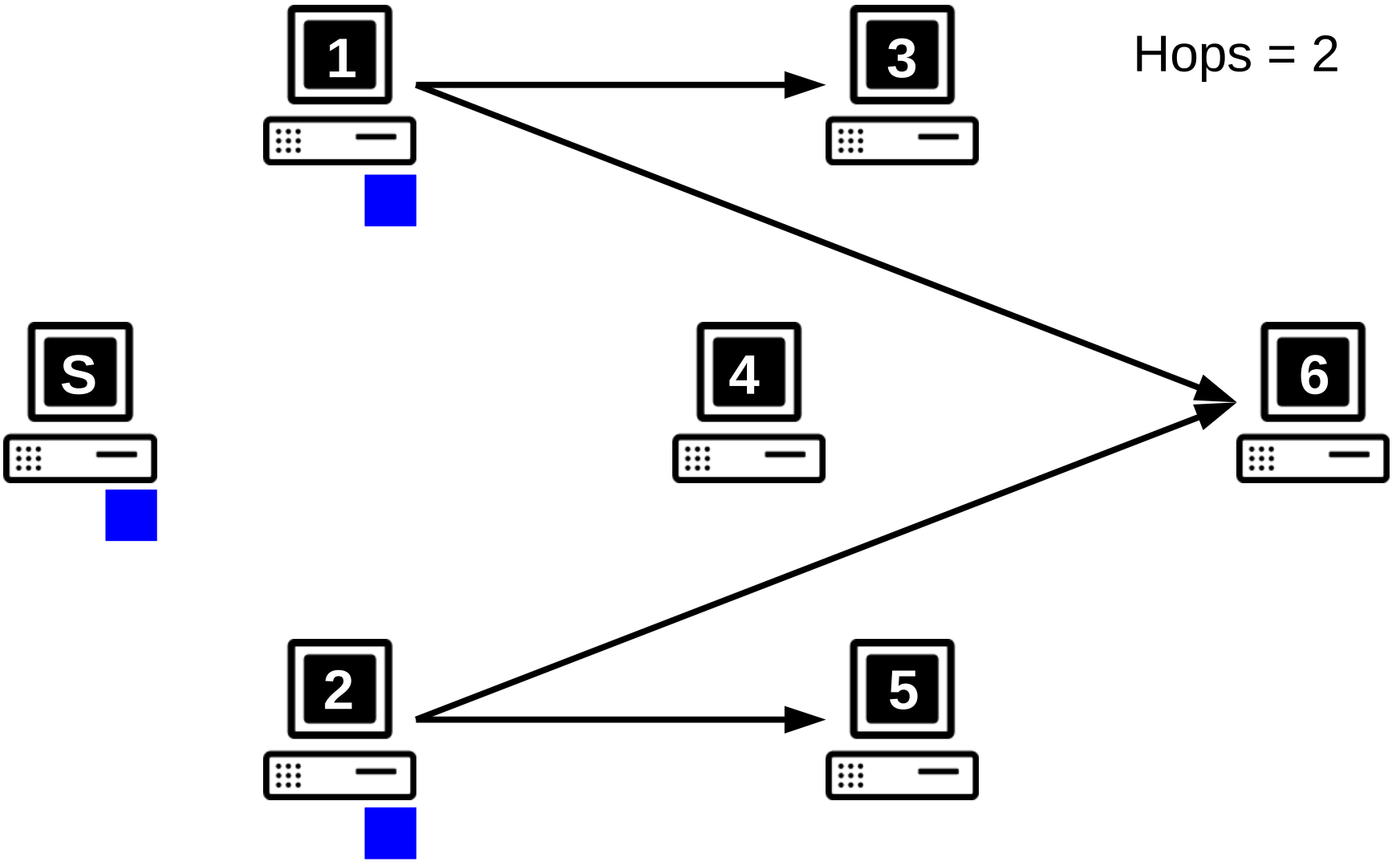
Content Dissemination: Gossip



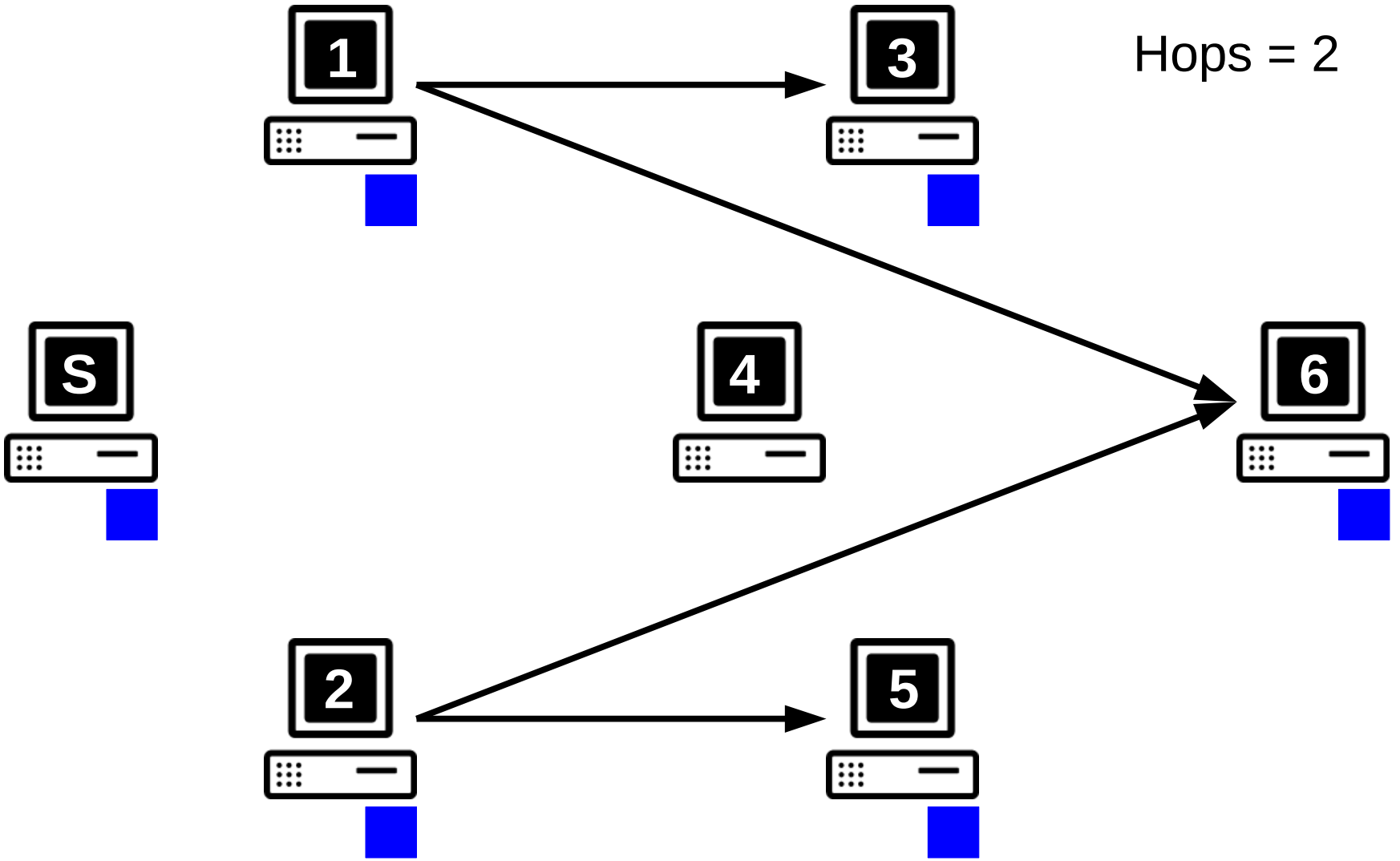
Content Dissemination: Gossip



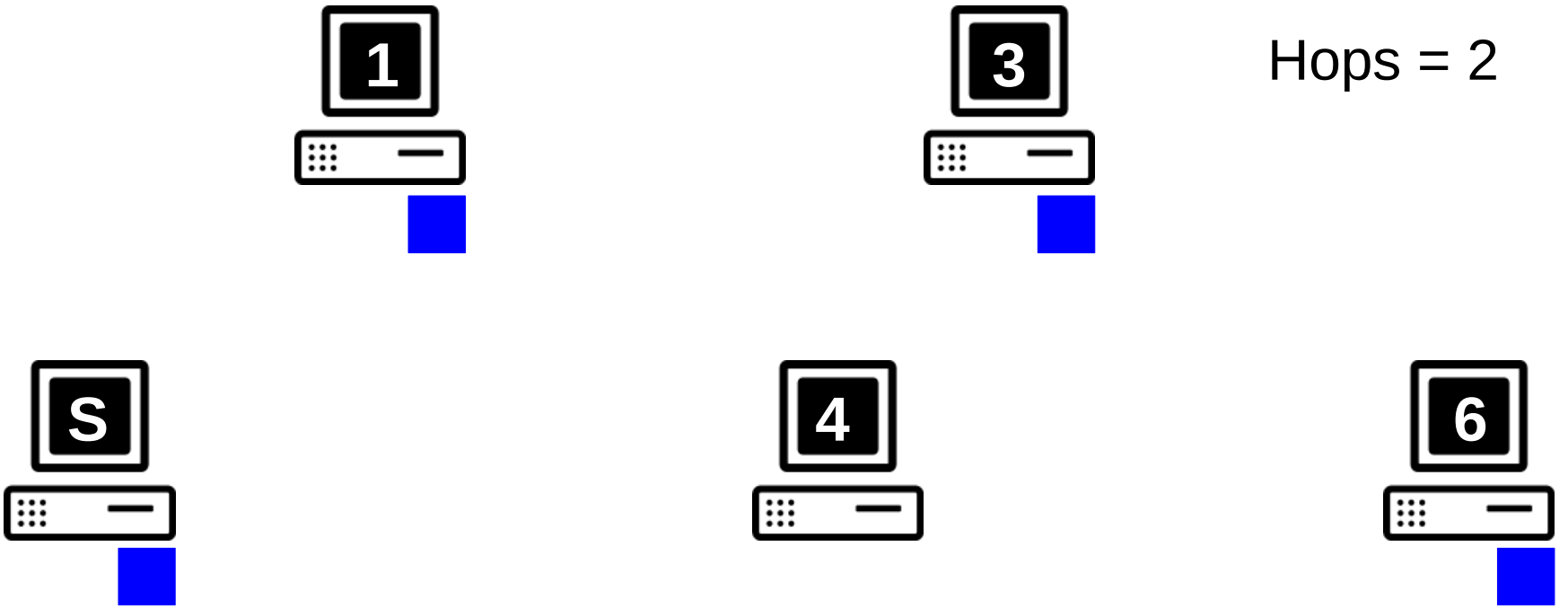
Content Dissemination: Gossip



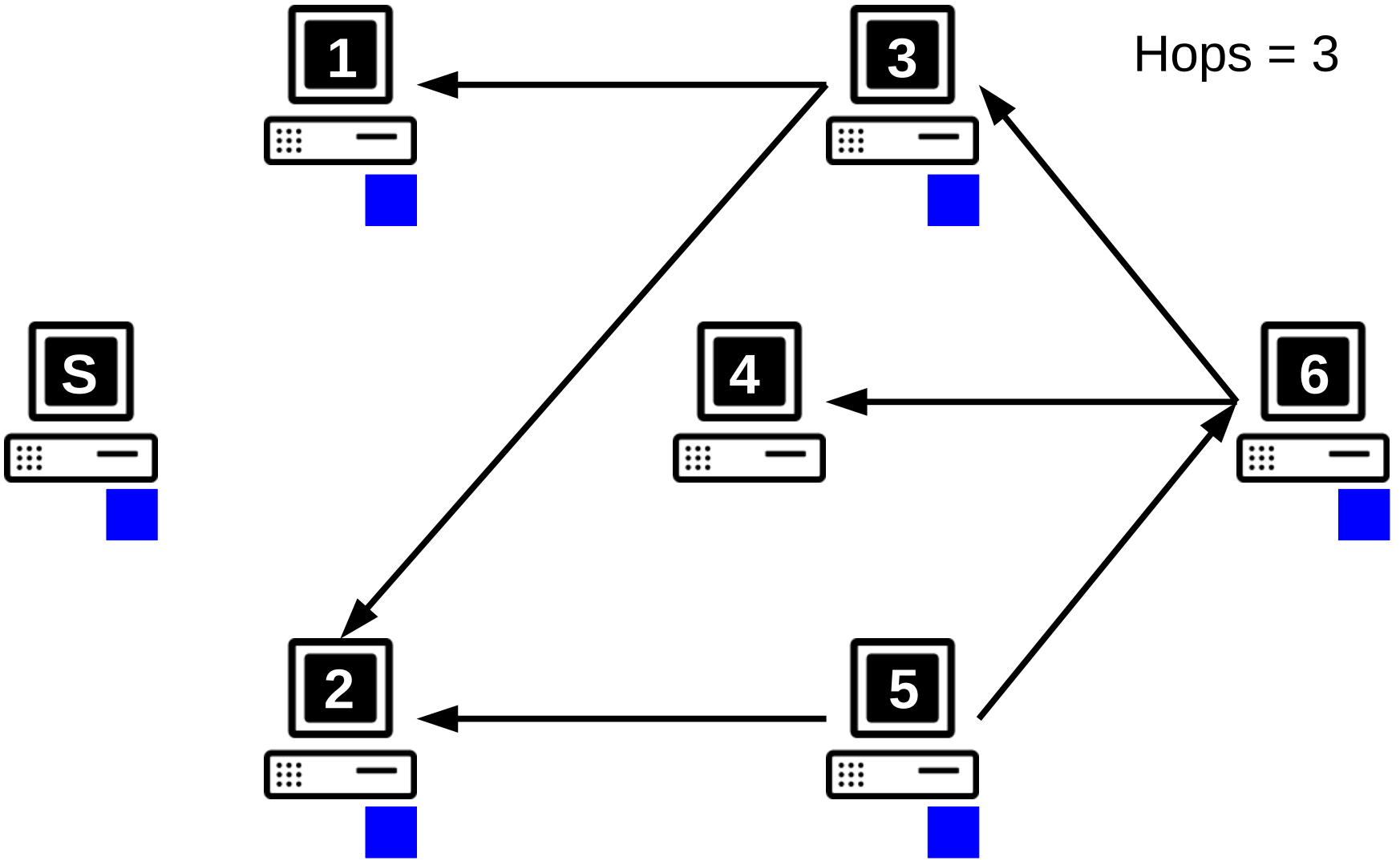
Content Dissemination: Gossip



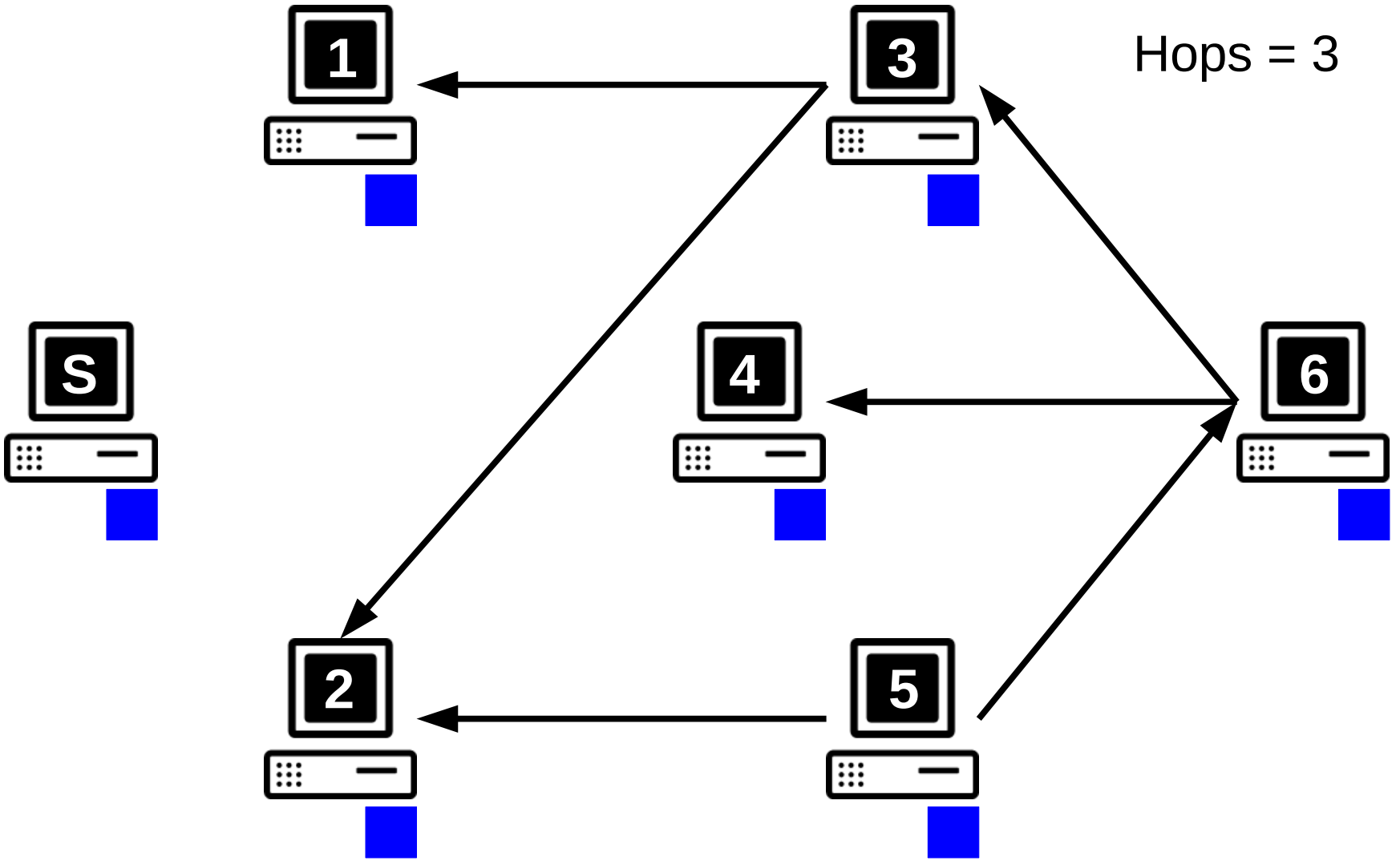
Content Dissemination: Gossip



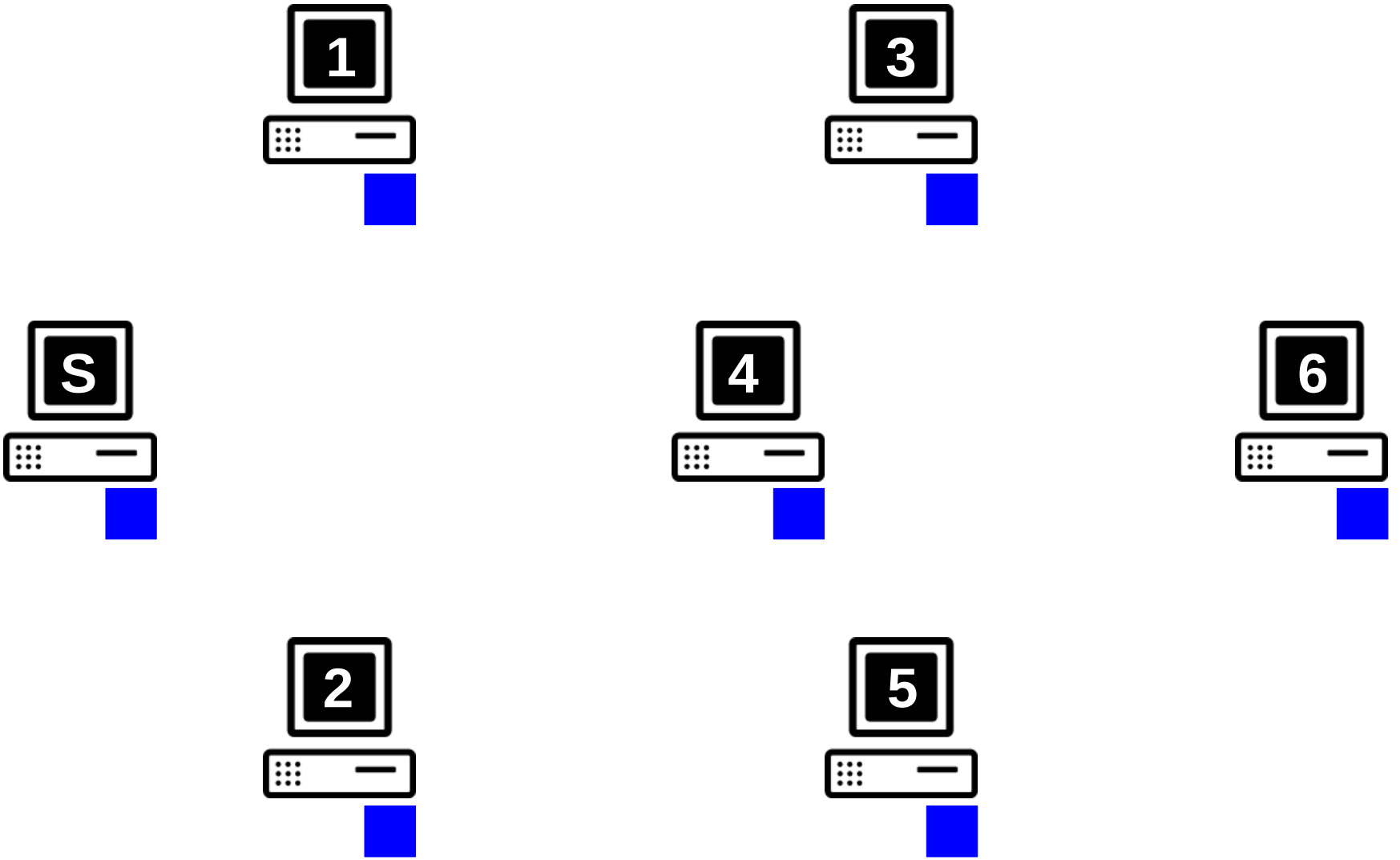
Content Dissemination: Gossip



Content Dissemination: Gossip



Content Dissemination: Gossip



Rational Nodes

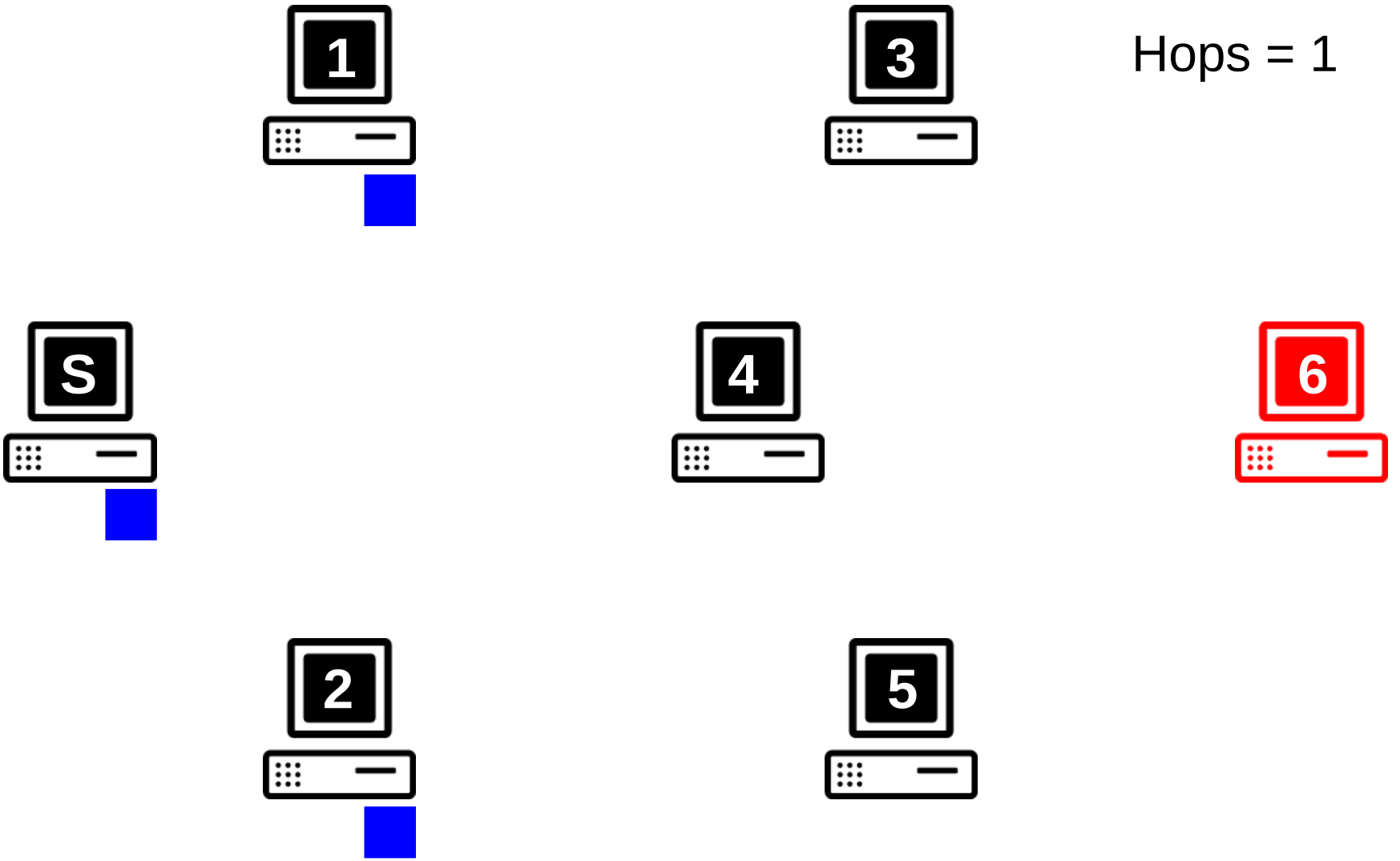
BAR Model: Rational nodes aim at **getting the content** at **the lowest possible overhead.**

Specifically, the benefit of rational nodes can be represented along the following axes:

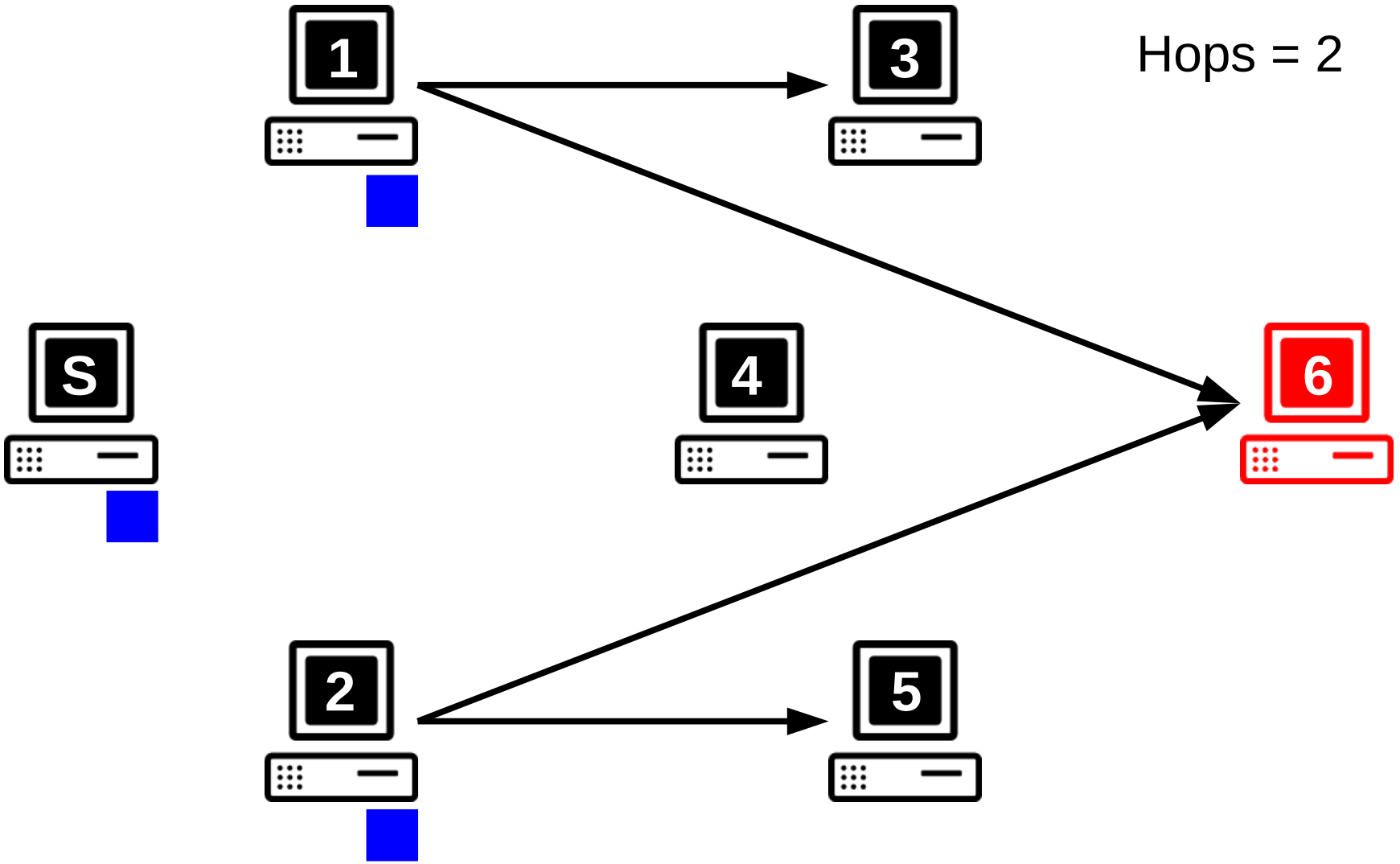
- **(Stream quality)** Receiving as much as possible stream updates,
- **(Communication)** Sending as little as possible stream updates or protocol messages,
- **(Computation)** Performing as few as possible computations,
- **(Risk)** Minimizing the risks.

Rational nodes would deviate in any sort from the protocol, as long as they increase their benefit. Doing so, they will potentially limit the good dissemination of updates.

