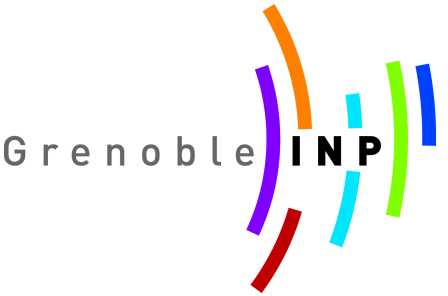
AcTinG:
Accurate Freerider Tracking in Gossip

Sonia Ben Mokhtar, **Jérémie Decouchant**, Vivien Quéma



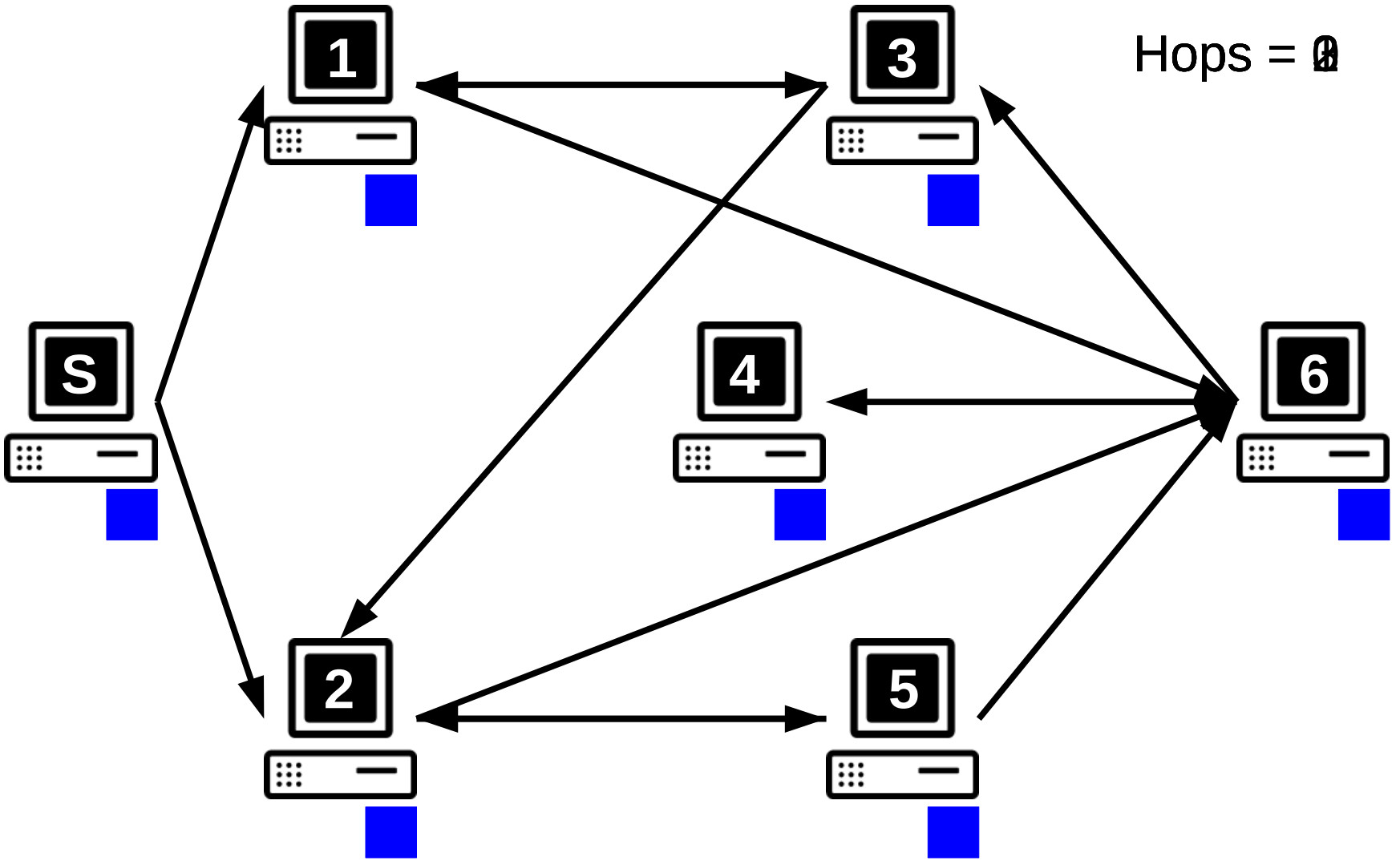
Peer-To-Peer Content Sharing

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
- Limited costs (e.g., bandwidth)
- No dedicated servers

Content Dissemination using Gossip



Rational Nodes

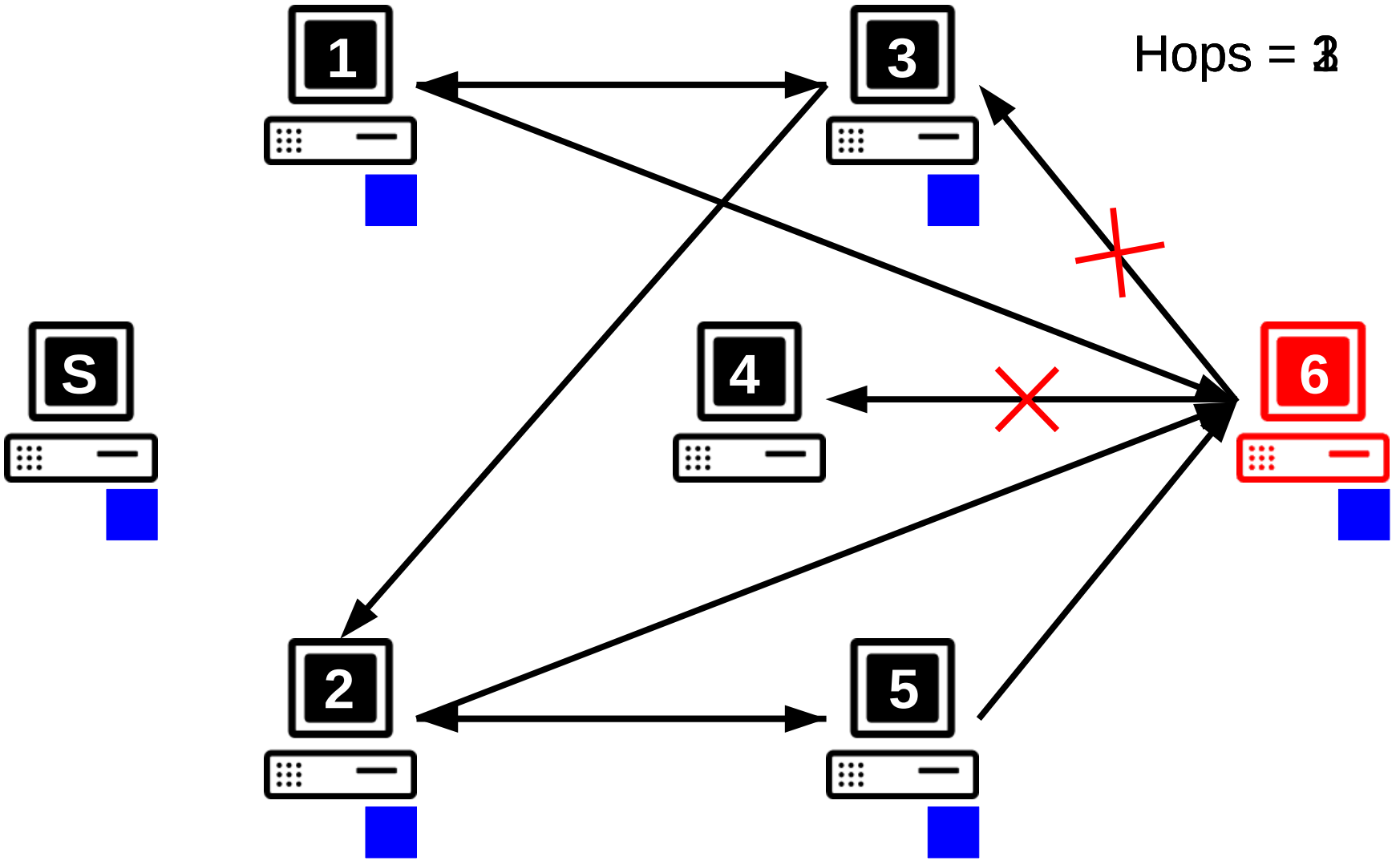
Rational nodes aim at getting the content at the lowest possible overhead (they are not Byzantine nodes).

They aim at **maximizing their own benefit**, according to:

- Stream quality
- Communication overhead
- Computation overhead
- Risk

Their strategy can potentially harm the good dissemination of updates in the system.

Gossip with a Rational Node



Problem: Colluding Rational Nodes

Several solutions have been designed to handle solitary rational nodes.

However, rational nodes may collude to:

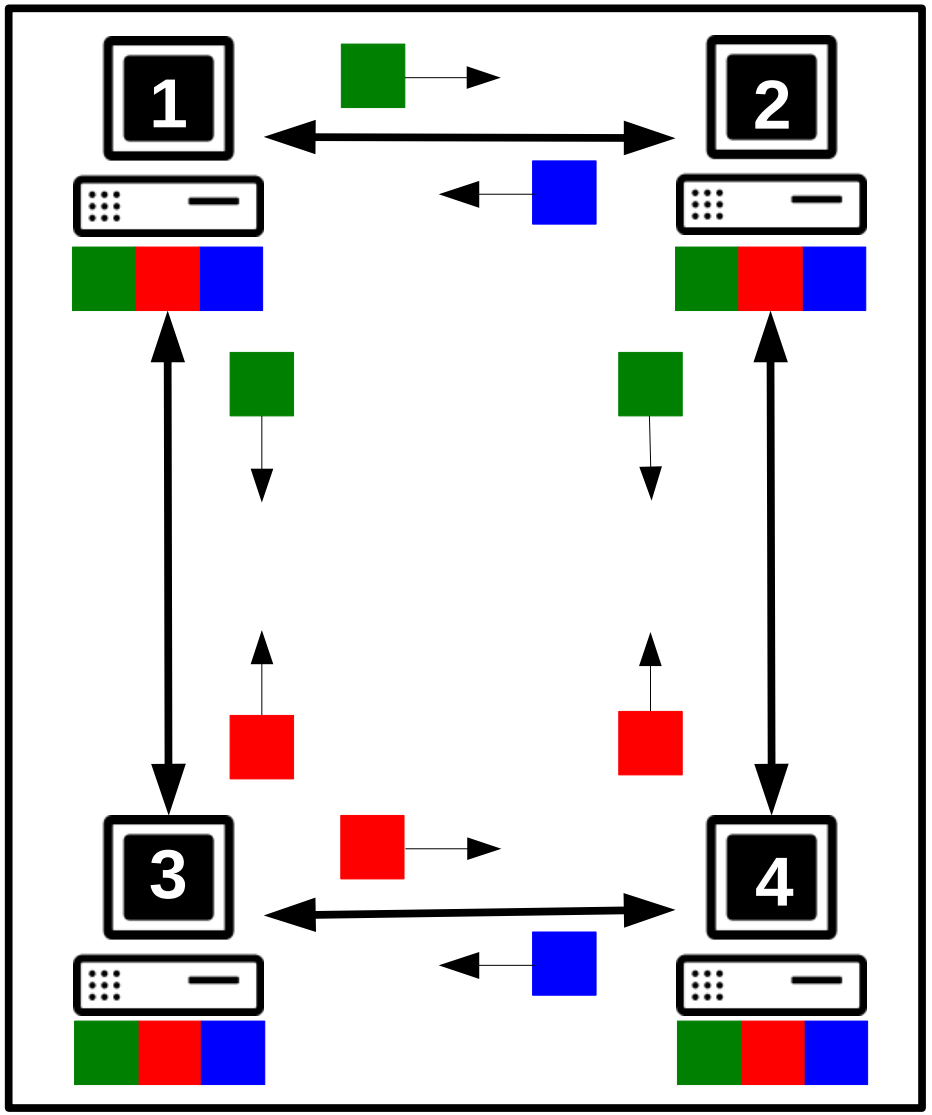
- escape the mechanisms designed to detect individual rational nodes,
- increase their benefit.

Real-life collusions have been observed in Maze, a file sharing system, and until now, could not be prevented.

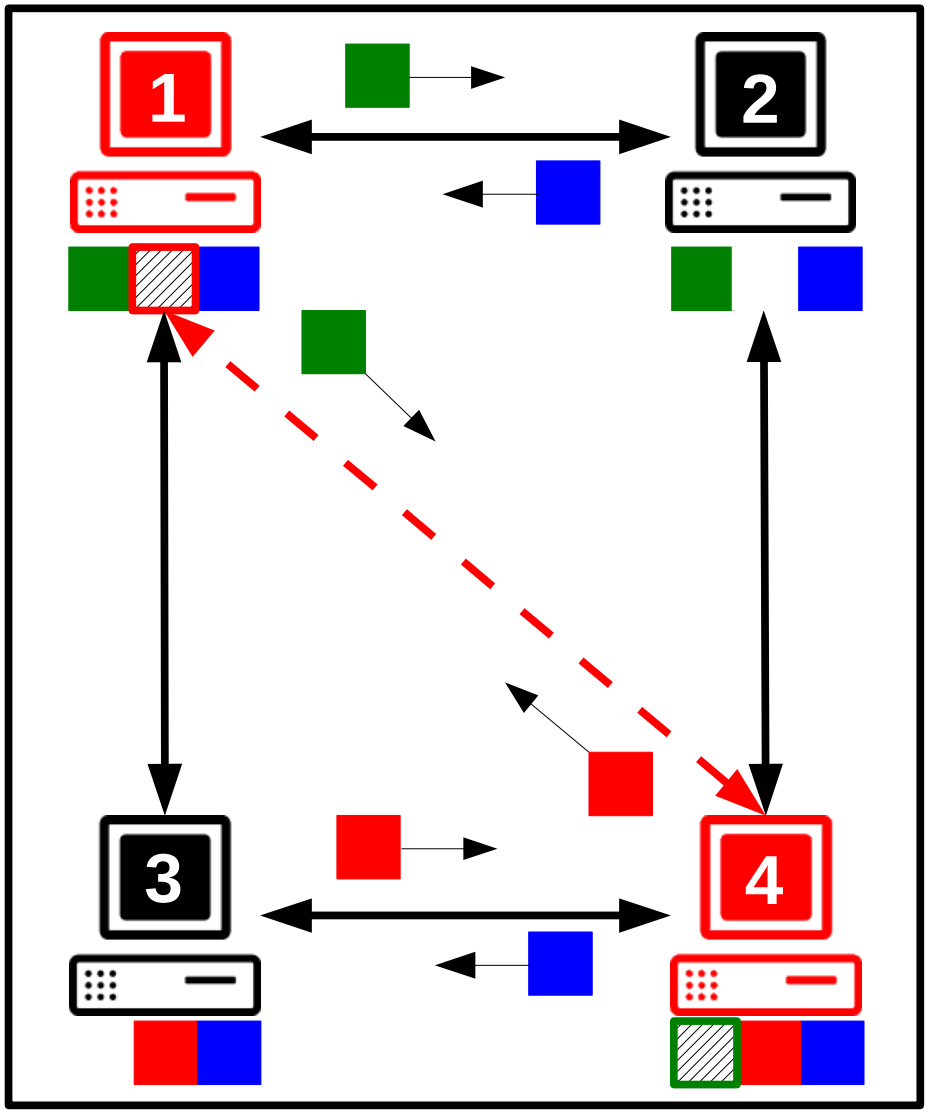
Outline

- Impact of collusions on Gossip
  - Example with symmetric exchanges
  - Results of experiments on existing protocols
- Key ideas of AcTinG
- Performance evaluation

Colluders with Symmetric Exchanges



Without colluders

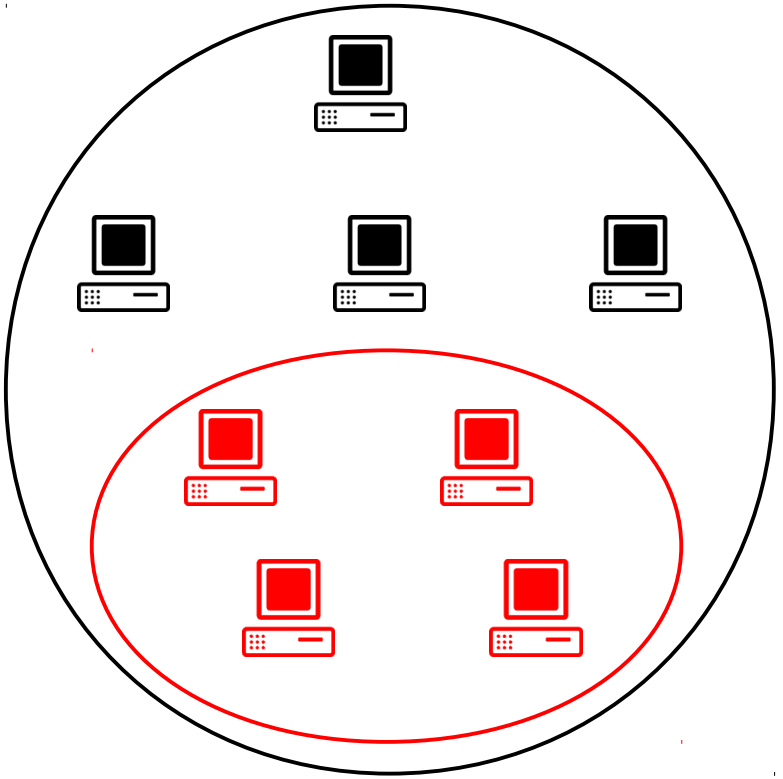


With colluders

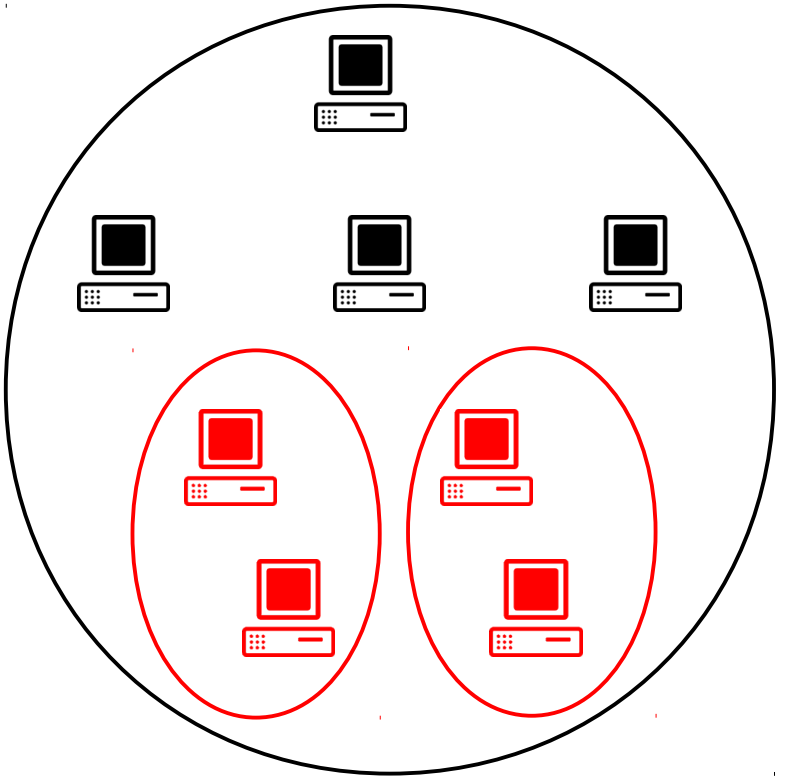
Impact of Colluders : Scenarios

A given proportion of the audience is made of one or several groups of colluding nodes.