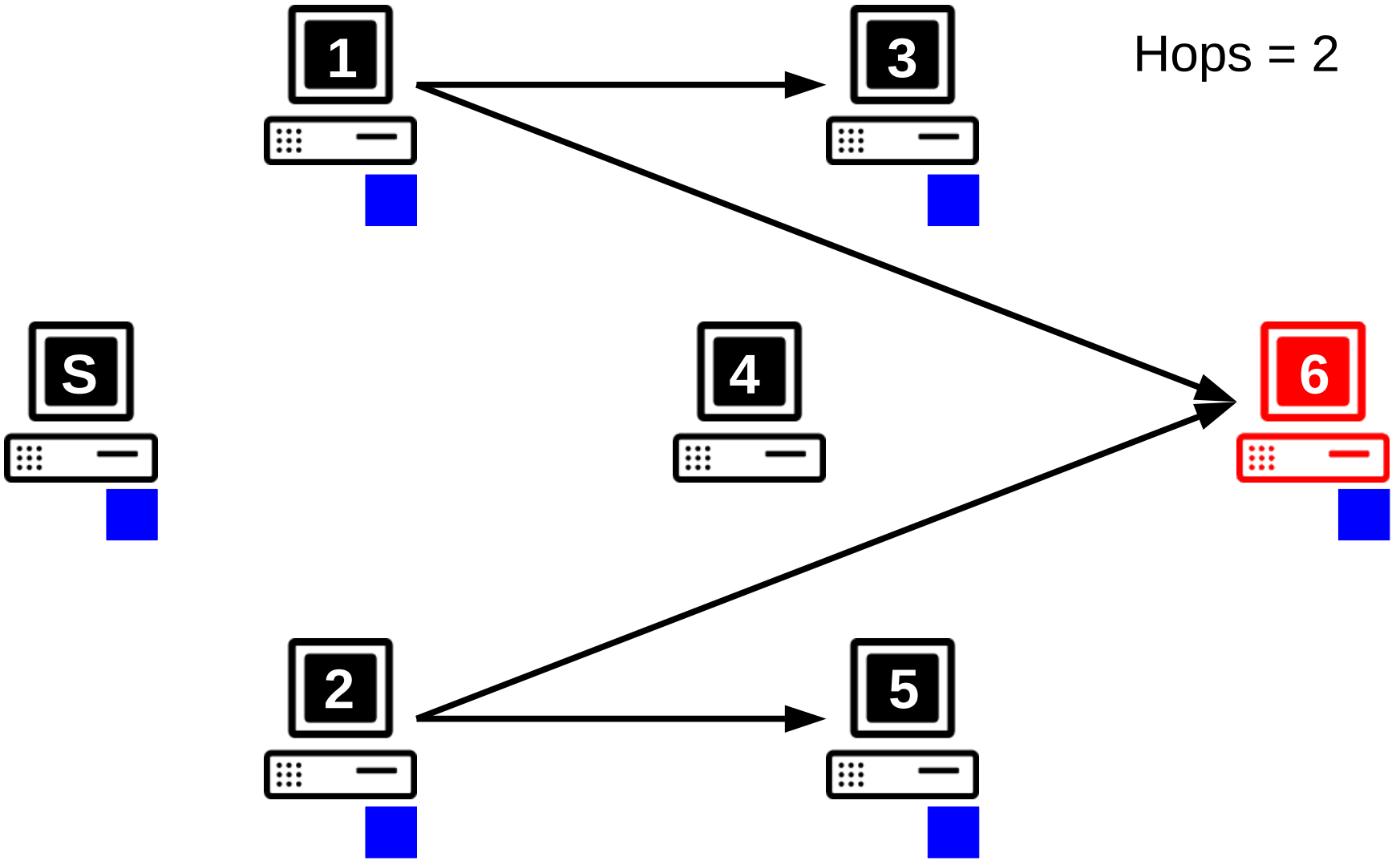
Gossip with a Rational Node



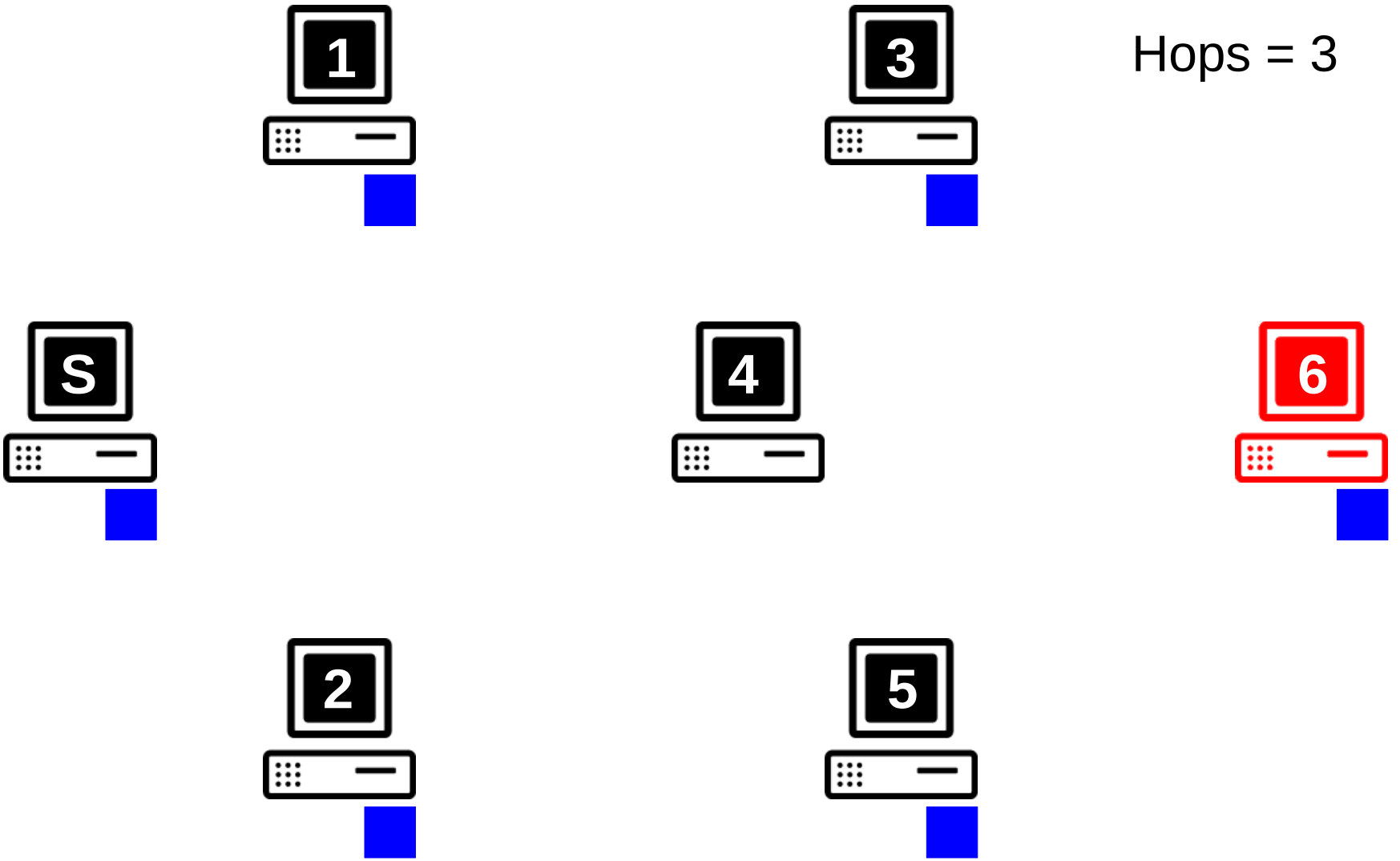
Gossip with a Rational Node



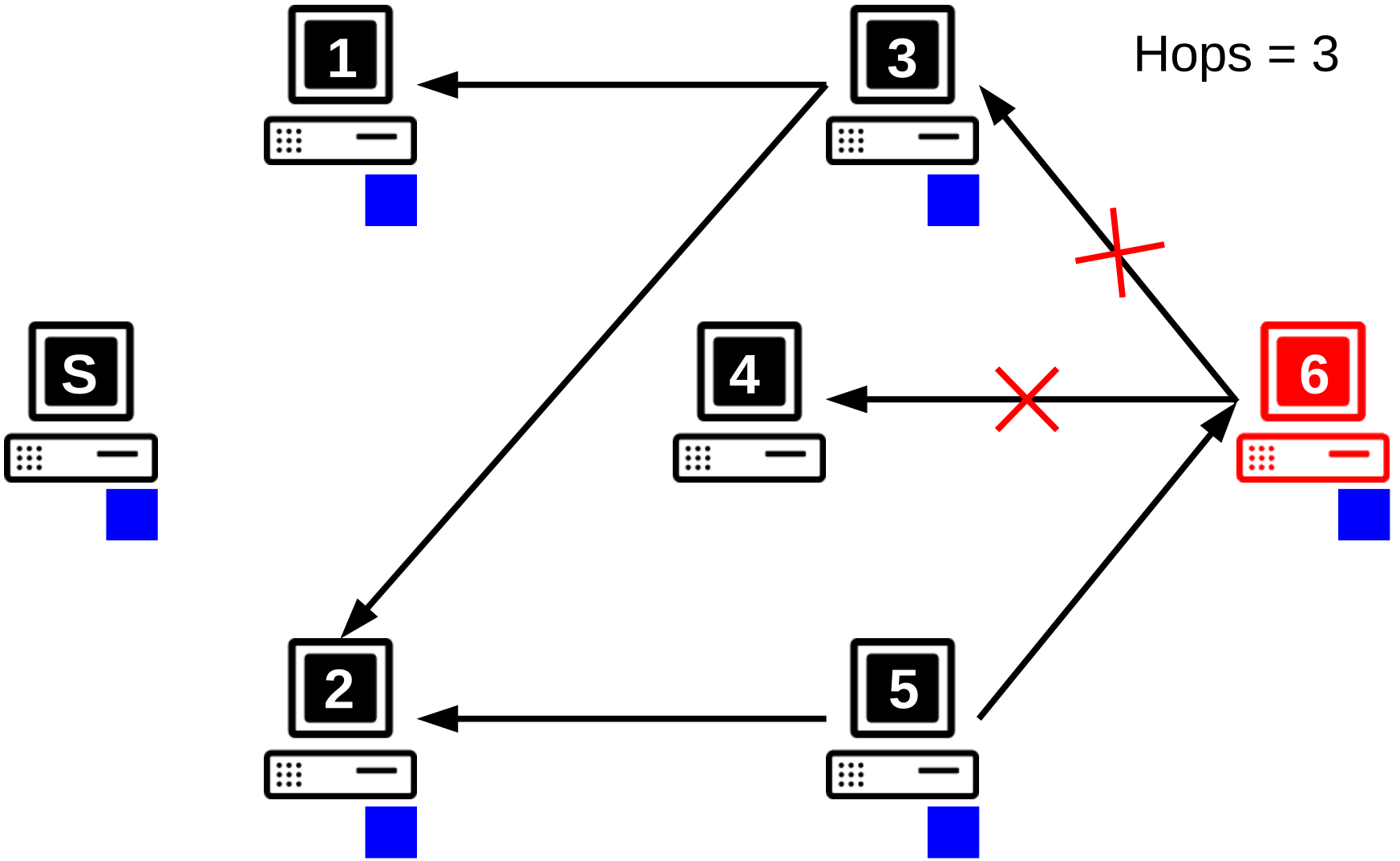
Gossip with a Rational Node



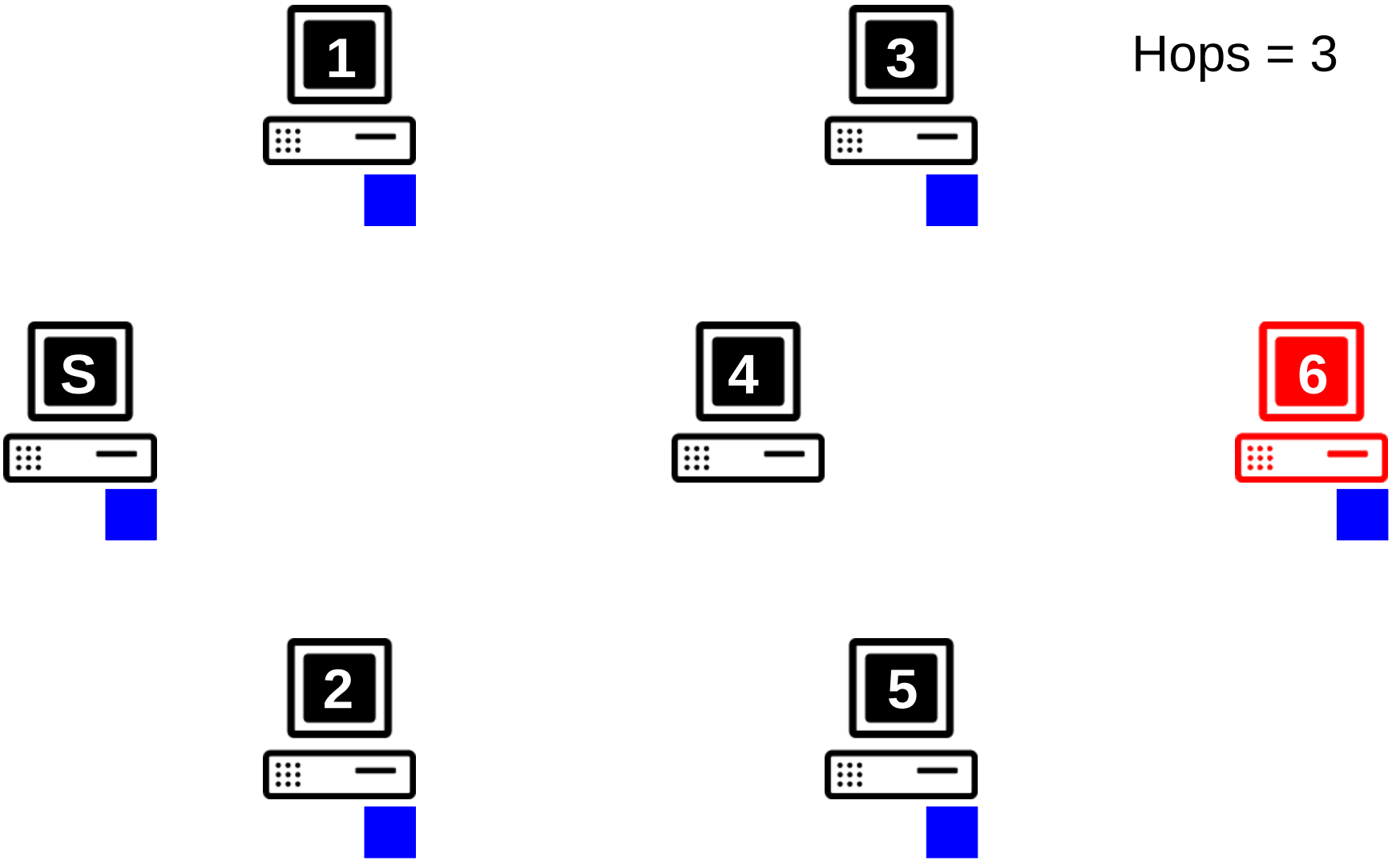
Gossip with a Rational Node



Gossip with a Rational Node



Gossip with a Rational Node



Rational Resilient Protocols

- **BAR Gossip** [OSDI'06], FlightPath [OSDI'08]
  - Delete the randomness of partners associations,
  - Symmetric exchanges (Tit-for-Tat),
  - Cryptographic primitives to delay gratification.
- **LiFTinG** [2011]
  - Asymmetric exchanges,
  - Nodes have a score, and audit each others through unprotected mechanisms, which leads to false positives.
- **PeerReview** [MMCN'08]
  - Secure logs,
  - Audits done by volunteer witnesses.

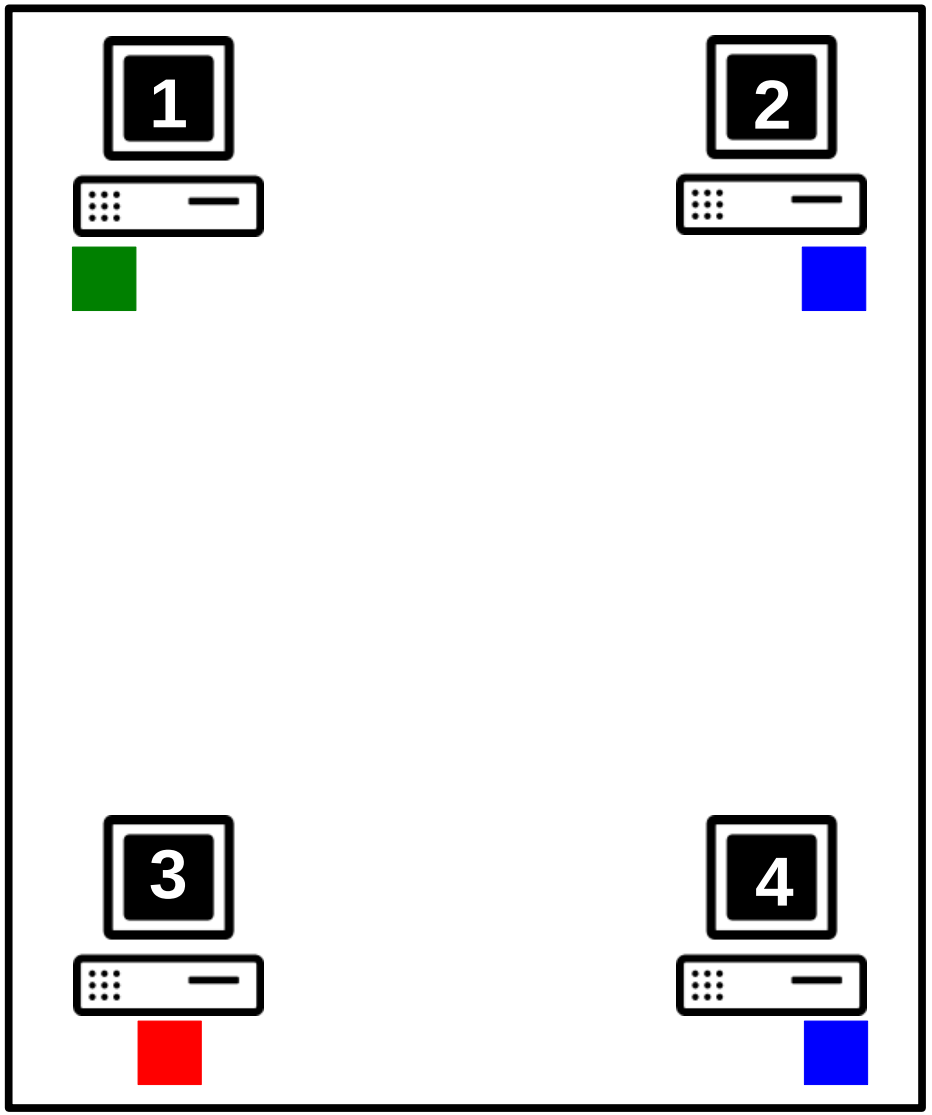
Colluding Rational Nodes

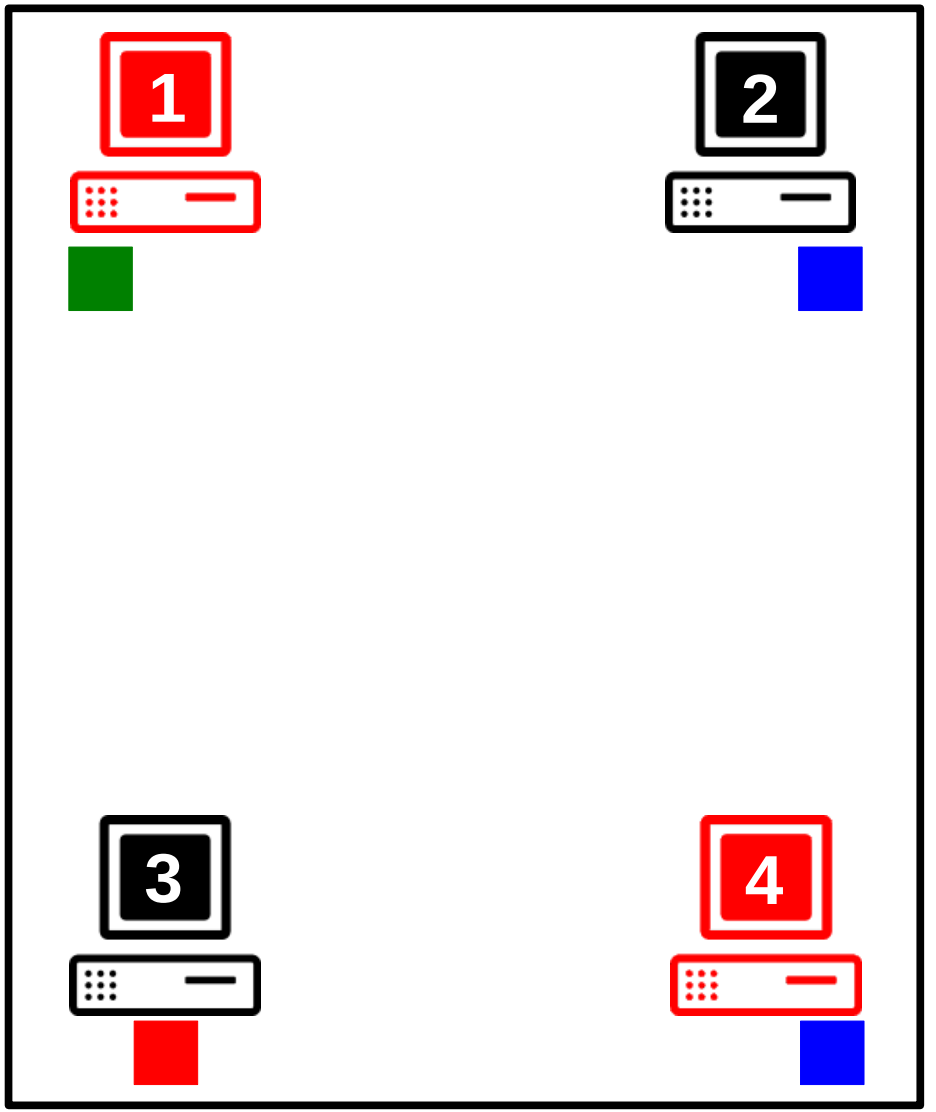
However, as some papers acknowledge rational nodes may collude with each other to:

- escape the mechanisms designed to detect individual rational nodes,
- Increase their benefit.

A first step in our work was to measure the impact of this type of deviations on the state-of-the-art gossip protocols.

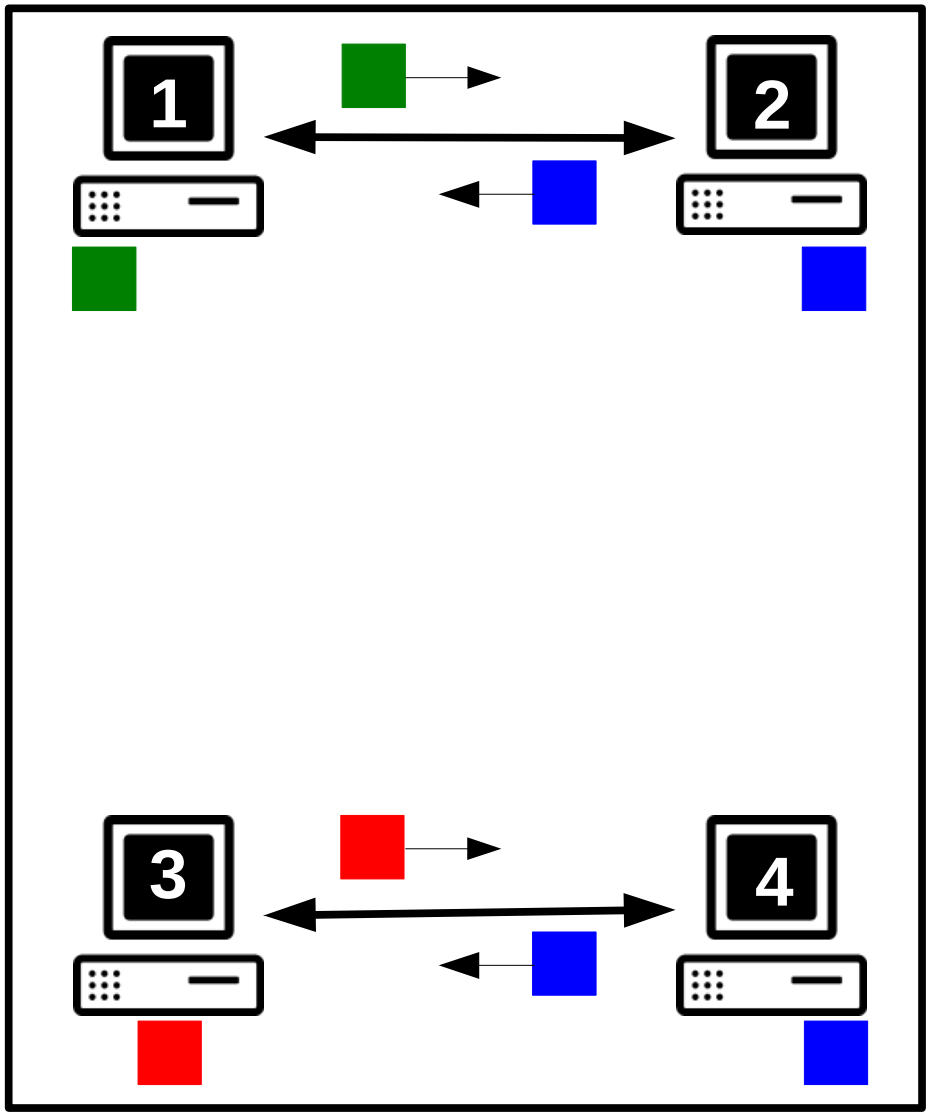
Symmetric Exchanges with Colluders





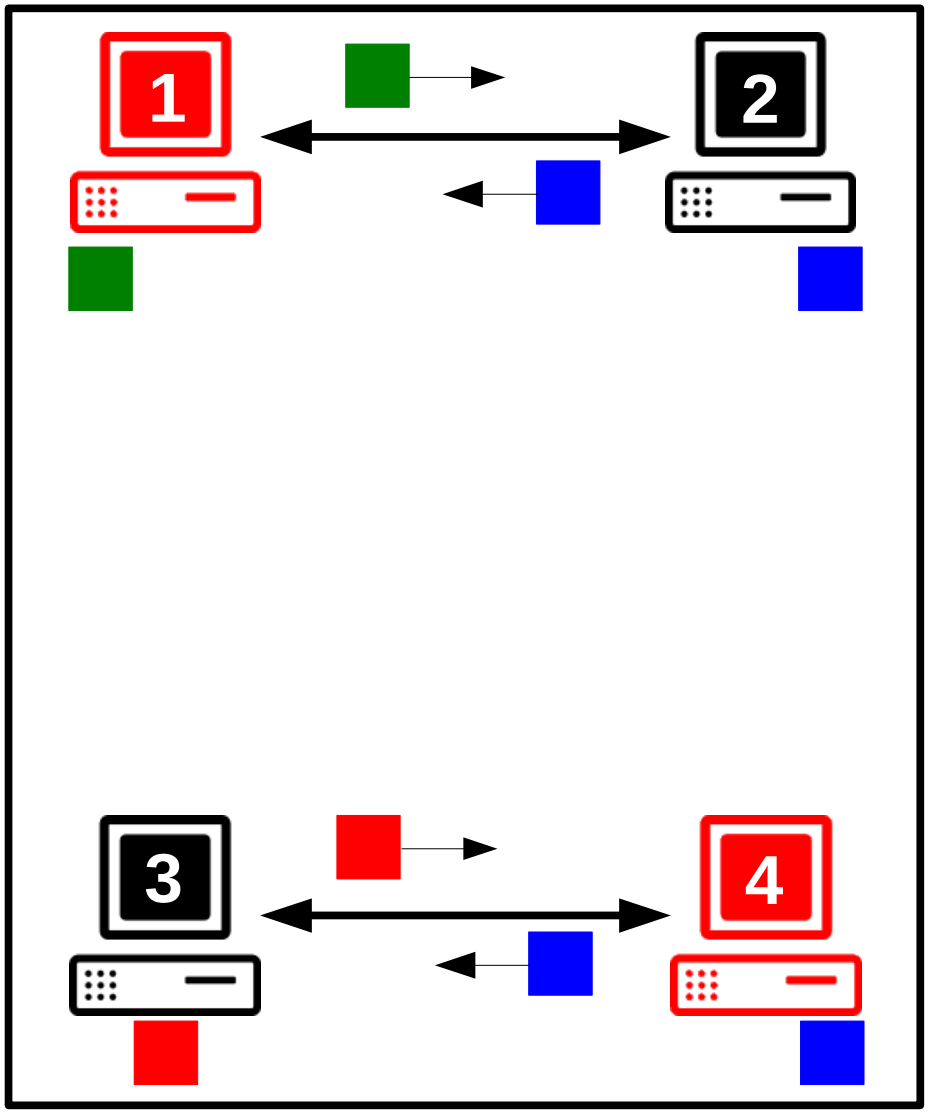
Without colluders Hops = 1 **With colluders**

Symmetric Exchanges with Colluders



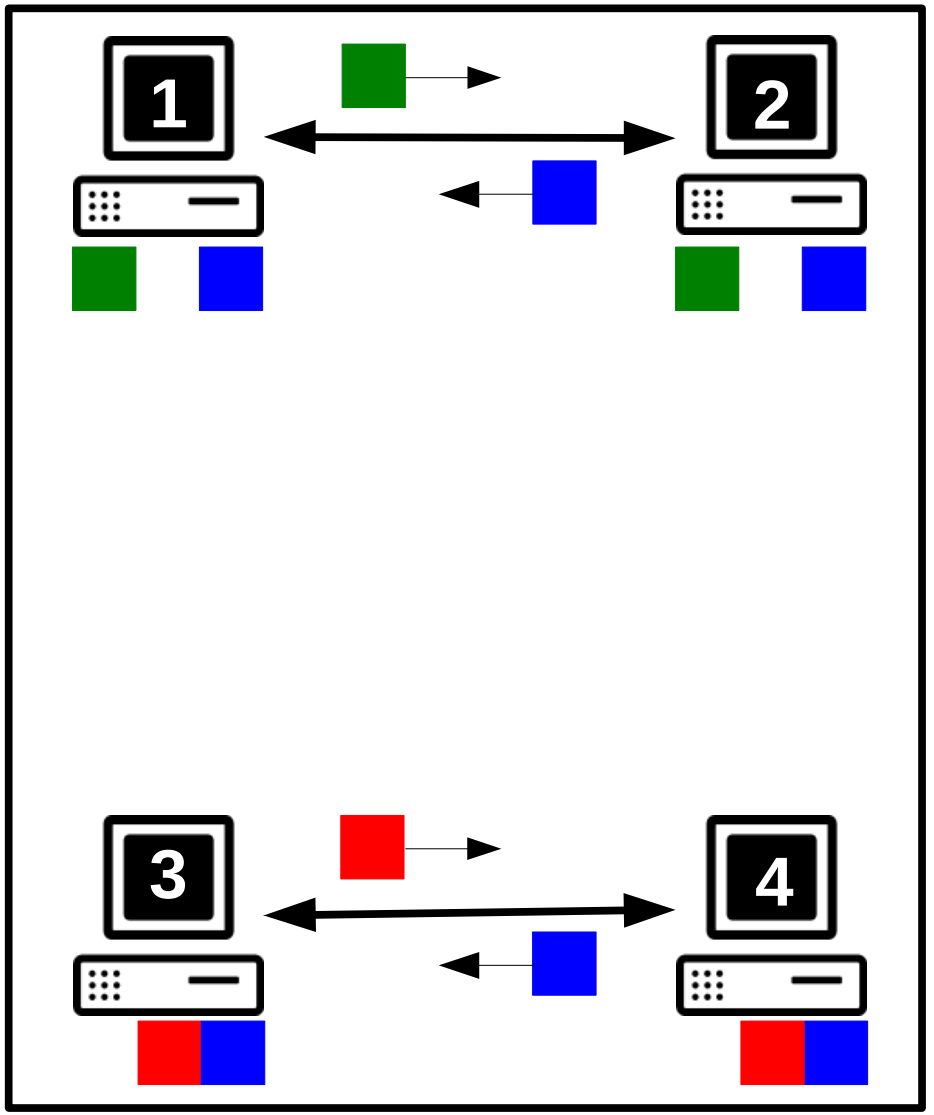
Without colluders

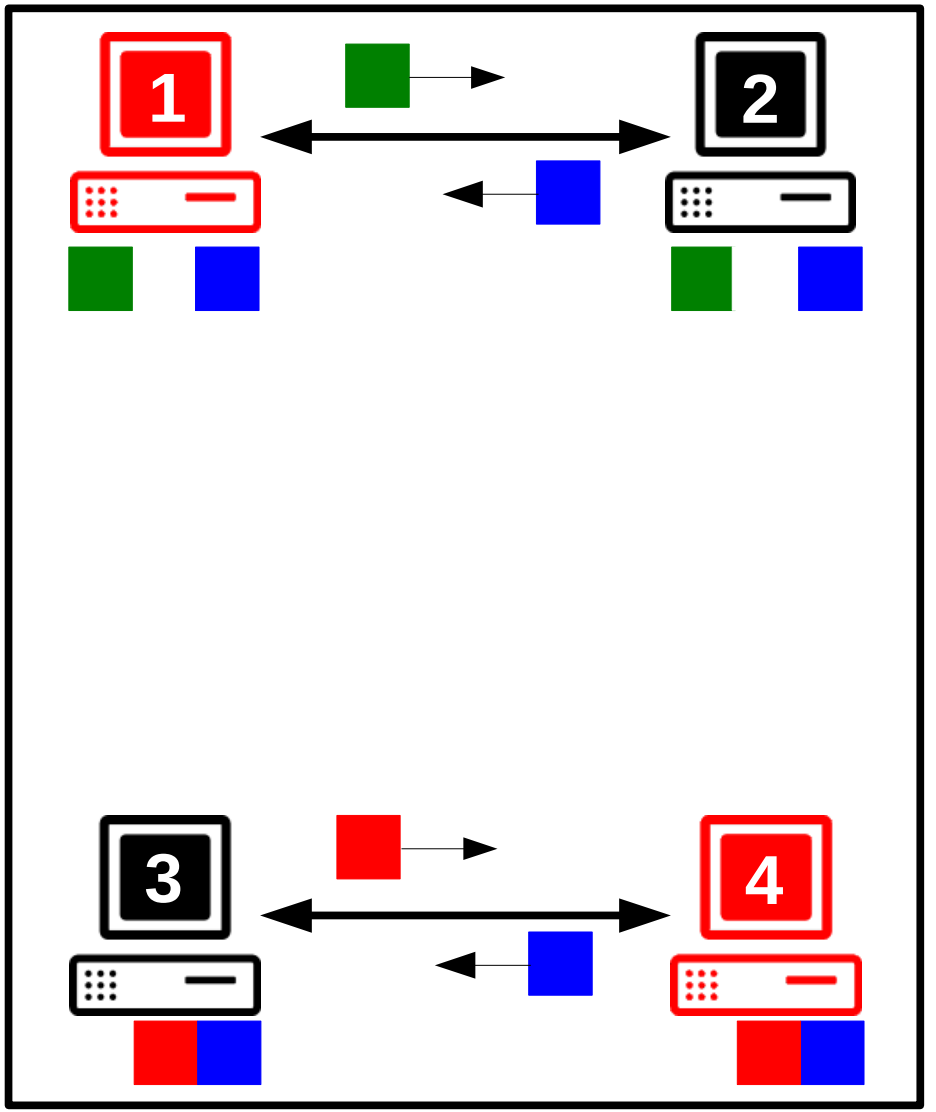
Hops = 1



With colluders

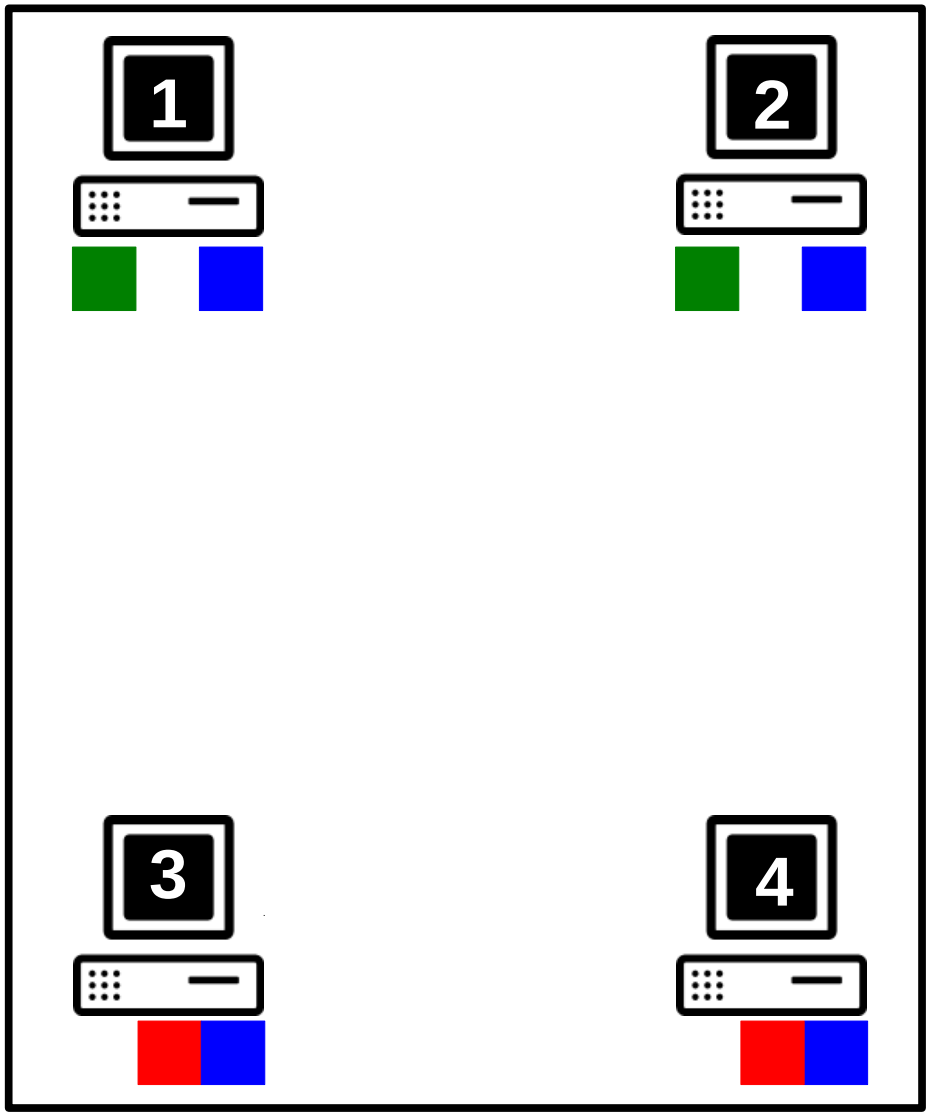
Symmetric Exchanges with Colluders





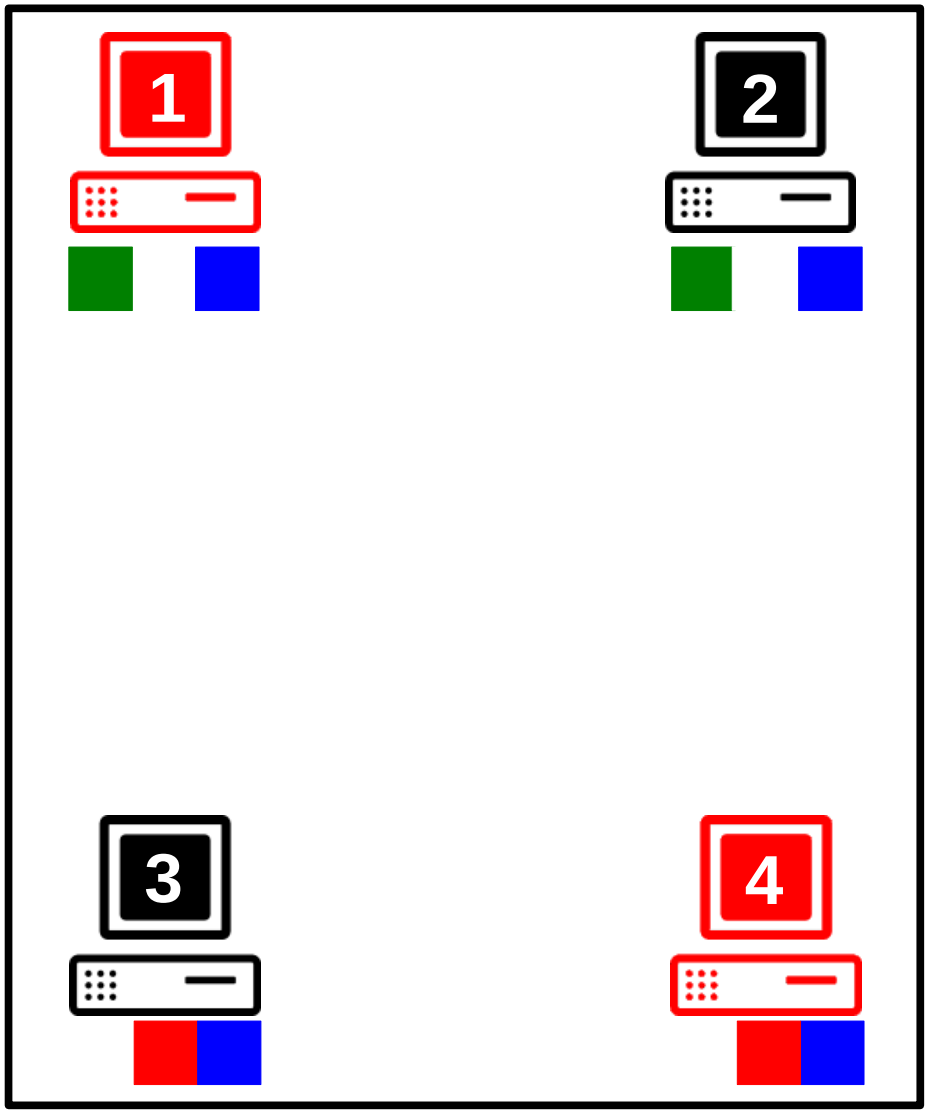
Without colluders Hops = 1 **With colluders**