We measure how well correct nodes receive the stream.



Experiment 1



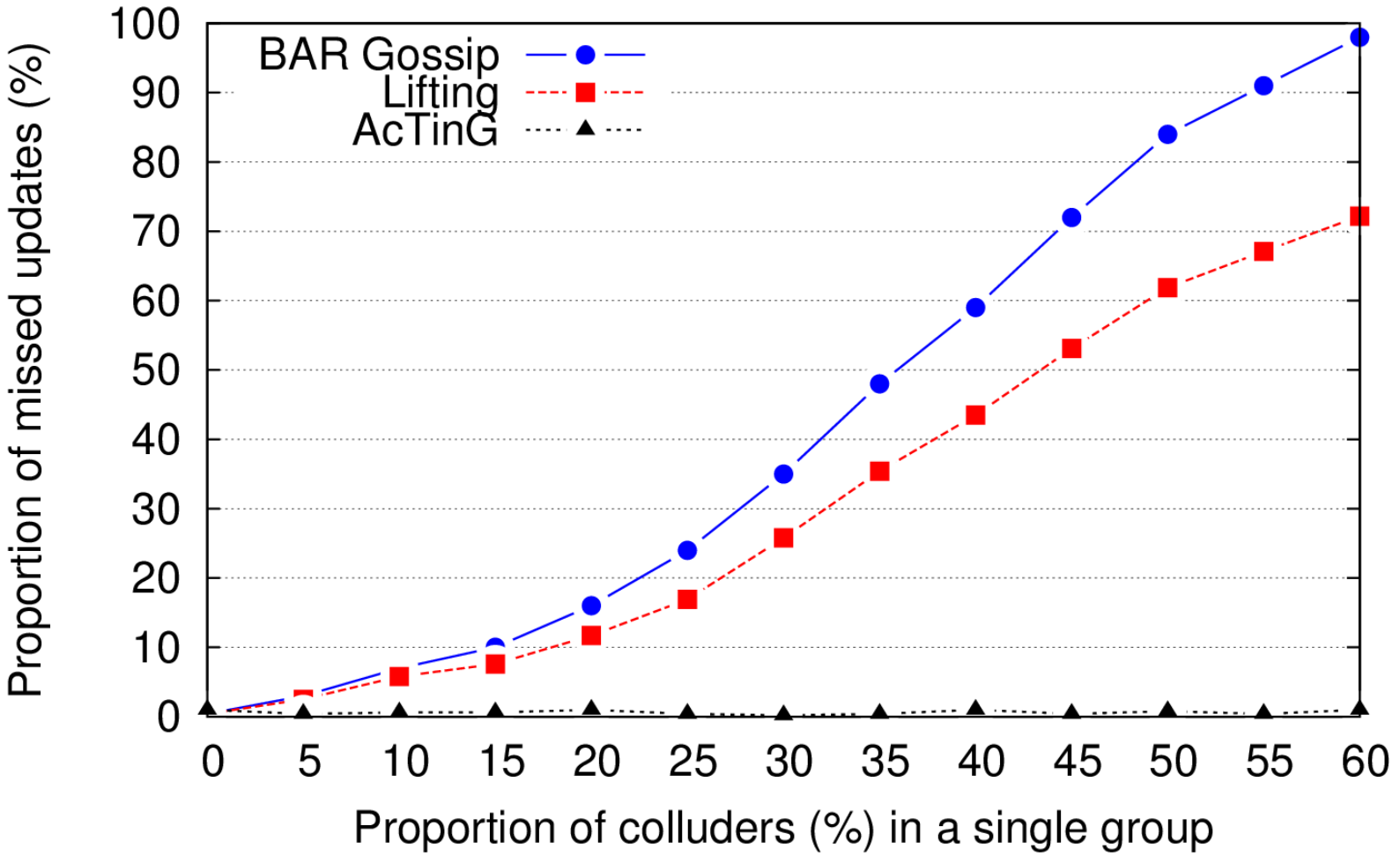
Experiment 2

Measuring the Impact of Colluders

Colluders execute every possible deviation that increases their benefit, or protect them.

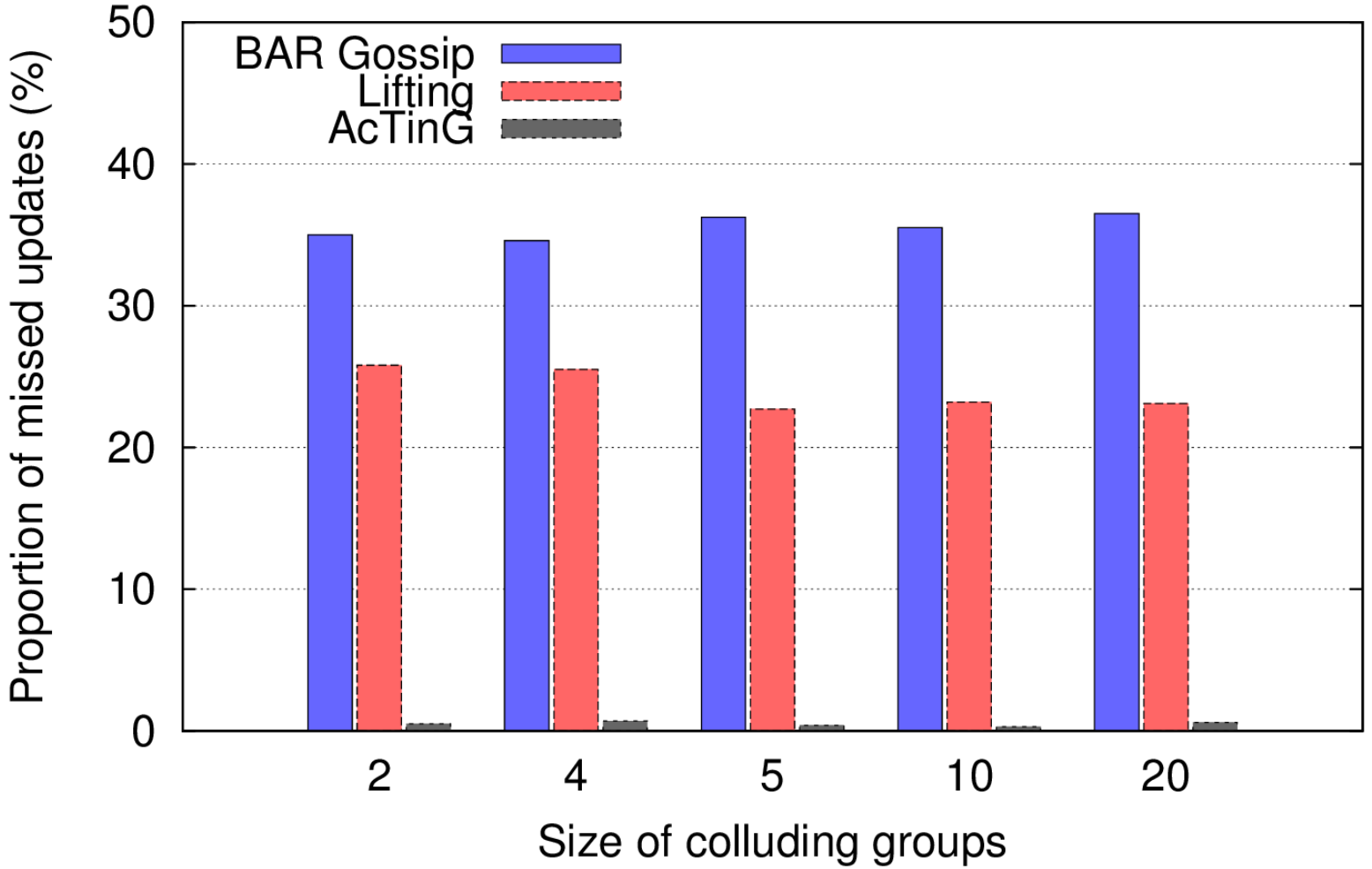
- **BAR Gossip**
  - No participation in the optimistic push protocol,
  - Exchange of updates in priority between colluders.
- **LiFTinG**
  - No participation in the verification protocols,
  - Blames are affected to correct nodes, forcing the administrator to exclude nobody, or to exclude a lot of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

Exp. 1: Single group of colluders



15% of colluders → 10% of the updates are not received.