Symmetric Exchanges with Colluders



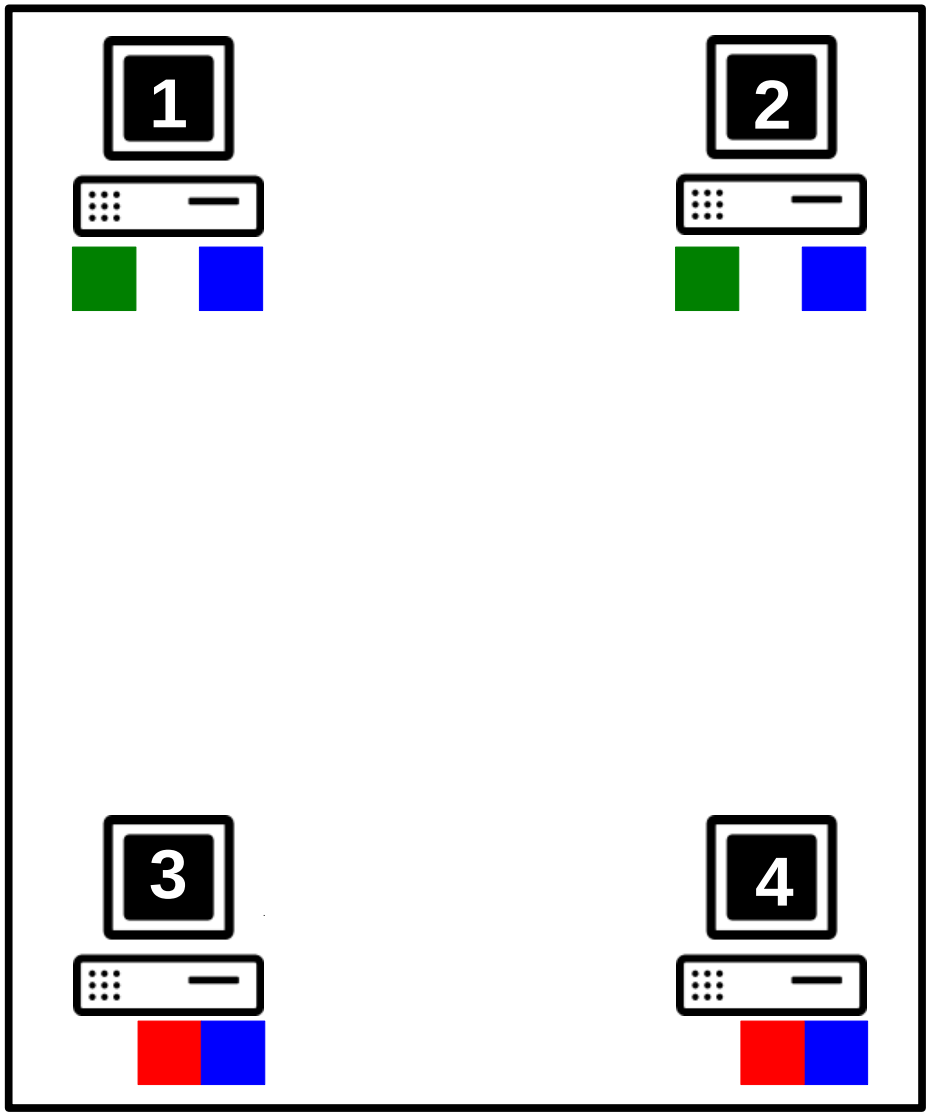
Without colluders

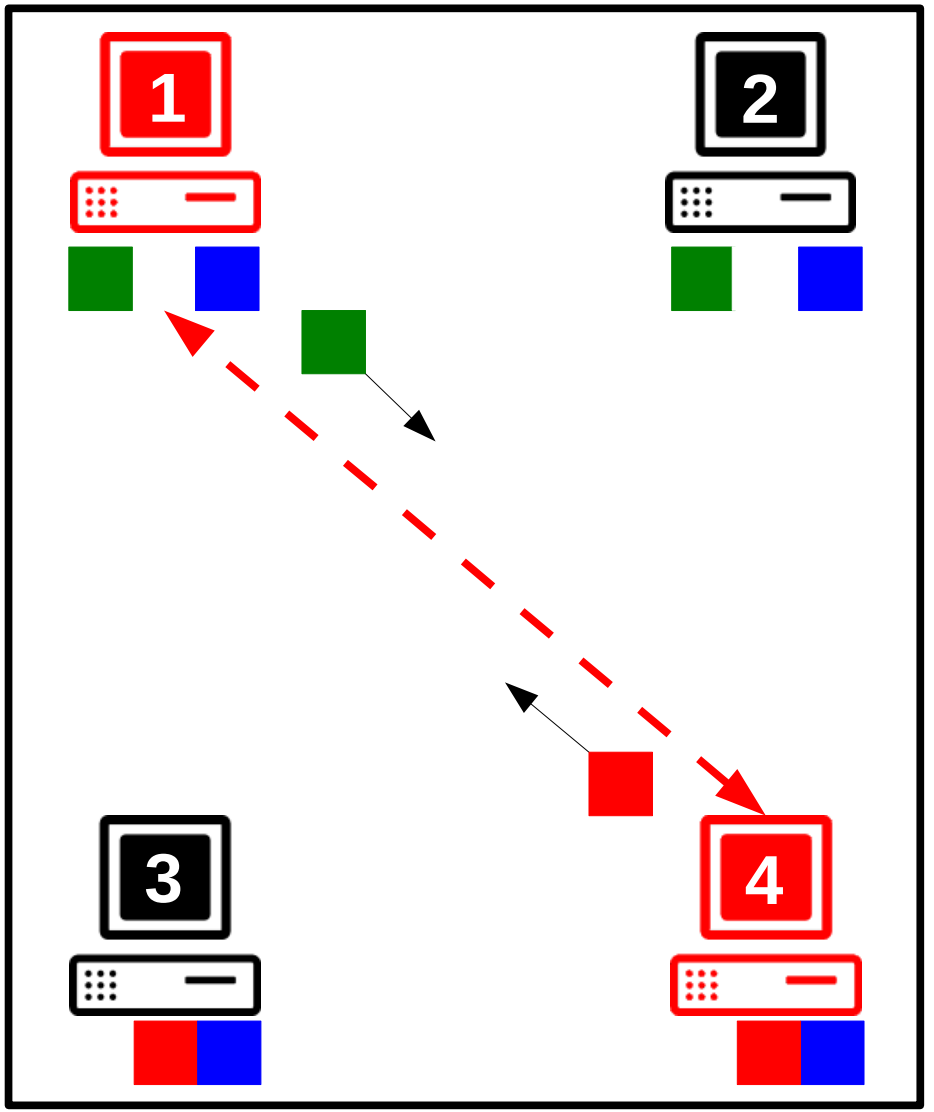
Hops = 1



With colluders

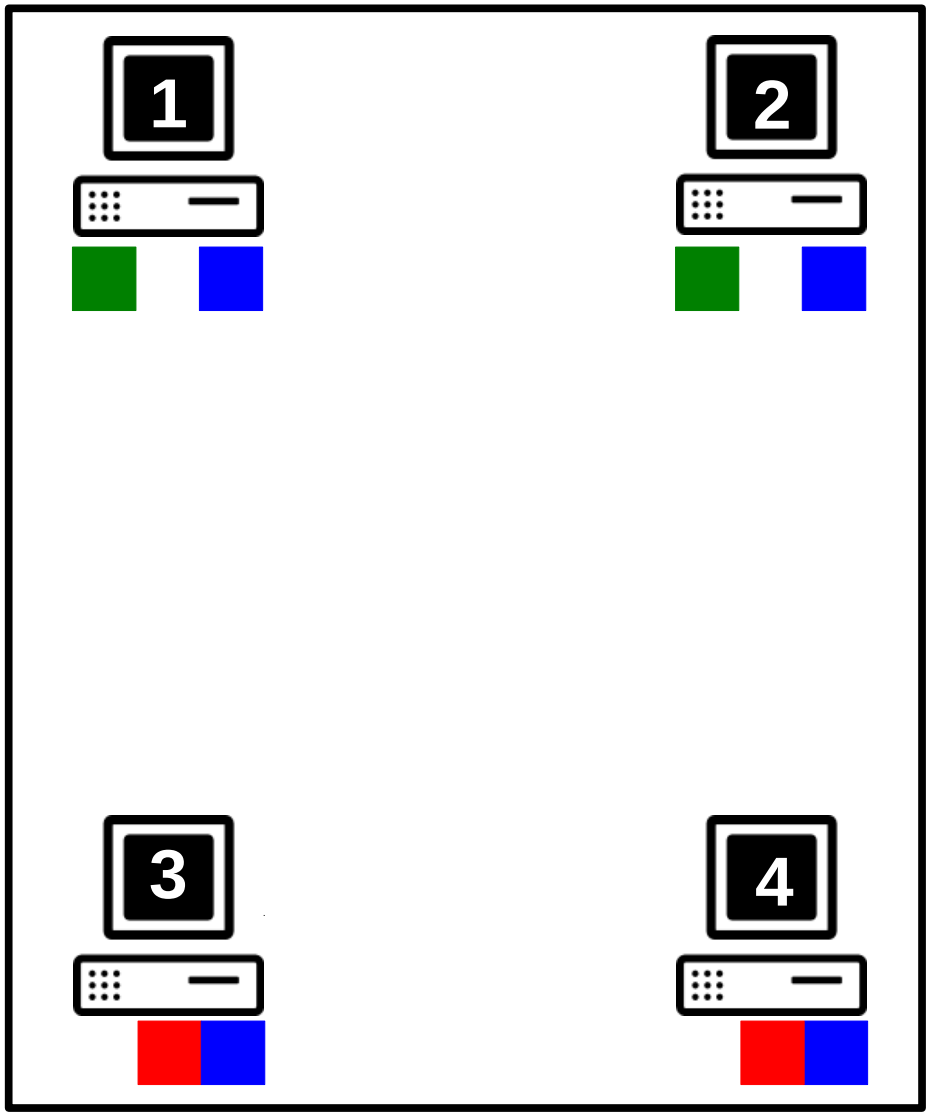
Symmetric Exchanges with Colluders

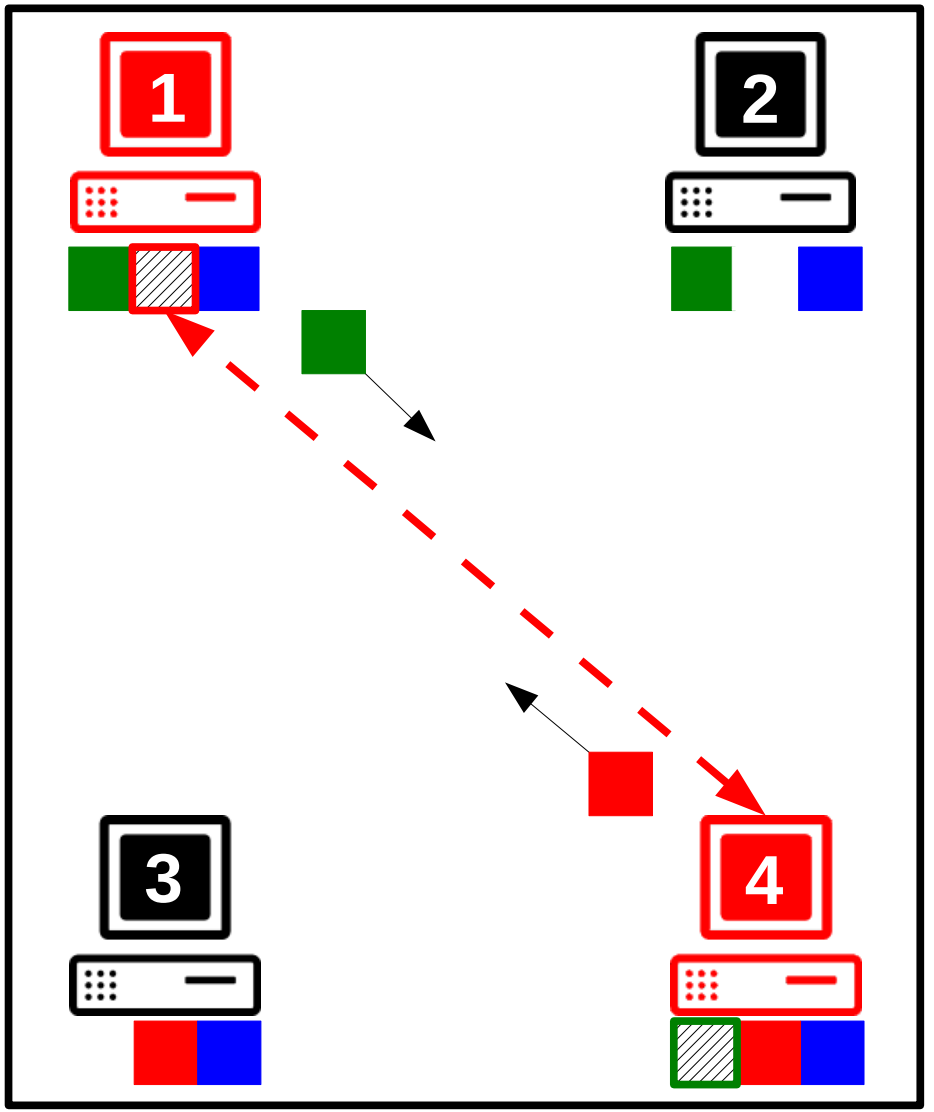




Without colluders Hops = 1 **With colluders**

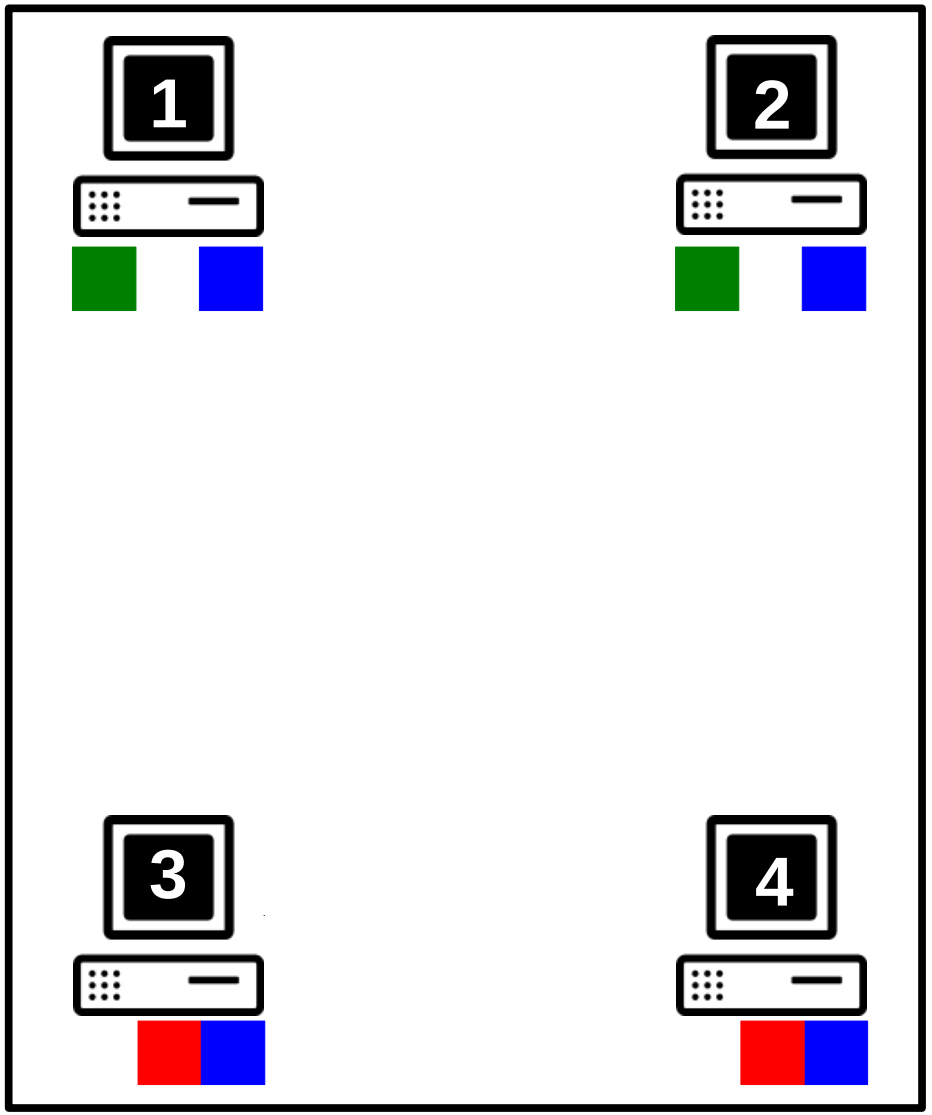
Symmetric Exchanges with Colluders

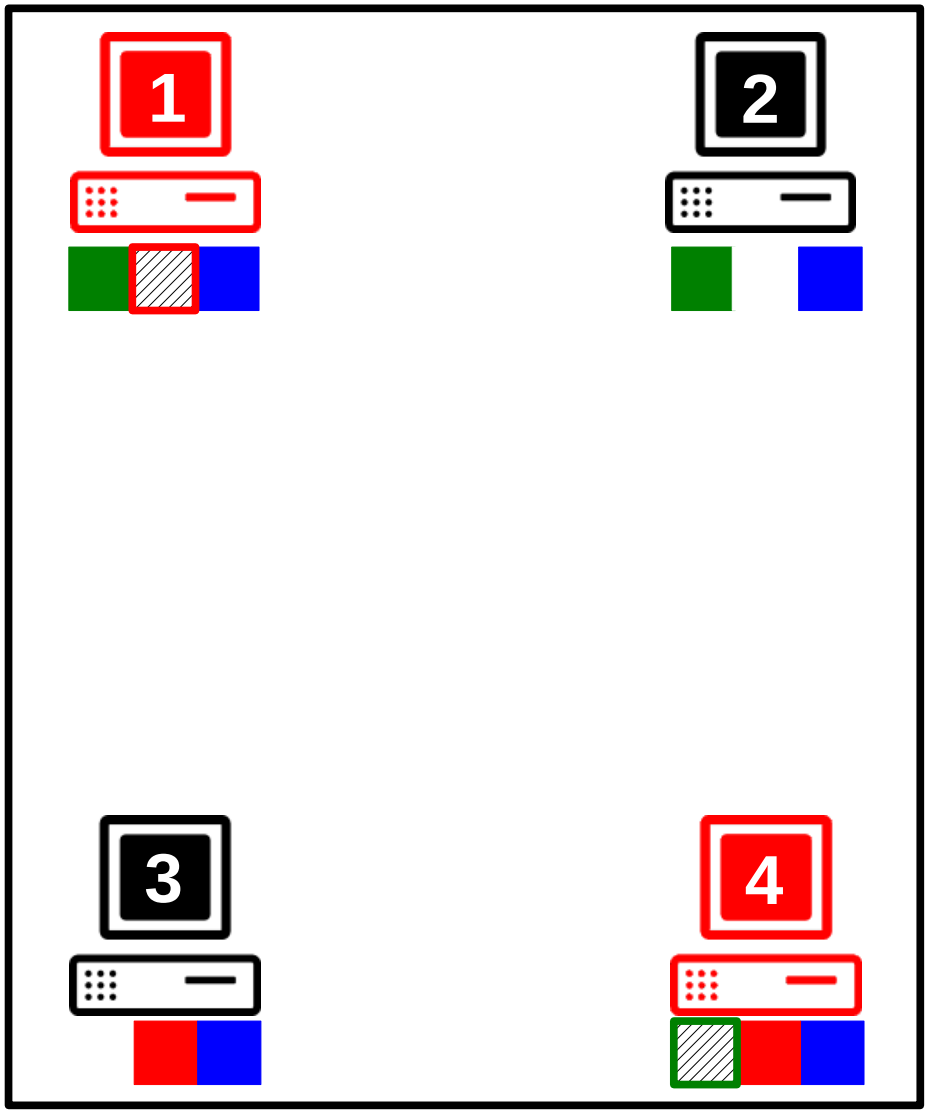




Without colluders Hops = 1 **With colluders**

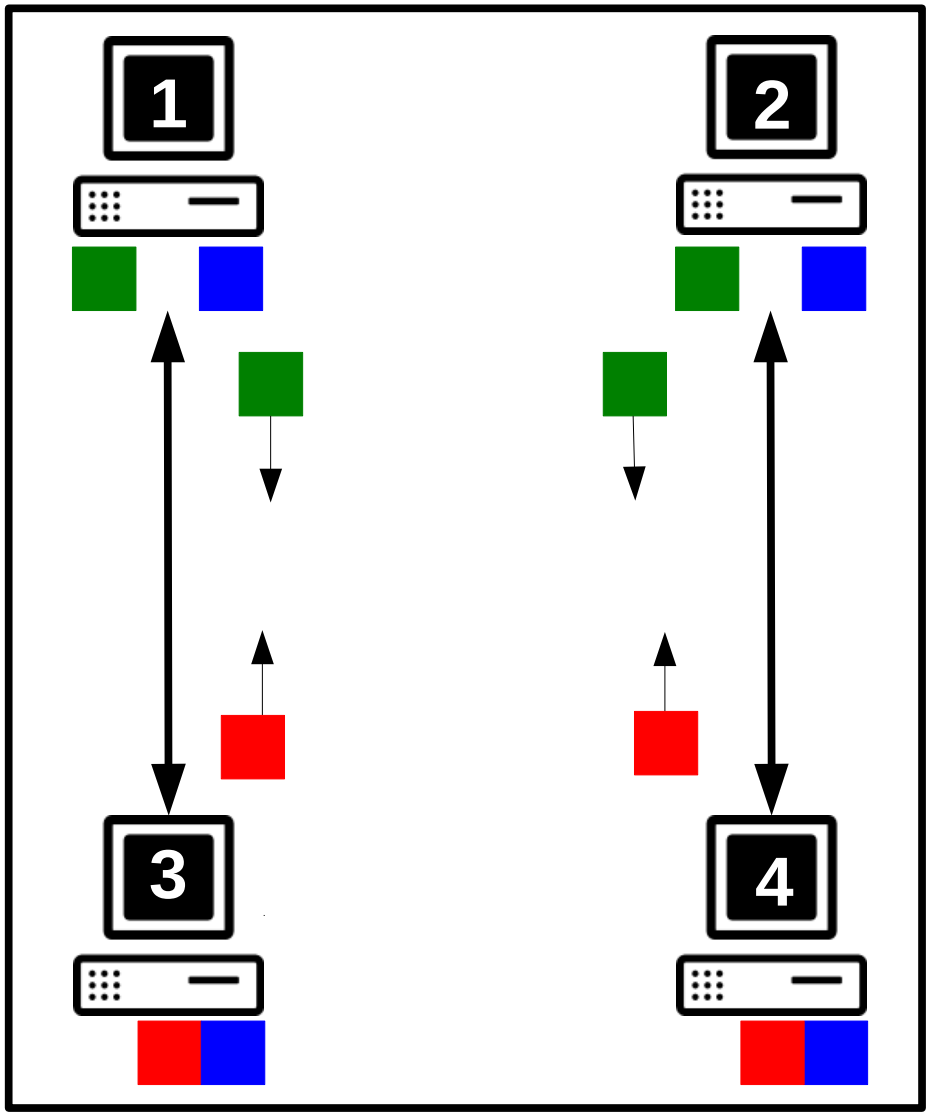
Symmetric Exchanges with Colluders

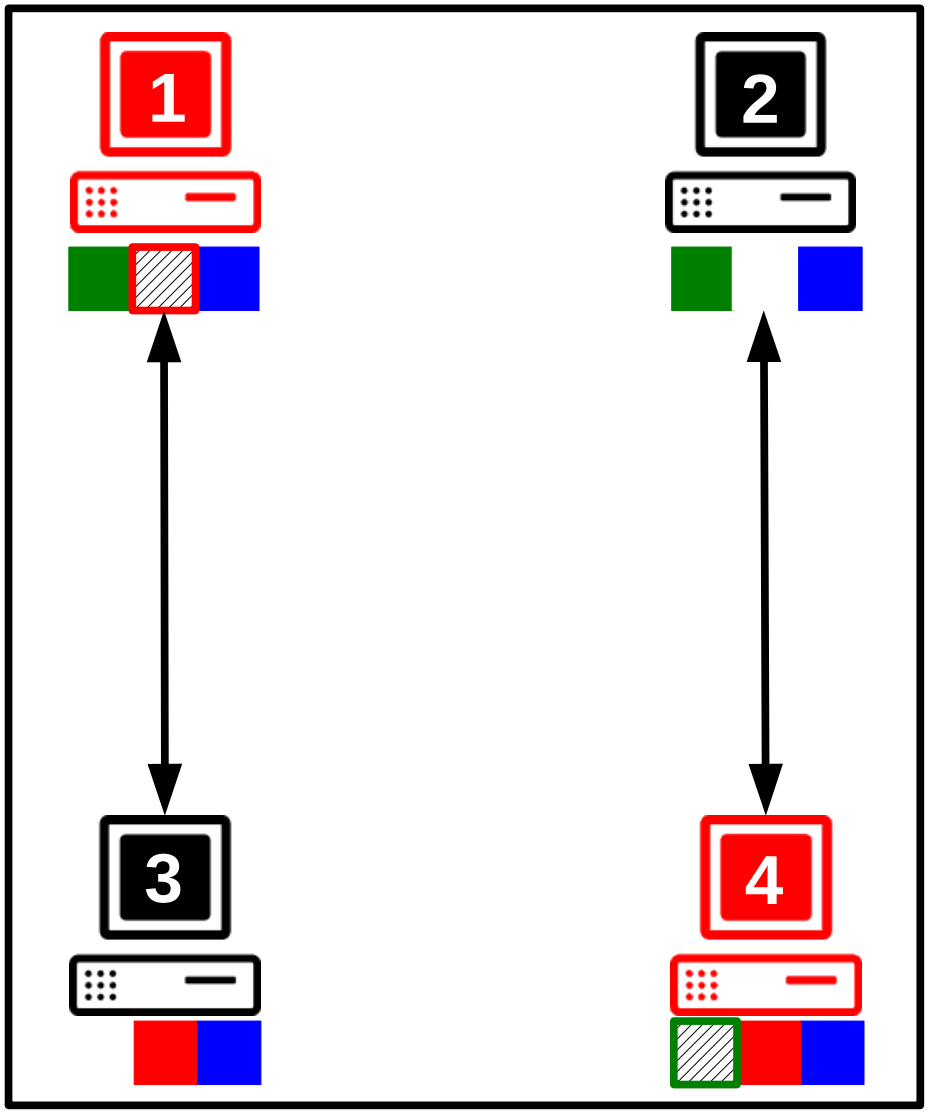




Without colluders Hops = 1 **With colluders**

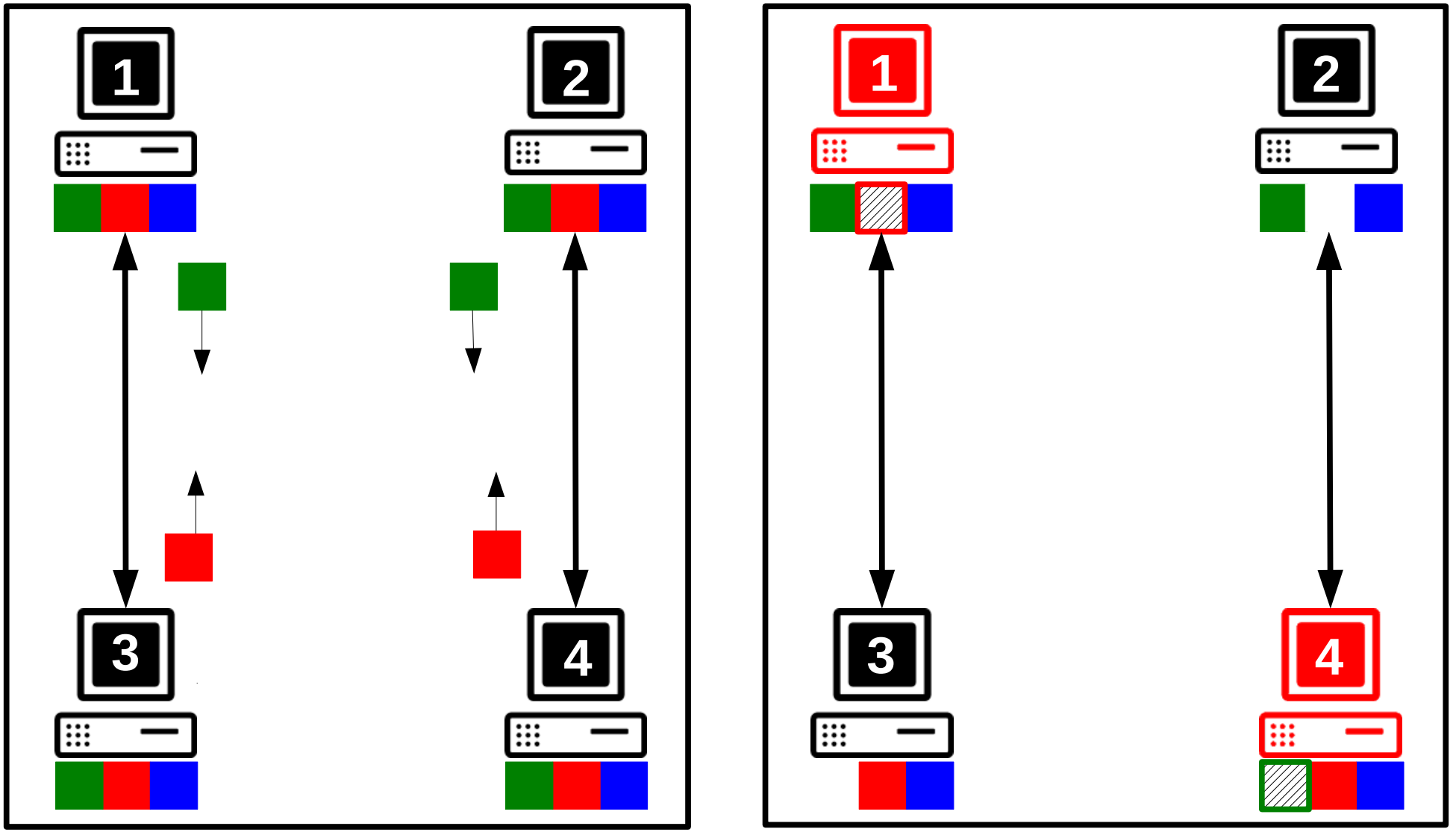
Symmetric Exchanges with Colluders





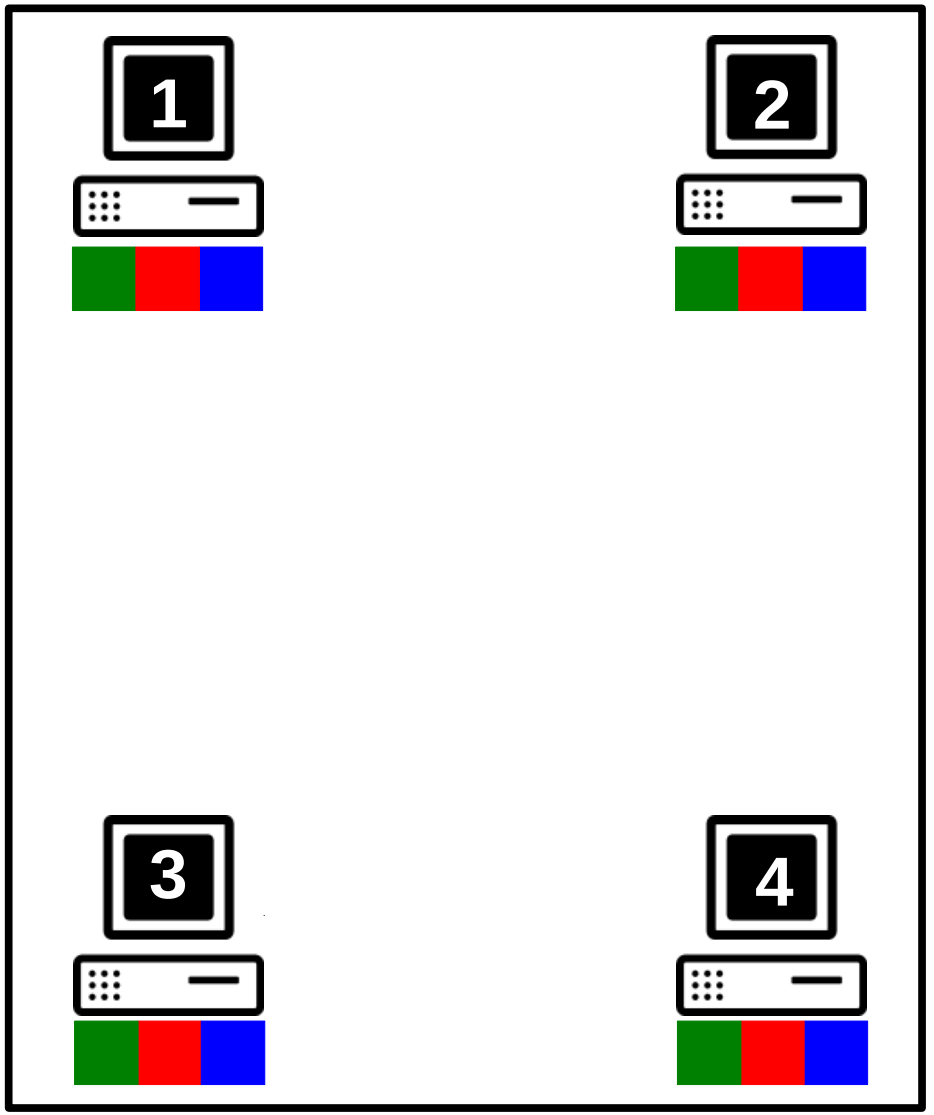
Without colluders Hops = 2 **With colluders**