Exp. 2: 30% of colluders in several groups



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

Impact of Colluders: Conclusion

Rational nodes can collude and increase their benefit, while not being detected.

They receive correctly the stream, but correct nodes do not.

Even small groups can harm the dissemination of updates.

Our goals

Current protocols can be heavily harmed by rational collusions.

We aim at designing the first gossip-based protocol such that:

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

Outline

- Impact of collusions on Gossip
- Key ideas of AcTinG
- Performance evaluation

Key Idea 1: Deterministic Behavior

It is possible to predict how a node interact with other nodes.

- Nodes are synchronized, and time is structured as a sequence of rounds.
- A PRNG guide associations between nodes.
- Sub-protocols are deterministic.

Limitation

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

Key Idea 2: Accountability

It should be possible to check the behaviour of nodes, and make them responsible from their declarations.

- Secure log, tamper evident and append only, to record the messages sent, or received.
- Session key pair consisting of a public and a private key, that is used to sign messages.

Limitation

However, nodes could hide the updates they received from accomplices. To do that, they would maintain different log versions.

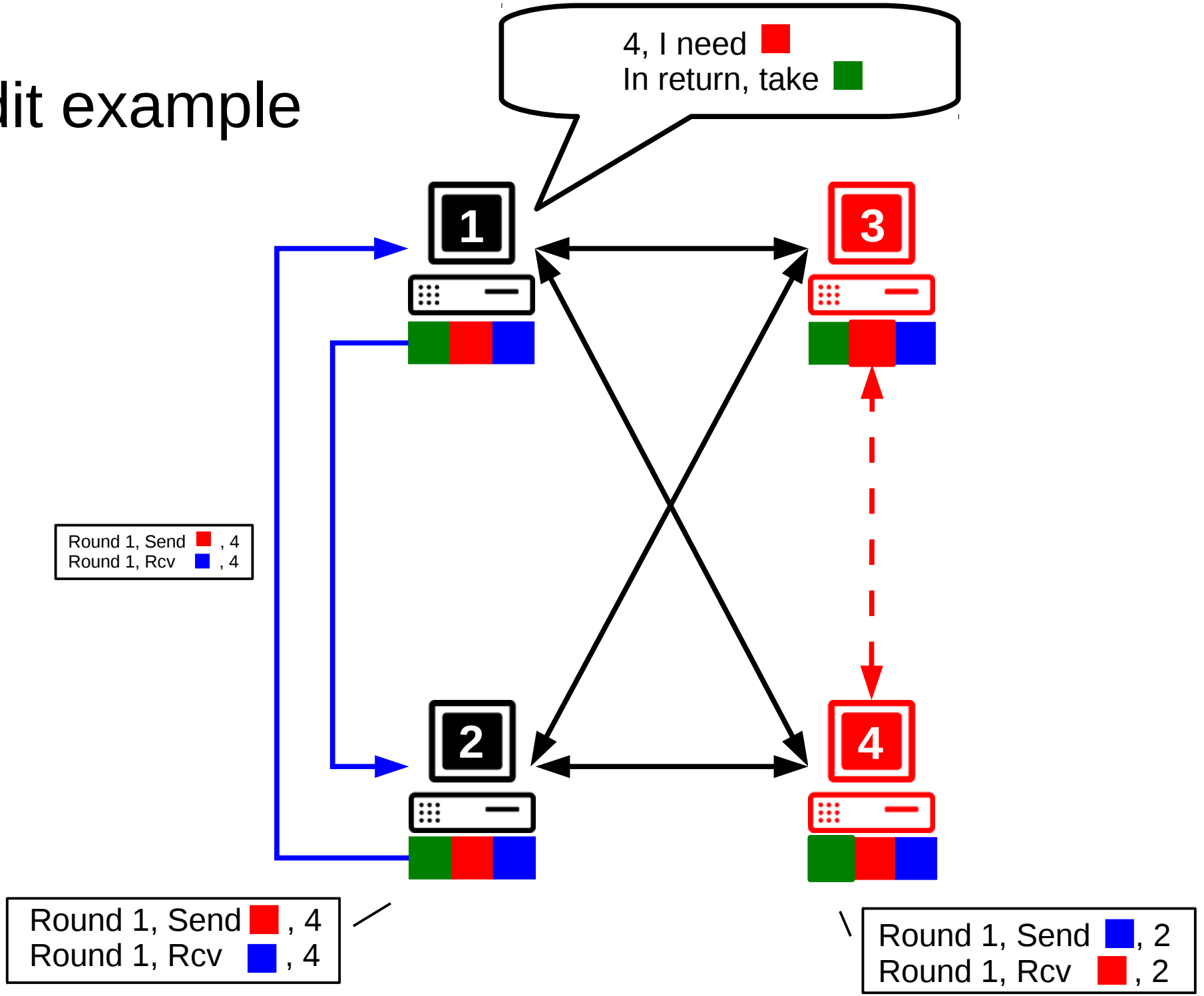
Key Idea 3: Audits

It should be possible to compare the declarations a node made to detect its lies.

- Audits which are **verifiable**, **random yet unpredictable**.
  - Verifiable: if a rational node decides not to audit other nodes, it should be eventually discovered by correct nodes.
  - Random: the overhead due to audits is decreased.
  - Unpredictable: if a rational node can predict whether or not it will be audited, then it would choose when to execute its deviations.

Deviating from the protocol means taking a high, and unpredictable, risk of being detected.

Audit example



AcTinG's properties

Finally, our protocol ensures that any node:

- Interacts with other nodes, and cannot avoid it;
- Receives the updates it did not receive officially;
- Sends the updates its partners did not receive officially;
- Audits its partners when it has to.

Nodes can still exchange updates off the record, but it is not interesting. They will be forced to receive them officially anyway.

Outline

- Impact of collusions on Gossip
- Key ideas of AcTinG
- Performance evaluation

Probabilistic audit

Audits are triggered randomly, with a probability of 5%.

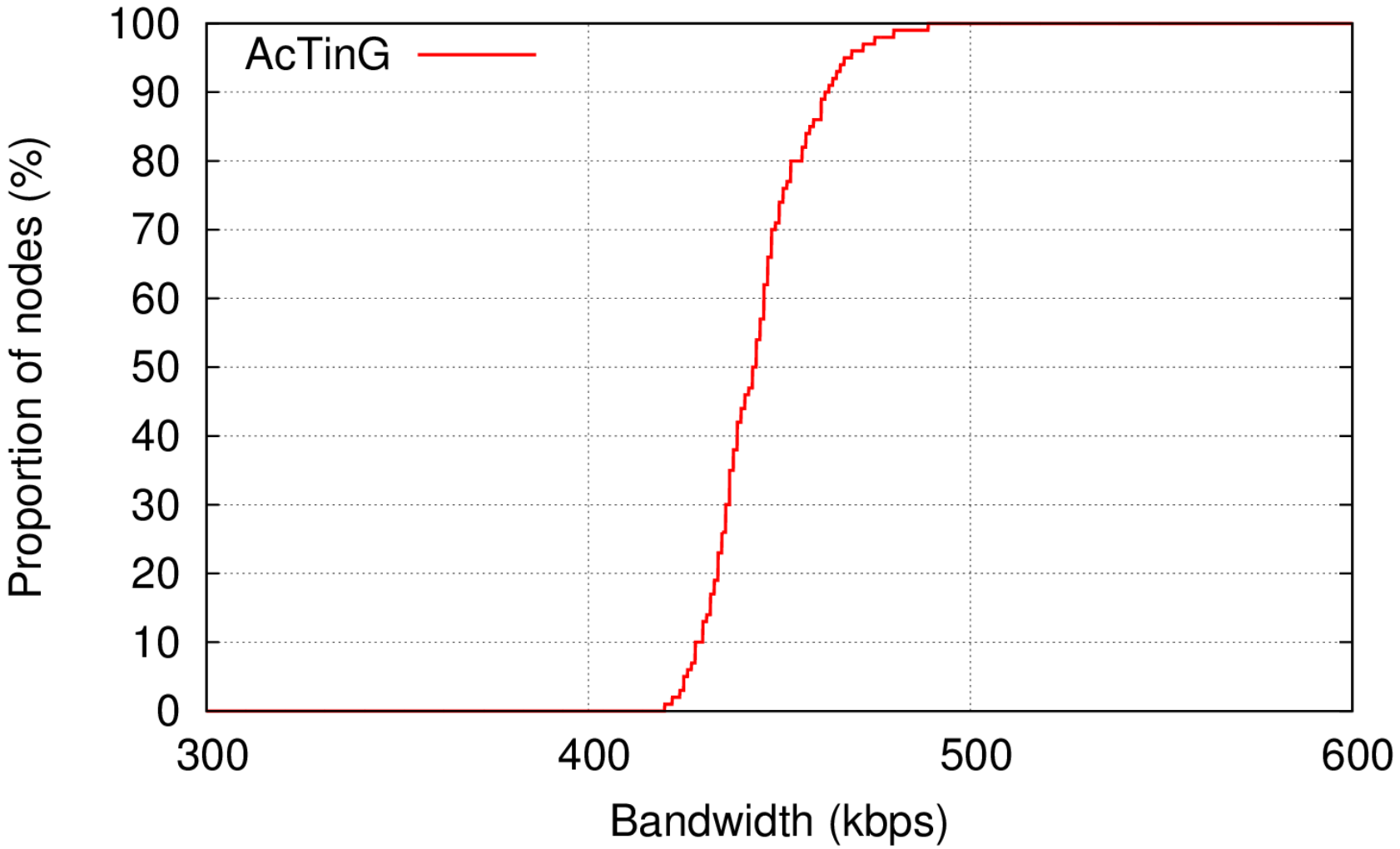
Updates expire N rounds after their release.