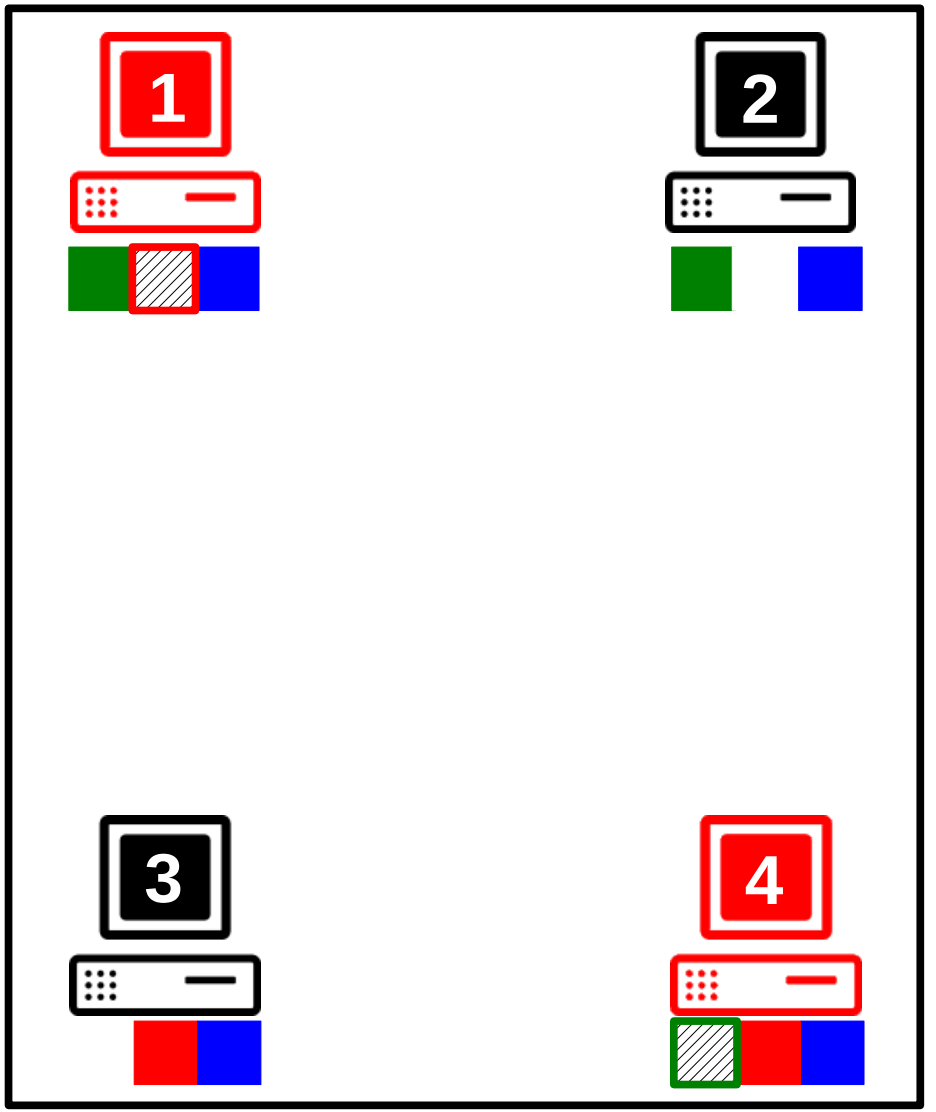
Symmetric Exchanges with Colluders



Without colluders Hops = 2 **With colluders**

Symmetric Exchanges with Colluders





Without colluders Hops = 2 **With colluders**

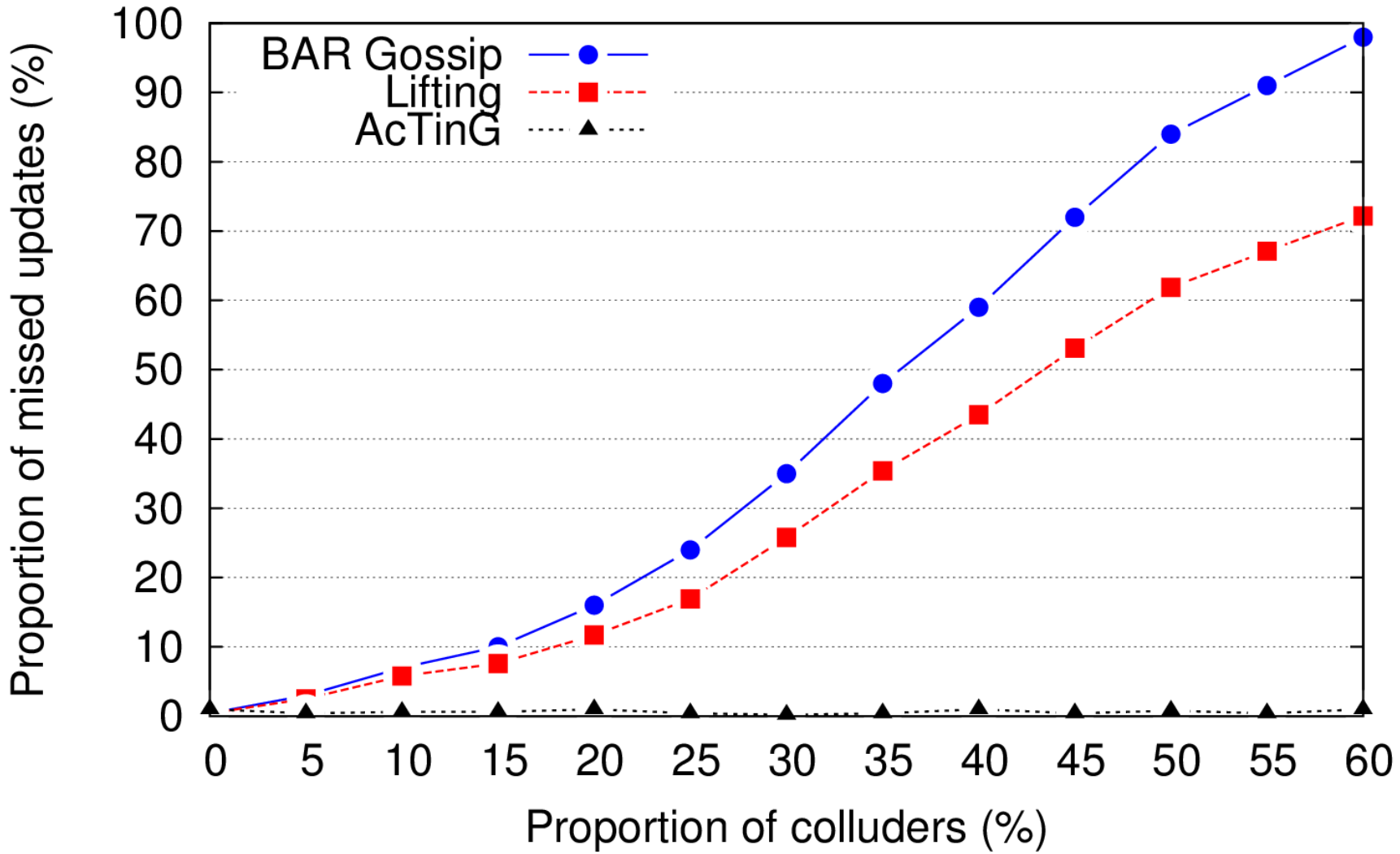
Measuring the Impact of Colluders

Colluders execute every possible deviation that increases their benefit.

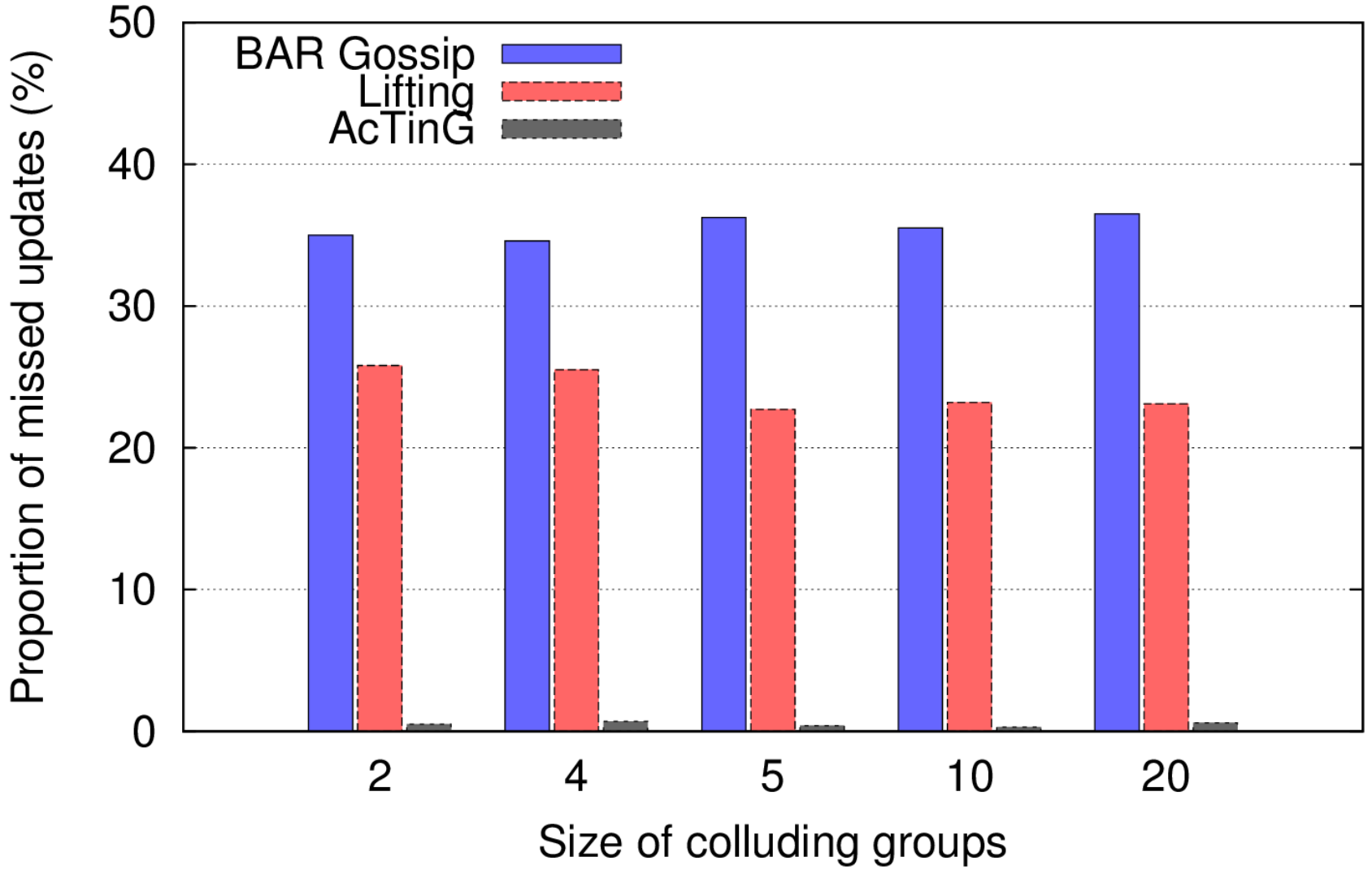
- **BAR Gossip**:
  - No participation in the optimistic push protocol,
  - Exchange of updates in priority between colluders.
- **LiFTinG**:
  - No participation in the verification protocols,
  - Blames are affected to correct nodes, forcing the administrator to exclude nobody (or to exclude a lot of correct nodes),
  - Lot of possible deviations: serve less updates than asked, contact less nodes...

A given proportion of the audience is made of colluders. We measure their impact on correct nodes:

- When they are in a single group,
- When they are divided in smaller independent groups.



15% of colluders in the system are enough for correct nodes not to be able to receive the content correctly.



The size of colluding groups does not change our observation. Even groups of 2 nodes degrade performance noticeably.

Goals

Given the current vulnerabilities of colluding rational deviations, we aim at designing the first gossip-based content dissemination protocol such that:

- Correct nodes **correctly receive the updates,** and **are never expelled,**
- A rational node whose deviations impact correct nodes' experience is eventually suspected by all correct nodes.

Key Idea 1: Deterministic Behavior

We can predict how a node must interact with other nodes.

- Nodes are roughly synchronized, and time is structured as a sequence of rounds in which nodes exchange updates.
- A PRNG guide associations between nodes.
- Define deterministic interactions (e.g., updates exchange, associations, etc.) between nodes.

However, this is not enough, because rational nodes can choose not to initiate exchanges.

Key Idea 2: Accountability

→ Nodes are accountable for what appears in their log. They must use it to log their interactions with other nodes, and the updates they receive/send.

To do so:

- Nodes are provided a session key pair consisting of a public and a private key, that they use to sign messages.
- Each node uses a secure log, that is tamper evident and append only, to record the messages it sent, or received. We build on the one described in PeerReview[Citation].

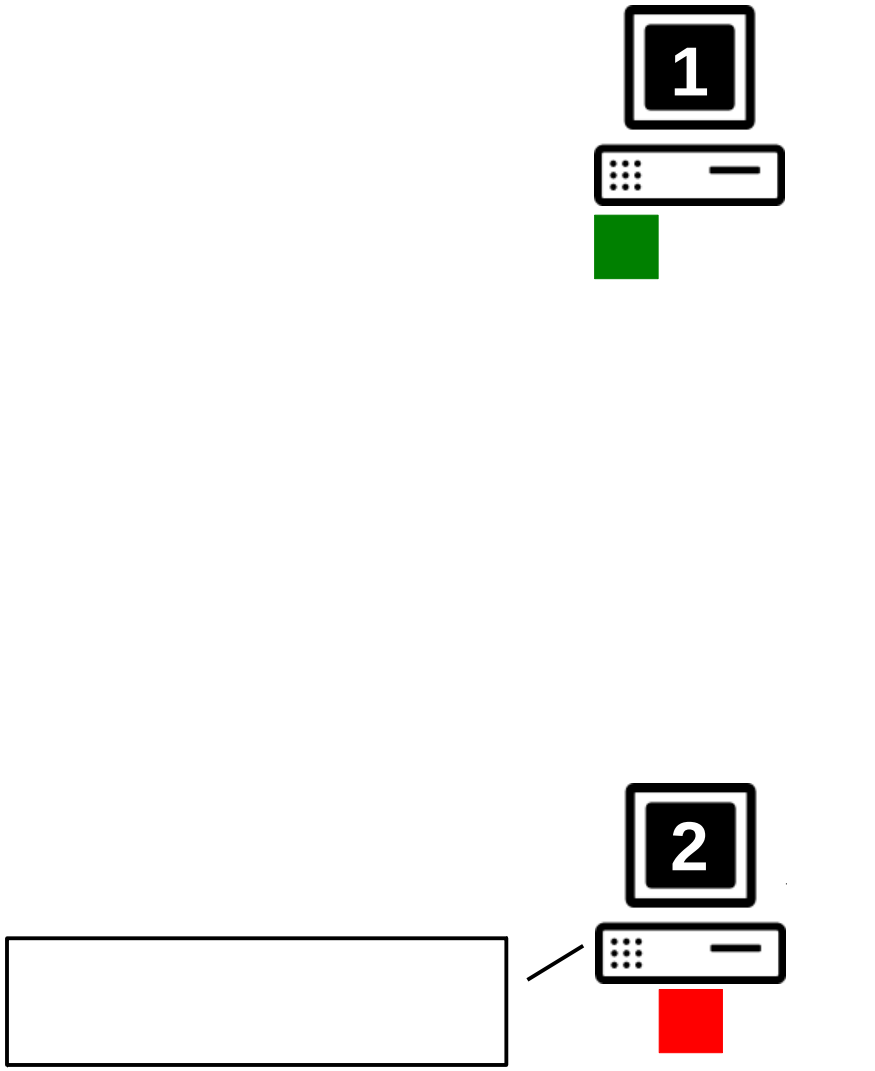
If any node can trust and check the information of the log of a node it is interacting with, the latter will be **obliged to send to its partners the updates it has**, and **to receive the updates it is missing**.

However, it is not enough to exchange logs. Colluders could maintain different logs and still deviate.

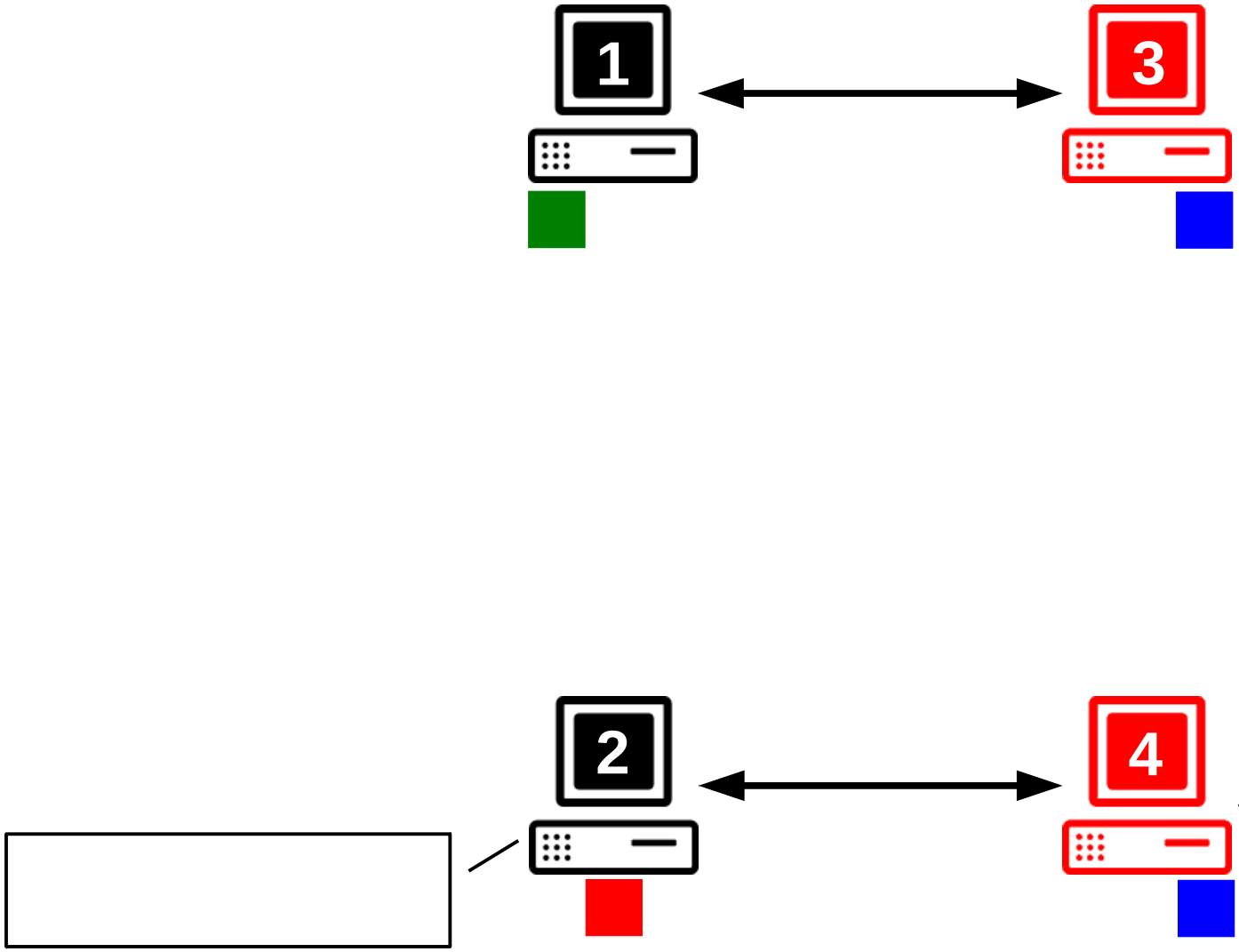
Key Idea 3: Audits

- During an audit, a node obtains the logs of several nodes that exchanged updates, and compare them, to detect deviations.
- Audits must be **verifiable**, **random yet unpredictable**.
  - Verifiable: if a rational node decides not to audit other nodes, it should be eventually discovered by correct nodes.
  - Unpredictable: if a rational node can predict whether or not it will be audited, then it would choose when to execute its deviations.
- Each time they decide to deviate from the protocol, nodes take a (high) risk of being detected, and cannot know if they will be detected. If so, they will be evicted from the system.

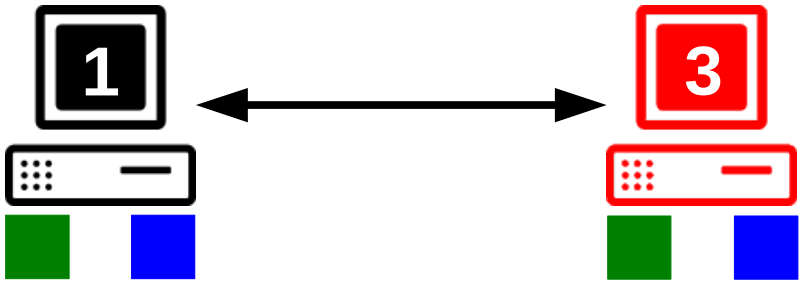
Audit example

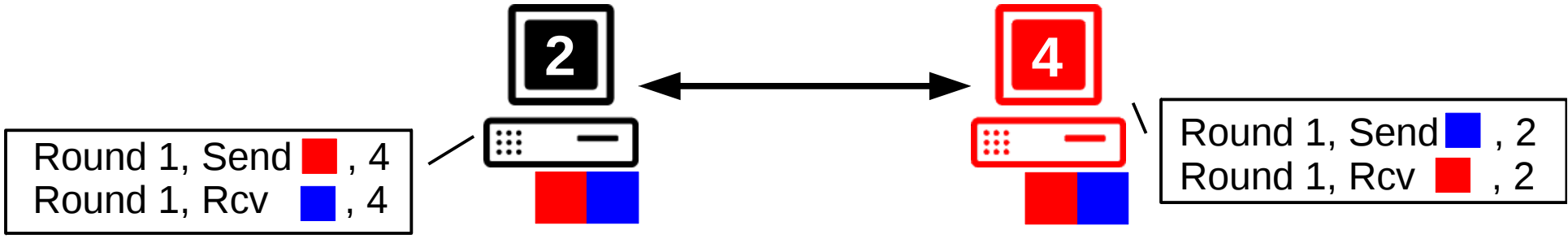


Audit example

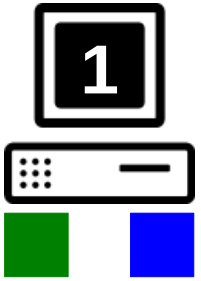


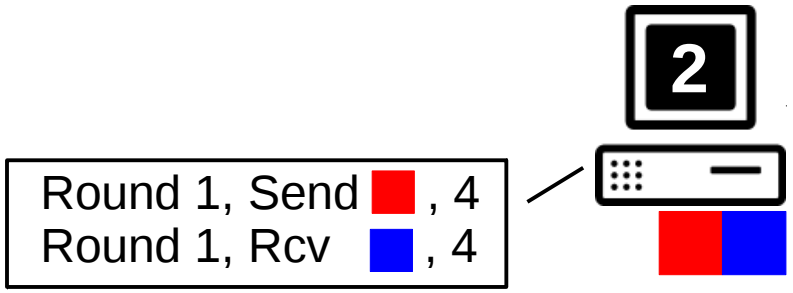
Audit example

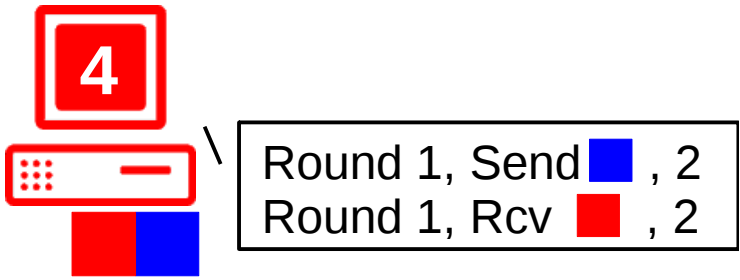




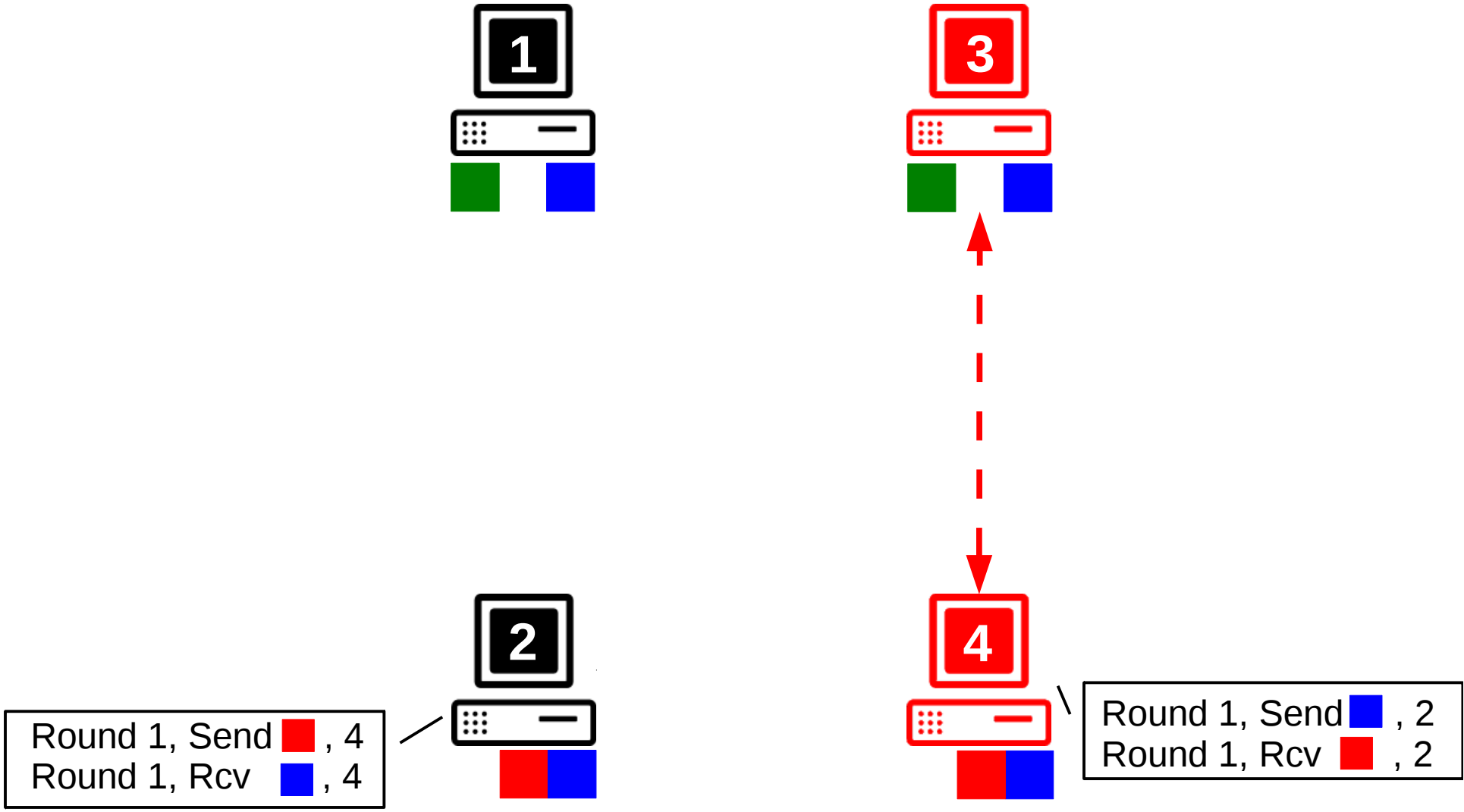
Audit example



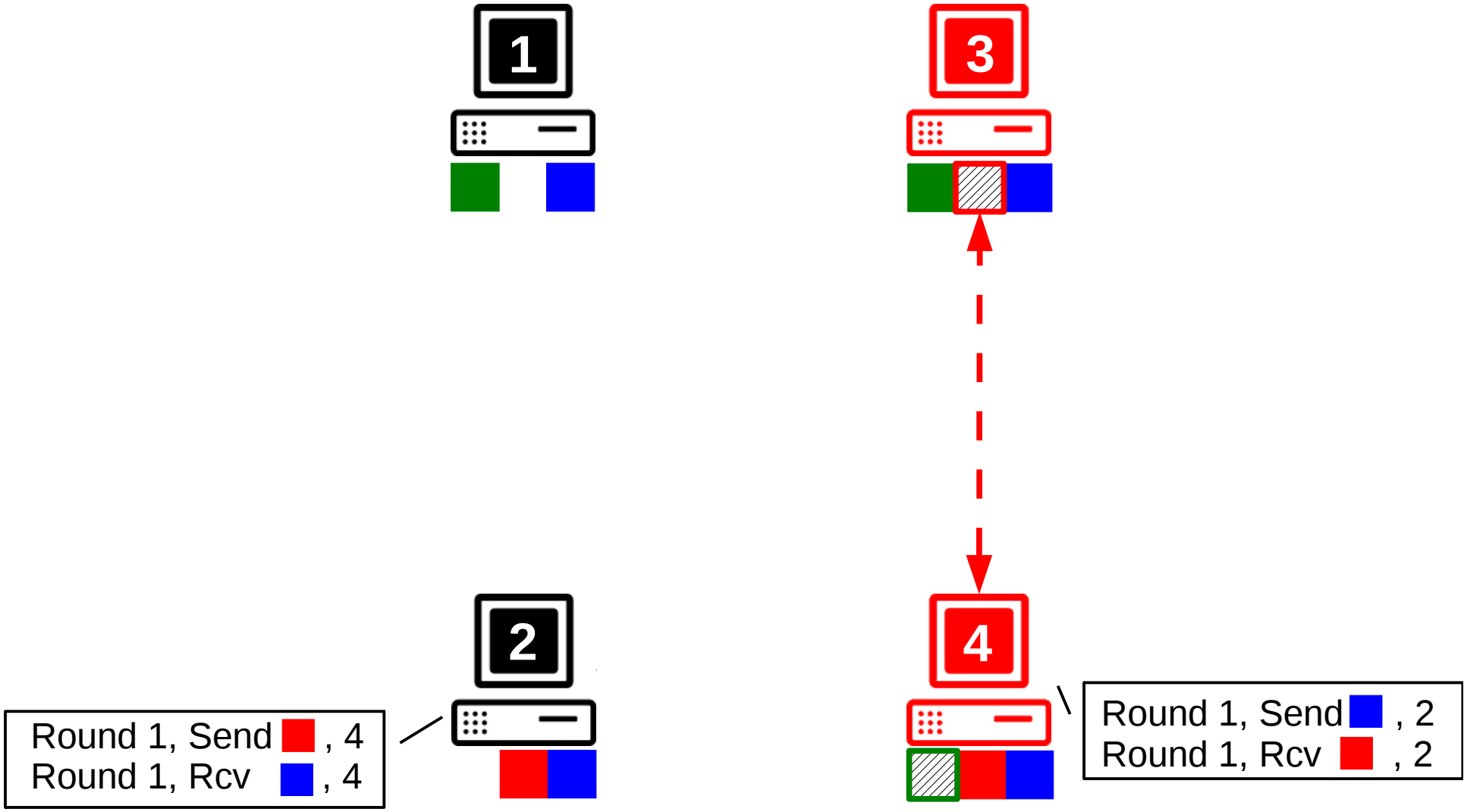




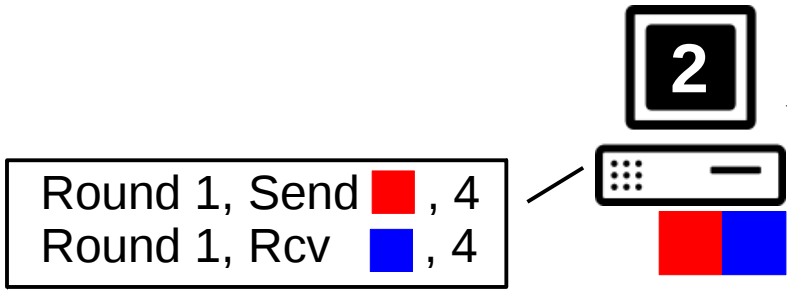
Audit example

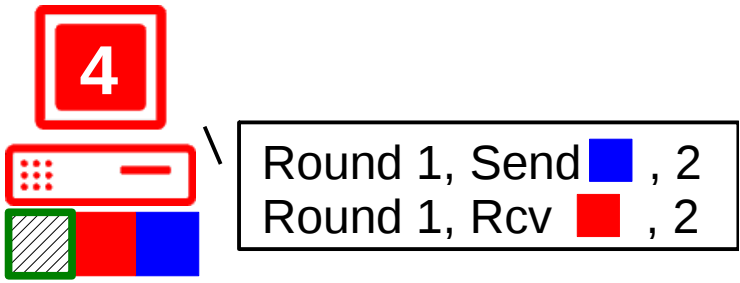


Audit example

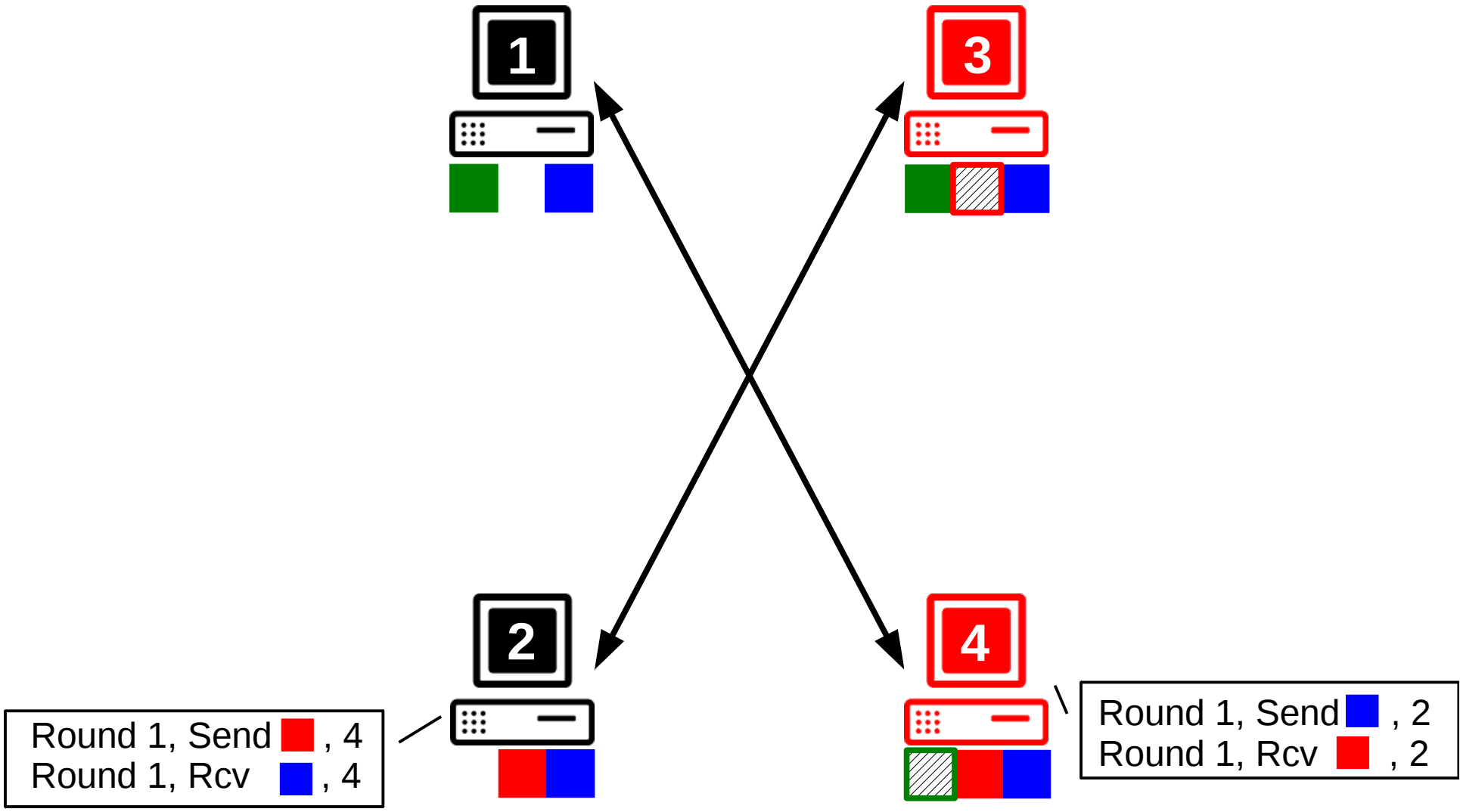


Audit example

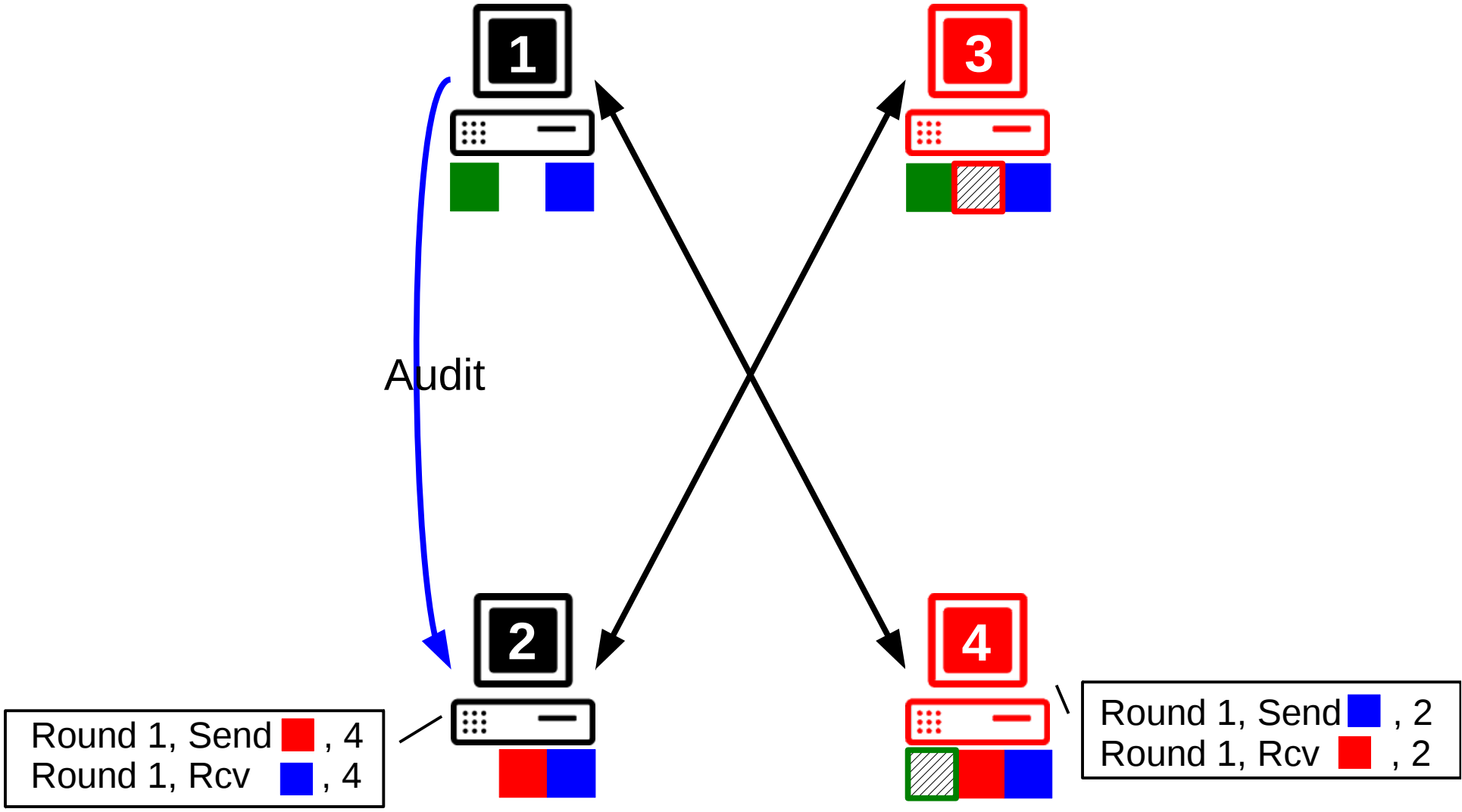




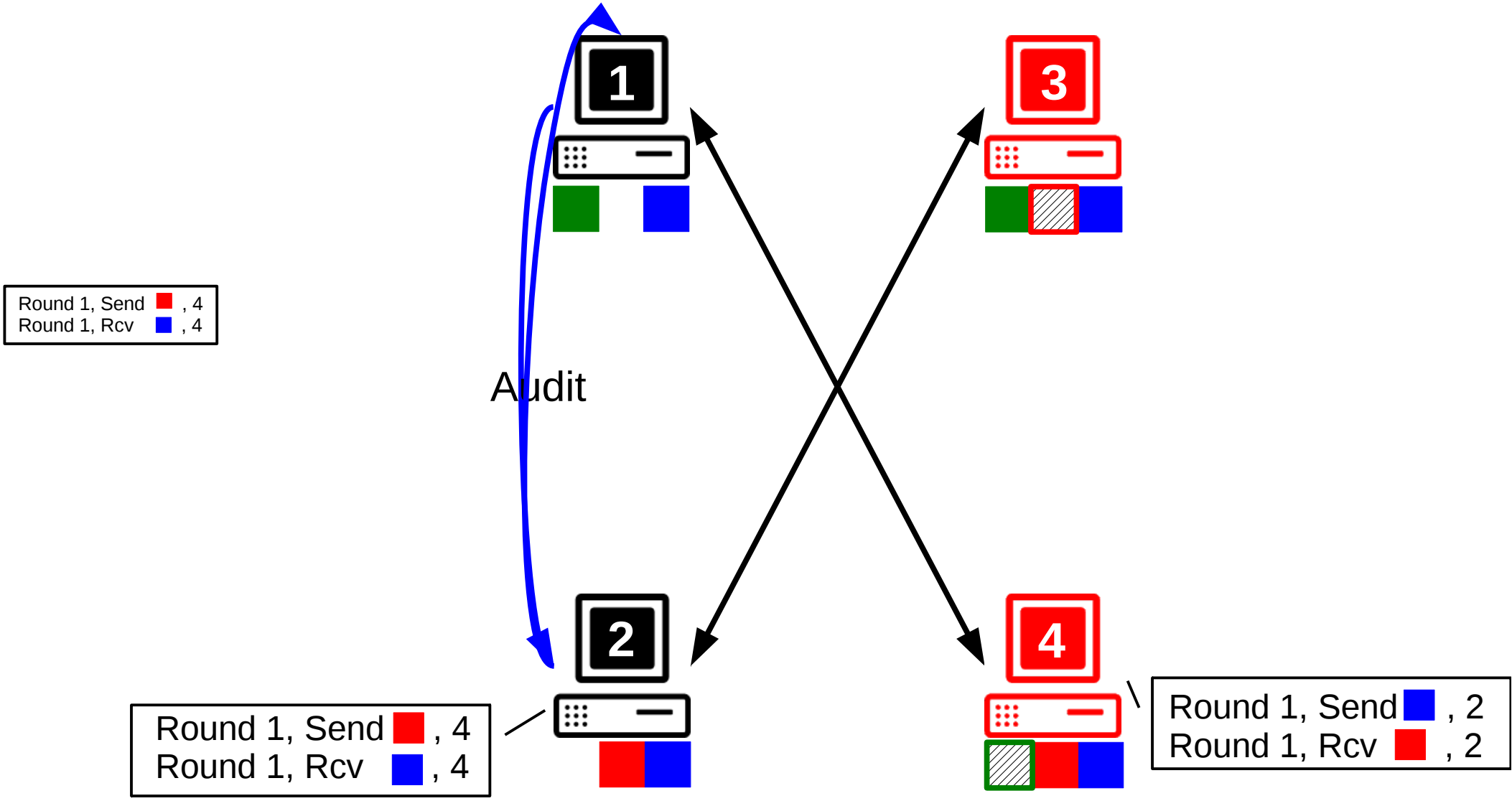
Audit example



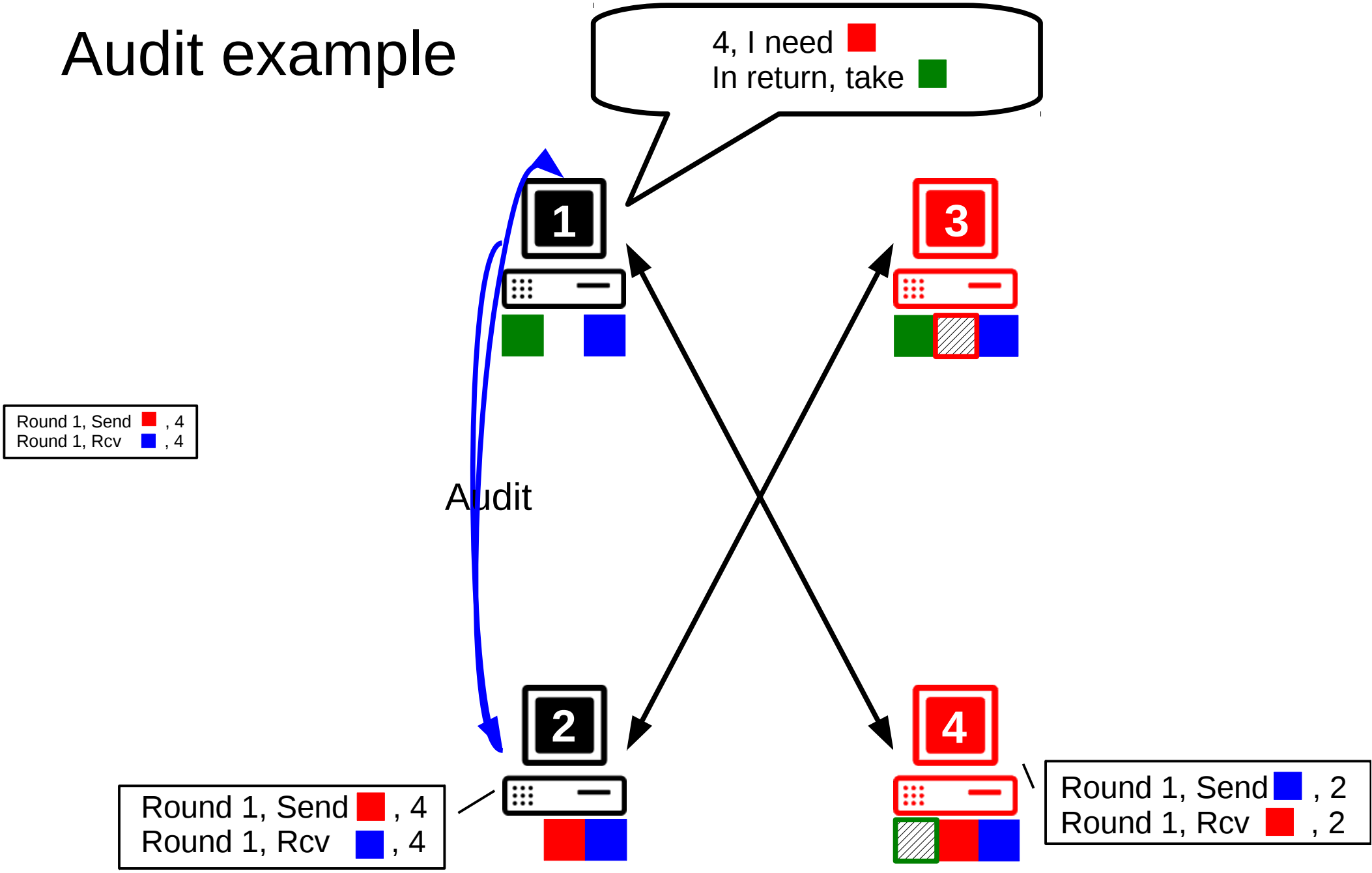
Audit example



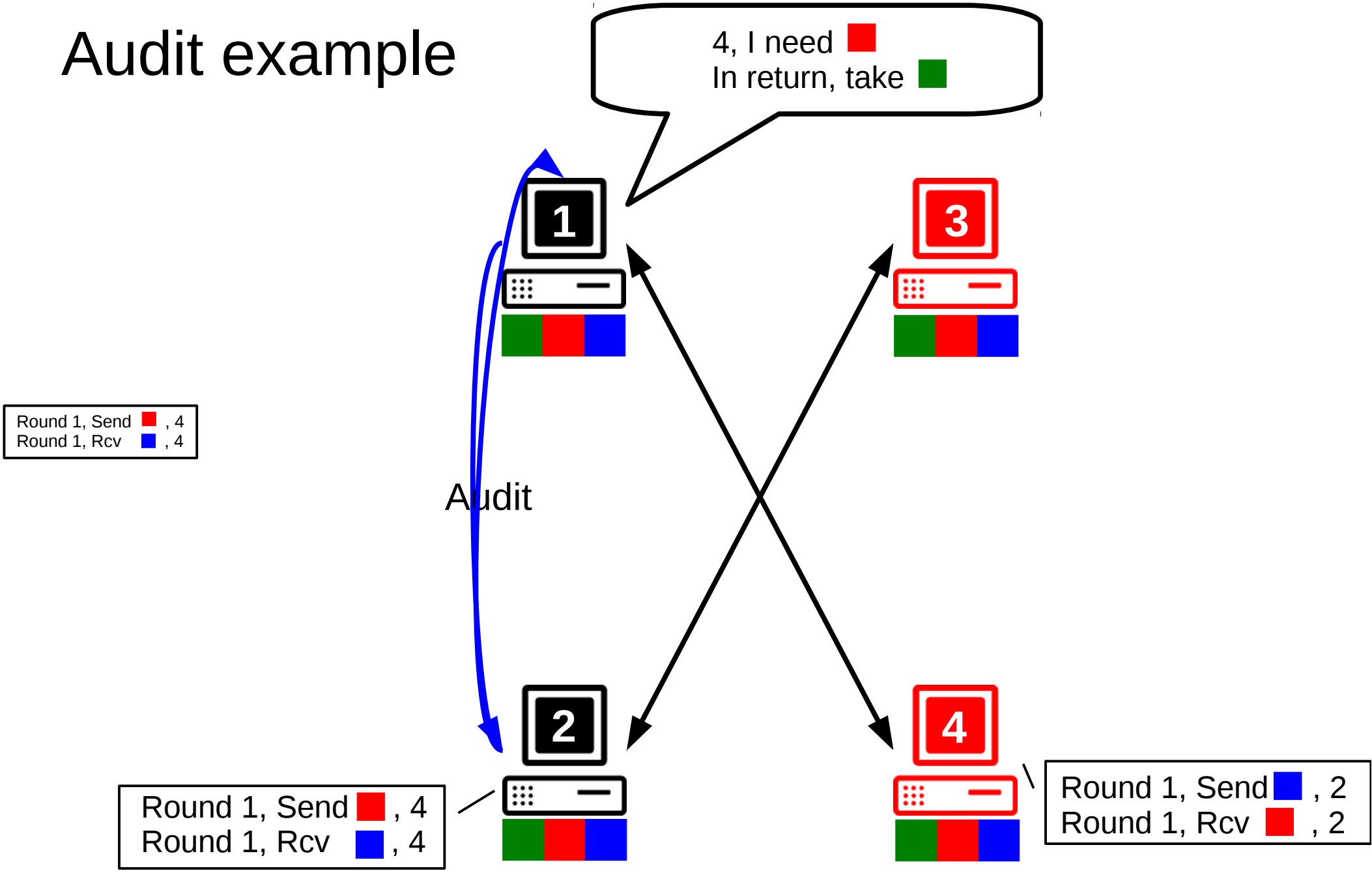