Audits concern the entries of the last N rounds, which means that a deviation can be observed by an auditing node during RTE rounds.

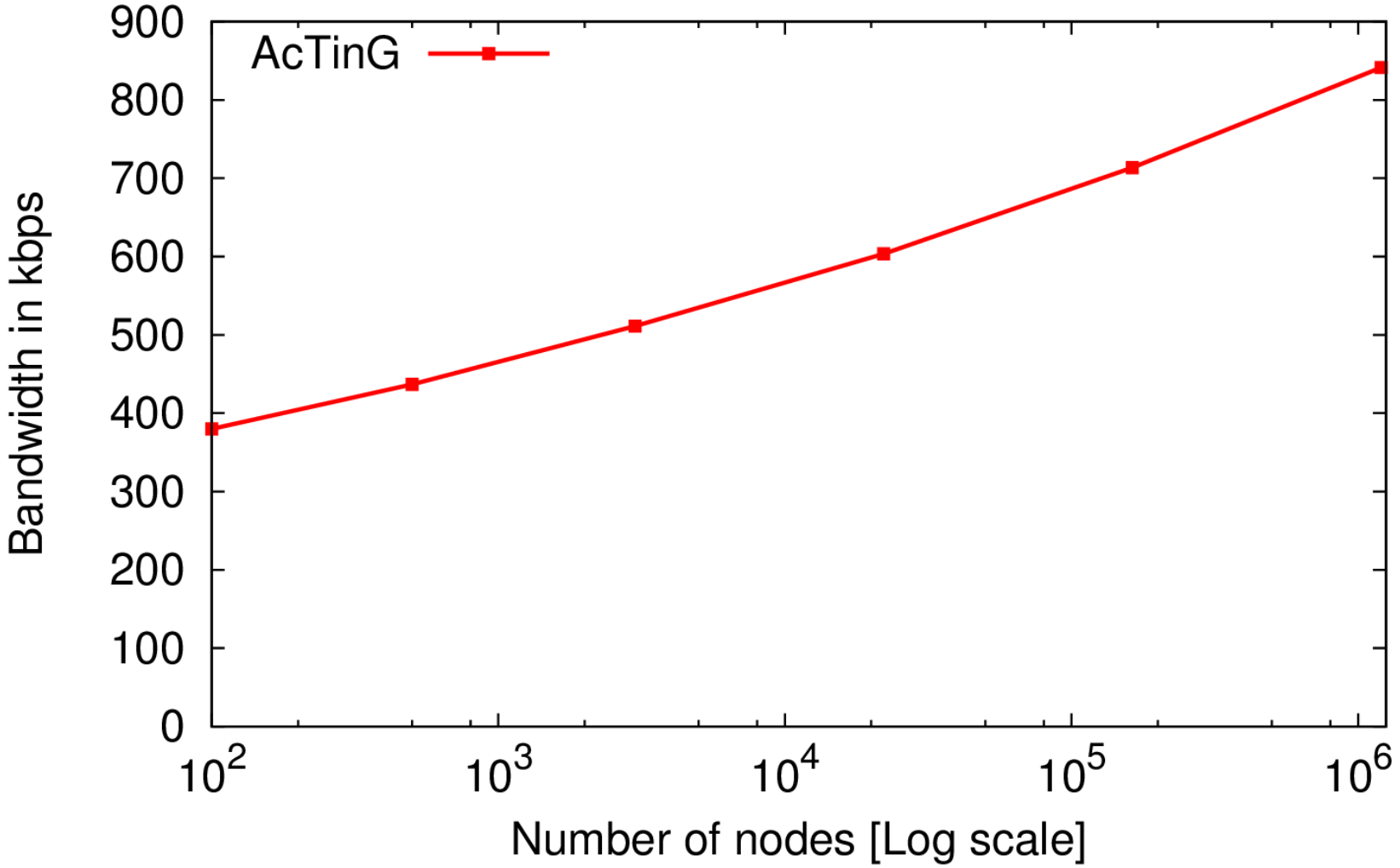
Our evaluations show that when 10% of the audience collude in a single group, the probability for a deviation to be detected is equal to 60%.

The only possible deviation bring a long term gain of 3% (if never detected).

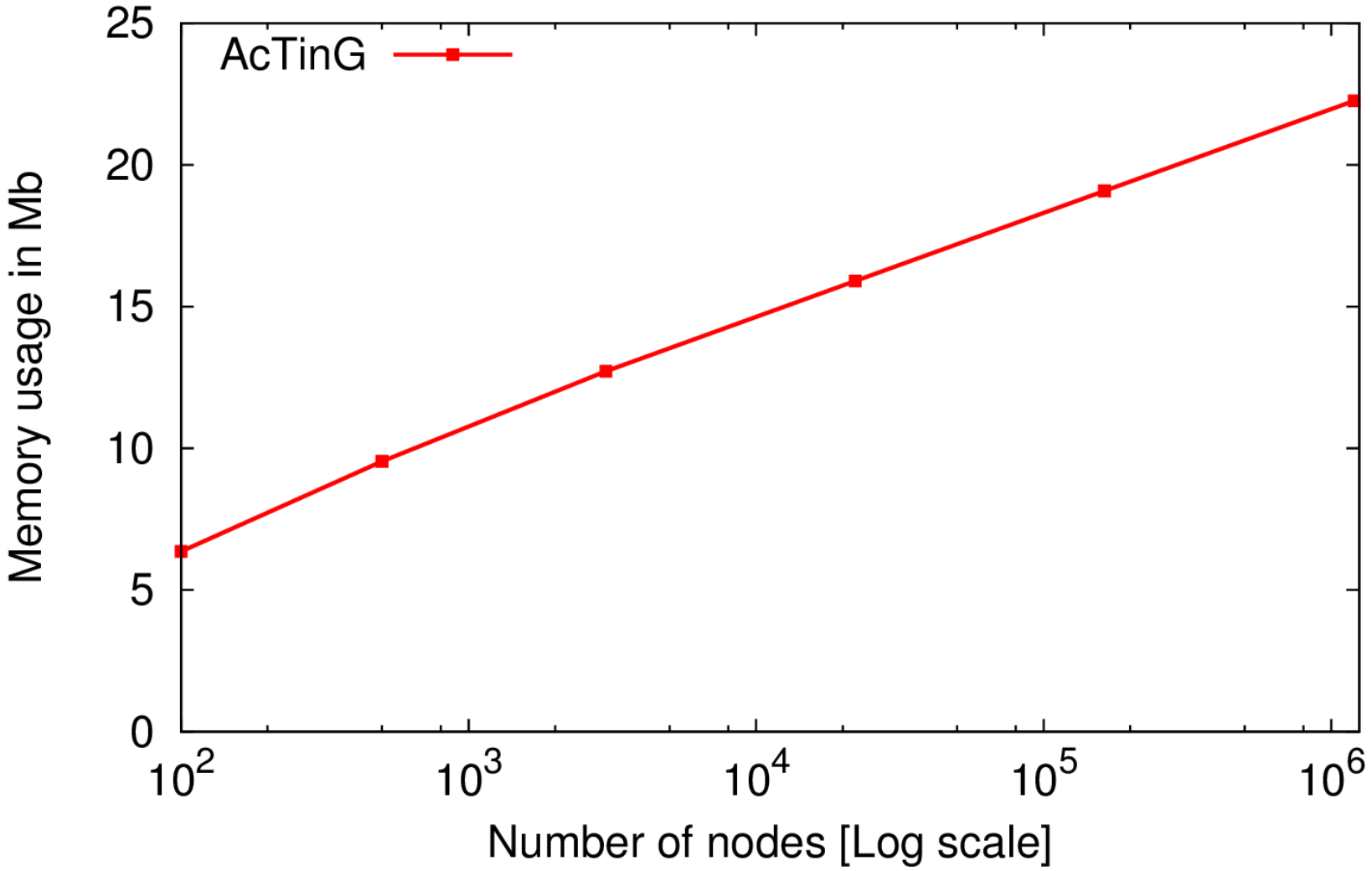
AcTinG's overhead - 300 Kbps stream



Scalability



Memory usage



Questions?

AcTinG:
Accurate Freerider Tracking in Gossip

Sonia Ben Mokhtar, **Jérémie Decouchant**, Vivien Quéma

Hello, my name is Jeremie Decouchant, I'm a French PhD student.

In this talk, I will present the principles of AcTinG, a novel gossip protocol.

It is a joint work with Dr. Sonia Ben Mokhtar, and Prof. Vivien Quéma.

Peer-To-Peer Content Sharing

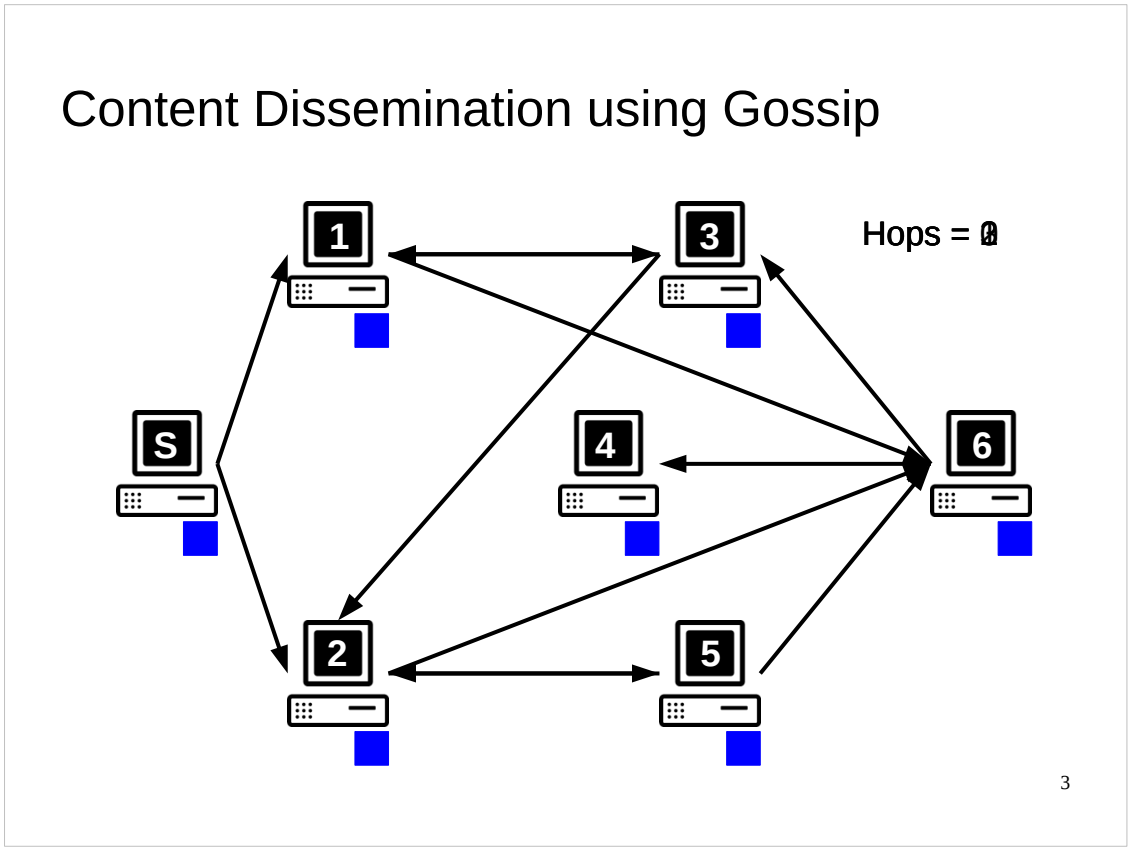
Content sharing applications account for a large portion of traffic over the Internet.

Relying on the P2P paradigm provides the following advantages :

- Robustness to failures, or churn
- Scalability
- Limited costs (e.g., bandwidth)
- No dedicated servers

Nowadays, there are a large variety of content sharing applications, and they account for a large portion of traffic over the Internet.

Relying on the P2P paradigm to support these applications provide several advantages like robustness to failures, scalability, etc.



One of the way to disseminate content is to rely on gossip.

The theory of gossip tells us that if each node forwards randomly the updates to enough nodes, then every peer will receive the stream with high probability.

In this animation a blue update is released by the node S and it is propagated from each node to two other nodes. After, 2 hops every node has received the blue update.

Rational Nodes

Rational nodes aim at getting the content at the lowest possible overhead (they are not Byzantine nodes).

They aim at **maximizing their own benefit**, according to:

- Stream quality
- Communication overhead
- Computation overhead
- Risk

Their strategy can potentially harm the good dissemination of updates in the system.

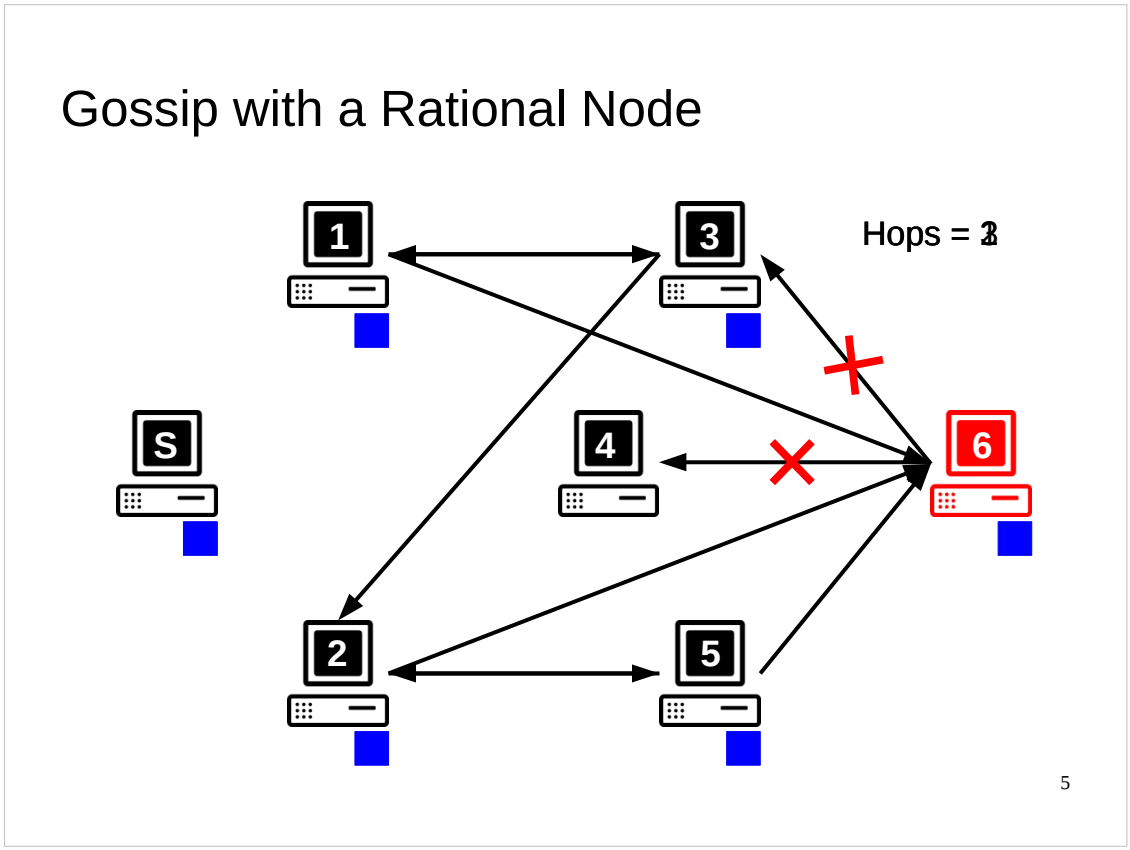
4

However, gossip protocols have to tolerate the presence of rational nodes, which aim at getting the content with the lowest possible overhead.

The benefit of rational nodes can be represented according to several axes because,
They want to receive as much as possible stream updates,
they want to send as little as possible messages
they want to perform as few as possible computations
they want to minimize the risk of being detected.

Rational nodes would deviate in any sort from the protocol, as long as they increase their benefit. However, they are not Byzantine nodes.

Their strategy can potentially harm the good dissemination of updates in the system. A gossip protocol has to be designed to protect correct nodes from the behavior of rational nodes.



In this animation, the node number 6 is now rational. Nodes 1 and 2 have just received the blue update. They propagate it correctly to nodes 3, 5 and 6.

After having received the blue update, node 6 will not send it as it should to nodes 3 and 5. Doing so, it saves a part of its bandwidth.

At the end of the propagation of the blue update, node 3 is not penalized as it received the update from node 1, but node 4 did not receive it, and therefore suffered from the rational behavior of node 6.

This is a simple example to illustrate individual rational behaviors, and why the gossip protocol has to be precisely designed to avoid them.

Problem: Colluding Rational Nodes

Several solutions have been designed to handle solitary rational nodes.

However, rational nodes may collude to:

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

Real-life collusions have been observed in Maze, a file sharing system, and until now, could not be prevented.

Several protocols have been designed to handle solitary rational nodes, among which BAR Gossip, LiFTing, and PeerReview.

However, rational nodes can cooperate to achieve their goals. Mainly to protect themselves from being detected, and to increase their benefit.

This type of collusion has been observed in Maze, a file sharing system.

In our paper, we precisely show why the rational collusion problem is not yet solved.