Audit example



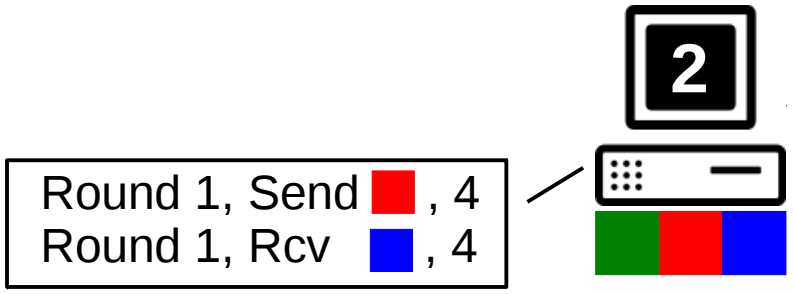
Audit example

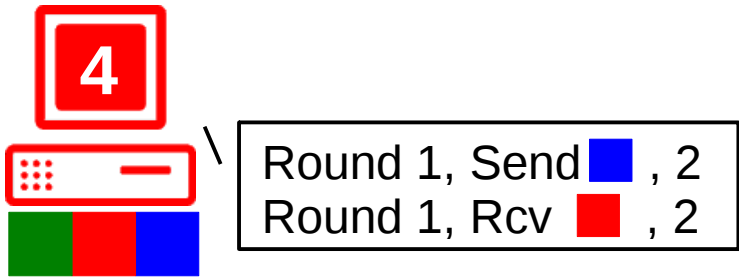


Audit example



Audit example





AcTinG's properties

Finally, we ensure that a node:

- Interact with other nodes, and cannot avoid to do so;
- Receives the updates it did not receive officially;
- Send the updates its partners did not receive officially;
- Correctly audit its partners when it has to.

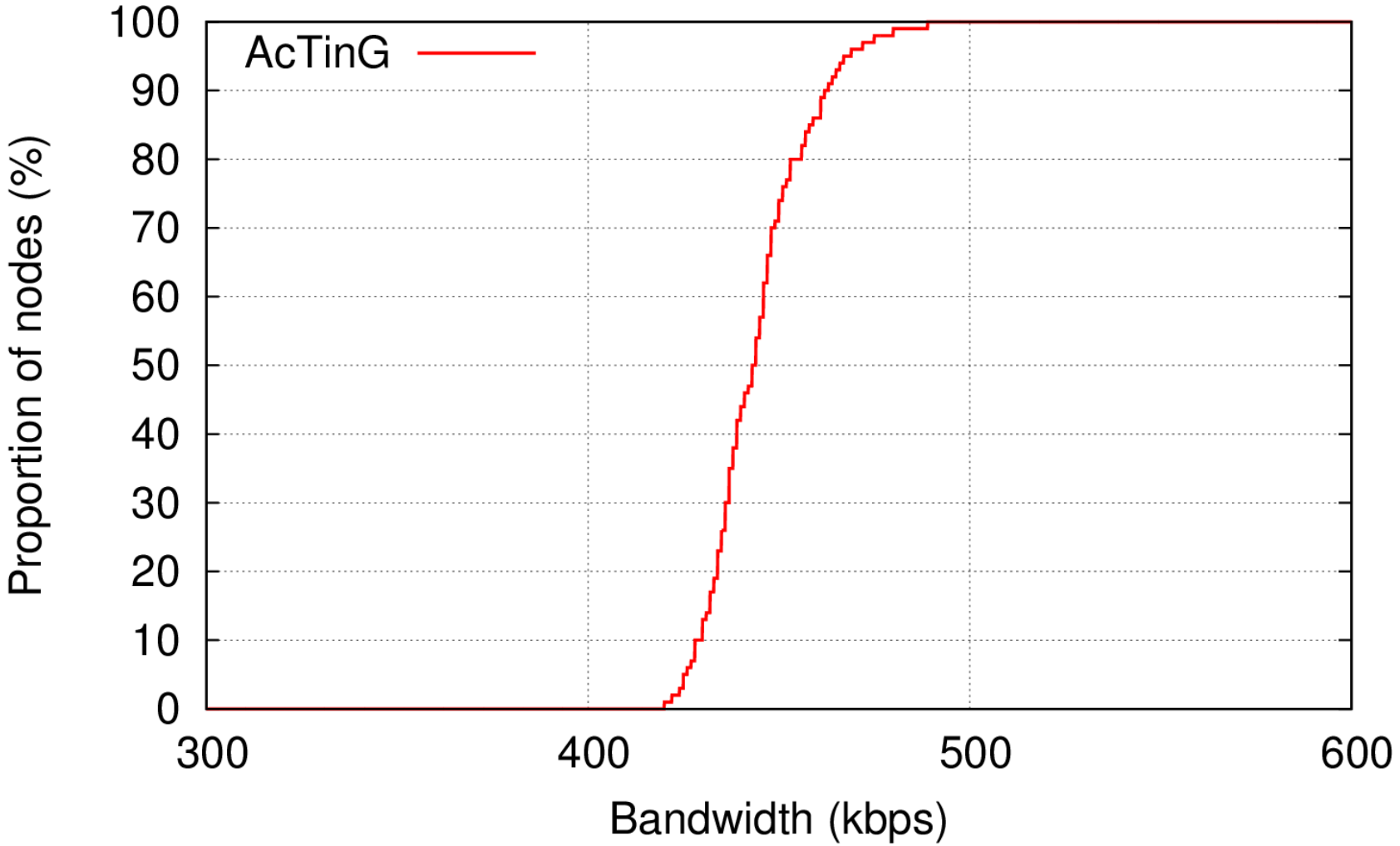
Implementation

- Java
- 400 machines
-

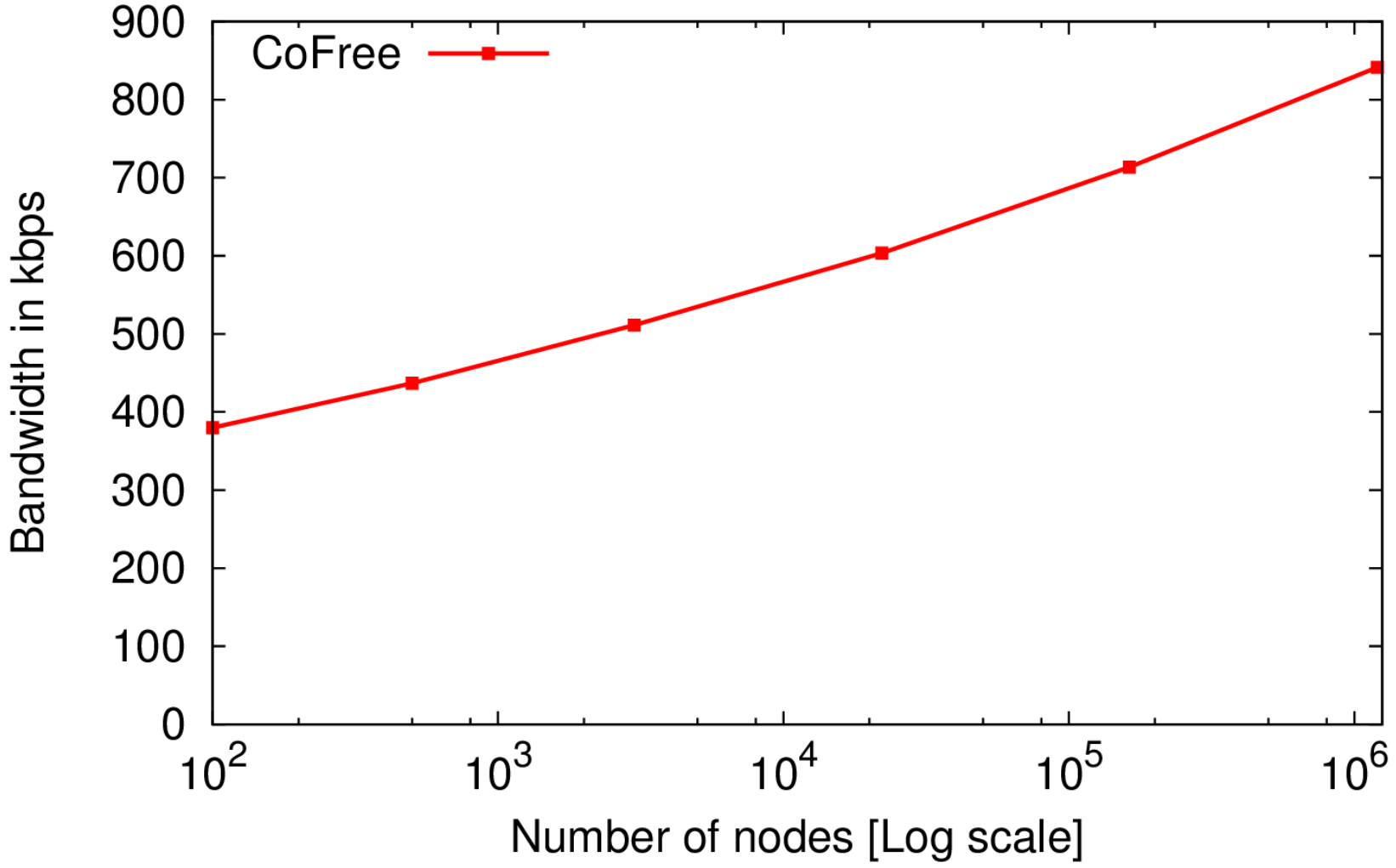
Probabilistic audit

- Audits are triggered randomly, with a given probability. Are deviations correctly detected?

How much does AcTinG cost, with a 300 Kbps stream?



Memory usage



Questions?