Outline

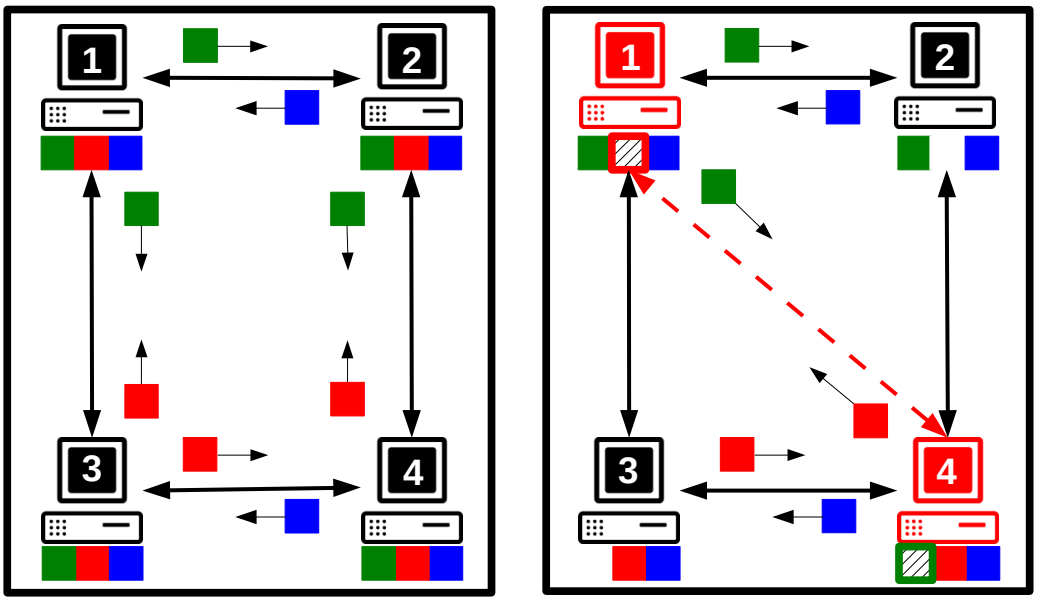
- Impact of collusions on Gossip
  - Example with symmetric exchanges
  - Results of experiments on existing protocols
- Key ideas of AcTinG
- Performance evaluation

In the rest of this presentation, I will show how collusions impact the dissemination of updates in Gossip protocols.

Then, I will give the key ideas of Acting, our protocol.

Finally, I will show some of the results we obtained to assess its good performance.

Colluders with Symmetric Exchanges



Without colluders

With colluders

We start with an animation where nodes exchange updates symmetrically (which is often referred as tit-for-tat), which means that a node receives the same quantity of updates it can send to its partner.

On the right, the red nodes 1 and 4 collude, they are part of a high quality network: low latency, high throughput. It is less costly for them to interact between themselves than with other nodes.

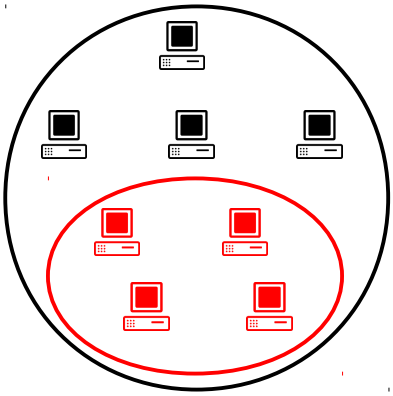
When nodes 1 and 4 exchange updates, outside the protocol, they receive their updates before other nodes. And they do not allow nodes 2 and 3 to exchange with them as they should.

At the end, on the left every nodes received the updates it needed, but on the right, nodes 2 and 3 missed one update each.

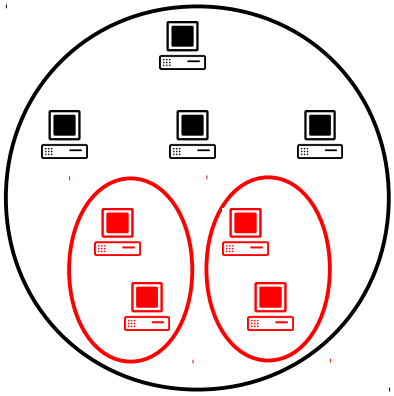
Impact of Colluders : Scenarios

A given proportion of the audience is made of one or several groups of colluding nodes.

We measure how well correct nodes receive the stream.



Experiment 1



Experiment 2

10

To measure the impact of colluders on the state of the art protocols, we considered two scenarios.

In the first set of experiments, there is only one group of colluders, and we change its size.

In the second set of experiments, we take the same quantity of colluders, and split them in several independent groups.

Measuring the Impact of Colluders

Colluders execute every possible deviation that increases their benefit, or protect them.

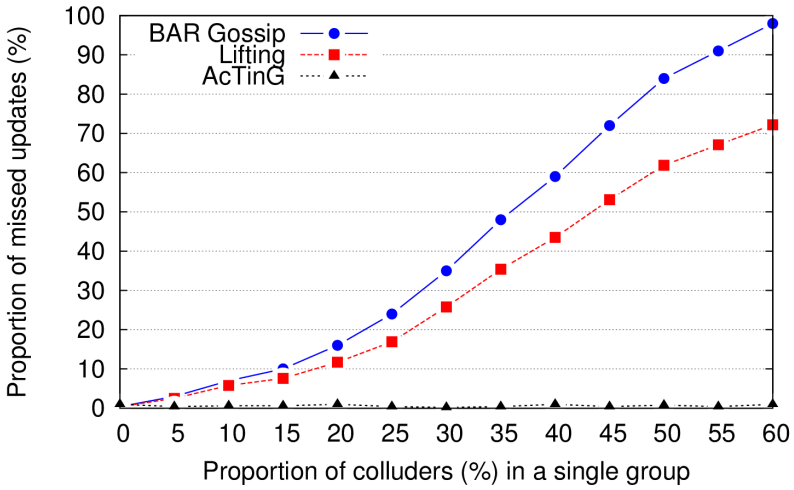
- **BAR Gossip**
  - No participation in the optimistic push protocol,
  - Exchange of updates in priority between colluders.
- **LiFTinG**
  - No participation in the verification protocols,
  - Blames are affected to correct nodes, forcing the administrator to exclude nobody, or to exclude a lot of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

In these experiments, colluders execute every possible deviations to increase their benefit, or protect themselves.

For exemple, with BAR Gossip, they do not participate in the optimistic push protocol, and exchange updates between each others.

With Lifting, they do not participate in the verification protocol, and wrongly blame other nodes (which is possible). In consequence, no nodes can be detected faulty with Lifting.

Exp. 1: Single group of colluders



15% of colluders → 10% of the updates are not received.

12

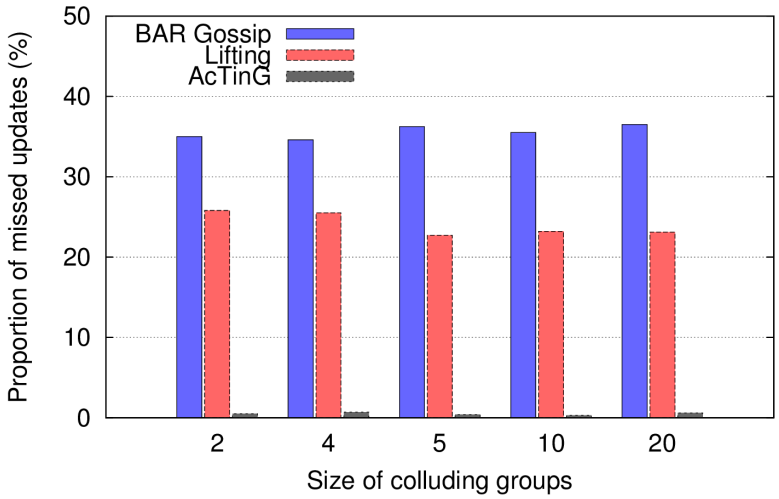
Here are the results of the first experiments.

The X axis gives the proportion of colluders in the system, and the Y axis gives the proportion of updates correct nodes do not receive correctly.

We show that colluding nodes can harm the performance of gossip-based systems when they form a single group of rational nodes.

For example, when 15% of the audience is made of colluders, correct nodes miss roughly 10% of the stream, which is already a lot.

Exp. 2: 30% of colluders in several groups



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

For our second sets of experiment, we show that the size of the colluding groups is not important.

Even when the nodes collude in groups of 2 nodes they can still consequently harm the correct nodes. We do not see it here, but they also receive the whole set of updates.

Impact of Colluders: Conclusion

Rational nodes can collude and increase their benefit, while not being detected.

They receive correctly the stream, but correct nodes do not.

Even small groups can harm the dissemination of updates.

We have showed that rational nodes can collude and increase their benefit, while not being detected.

During this deviation, they receive the whole stream, but correct nodes are harmed, and do not receive it completely.

In addition, even small groups of colluders can harm the dissemination of updates. It is then necessary to design a protocol such that correct nodes are protected from individual and colluding rational nodes.

Our goals

Current protocols can be heavily harmed by rational collusions.

We aim at designing the first gossip-based protocol such that:

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

Individual rational behaviours have been studied in various works. However, collective deviations are still possible, and have been observed. These collective deviations can noticeably degrade the performance of gossip-based protocols.

We aim at designing the first gossip-based protocol such that correct nodes correctly receive the updates, and are never expelled.

A rational node whose deviations impact a correct node's experience is eventually suspected by all correct nodes.

Outline

- Impact of collusions on Gossip
- Key ideas of AcTinG
- Performance evaluation

Let's start the second part of this presentation.

I will now show the key ideas of our solution, Acting.

Key Idea 1: Deterministic Behavior

It is possible to predict how a node interact with other nodes.

- Nodes are synchronized, and time is structured as a sequence of rounds.
- A PRNG guide associations between nodes.
- Sub-protocols are deterministic.

Limitation

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

First, we need to be able to predict how nodes interact with each other.

This is not new, and BAR Gossip was the first paper to use these conditions.

To do this nodes are synchronized, and time is structured as a sequence of rounds in which they interact. A pseudo-random generator is used to guide the associations between nodes. The last point is to ensure that the sub-protocols are deterministic.

However, this is not enough because nodes can choose not to participate in some parts of the protocol. They are not forced to do it.

Key Idea 2: Accountability

It should be possible to check the behaviour of nodes, and make them responsible from their declarations.

- Secure log, tamper evident and append only, to record the messages sent, or received.
- Session key pair consisting of a public and a private key, that is used to sign messages.

Limitation

However, nodes could hide the updates they received from accomplices. To do that, they would maintain different log versions.

The second idea is to be able to check the behaviour of nodes, and keep a trace of this verification.

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

To do it, we use a secure log that is tamper evident, append only to record the messages sent or received by a node.
This secure log needs a session key pair to sign messages.

However, as we show in our paper, nodes could still collude and hide the updates they receive from each other. They could maintain several logs, and

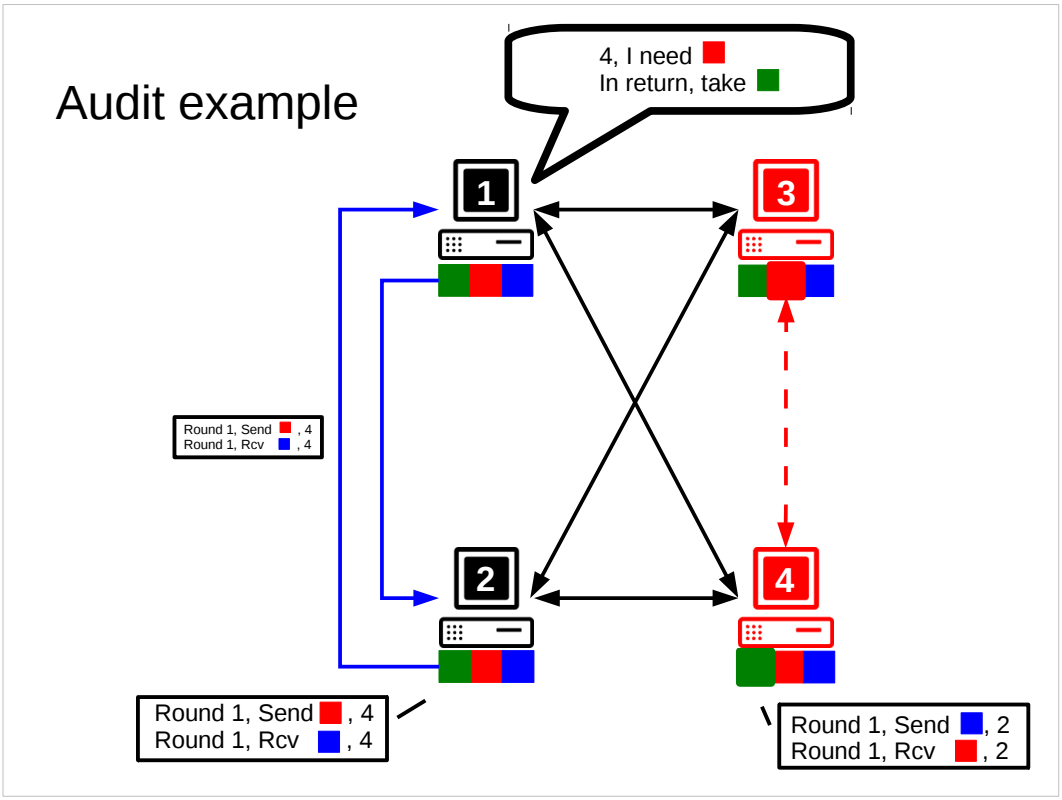
Key Idea 3: Audits

It should be possible to compare the declarations a node made to detect its lies.

- Audits which are **verifiable**, **random yet unpredictable**.
  - Verifiable: if a rational node decides not to audit other nodes, it should be eventually discovered by correct nodes.
  - Random: the overhead due to audits is decreased.
  - Unpredictable: if a rational node can predict whether or not it will be audited, then it would choose when to execute its deviations.

Deviating from the protocol means taking a high, and unpredictable, risk of being detected.

- The logs are consistent, by recomputing the hash values associated to log entries.
- The presence of the exchanges the nodes was supposed to initiate
- The correct reception, and emission, of updates.
- The audits were



We can now use asymmetric exchanges, and

AcTinG's properties

Finally, our protocol ensures that any node:

- Interacts with other nodes, and cannot avoid it;
- Receives the updates it did not receive officially;
- Sends the updates its partners did not receive officially;
- Audits its partners when it has to.

Nodes can still exchange updates off the record, but it is not interesting. They will be forced to receive them officially anyway.

Outline

- Impact of collusions on Gossip
- Key ideas of AcTinG
- Performance evaluation

In the rest of this presentation, we will show how collusions impact the dissemination of updates in Gossip protocols.

Then, I will give the key ideas of our solution, Acting.

Finally, I will show some of the results we obtained to assess its good performance.

Probabilistic audit

Audits are triggered randomly, with a probability of 5%.

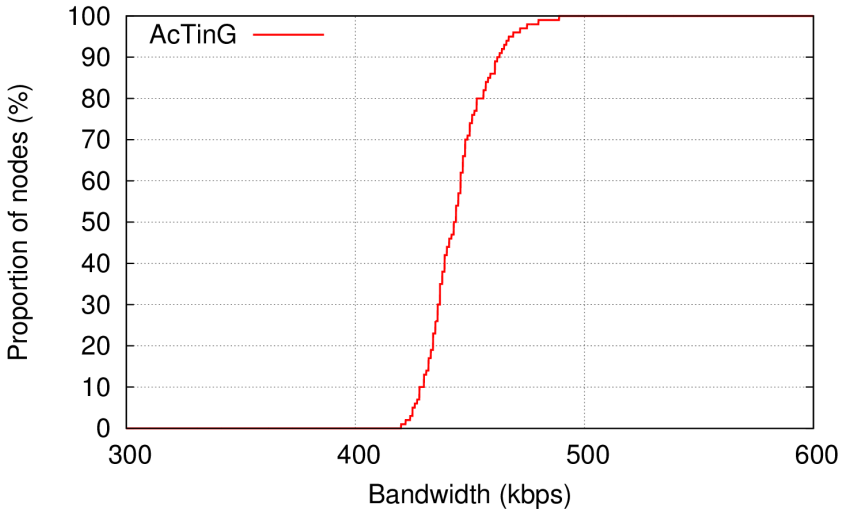
Updates expire N rounds after their release.

Audits concern the entries of the last N rounds, which means that a deviation can be observed by an auditing node during RTE rounds.

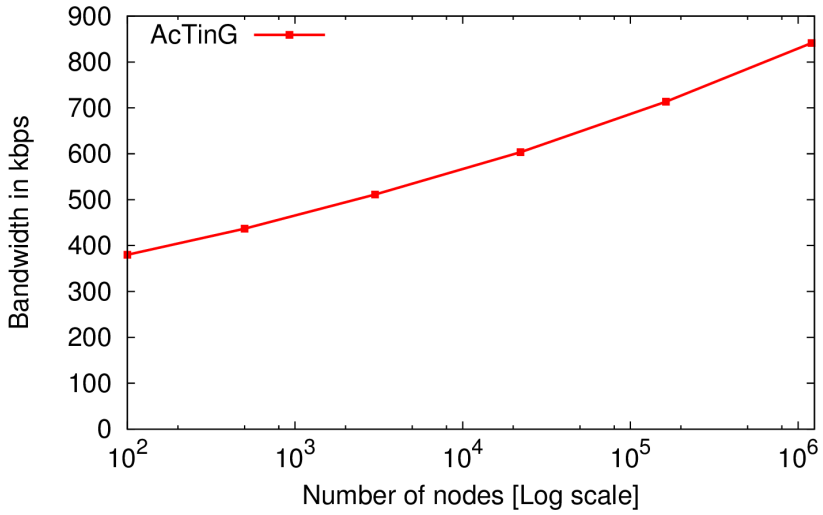
Our evaluations show that when 10% of the audience collude in a single group, the probability for a deviation to be detected is equal to 60%.

The only possible deviation bring a long term gain of 3% (if never detected).

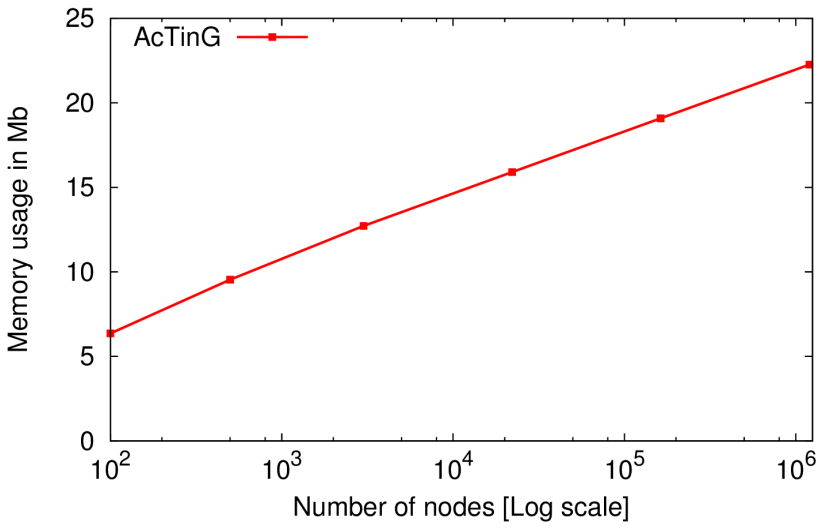
AcTinG's overhead - 300 Kbps stream



Scalability



Memory usage



Questions?