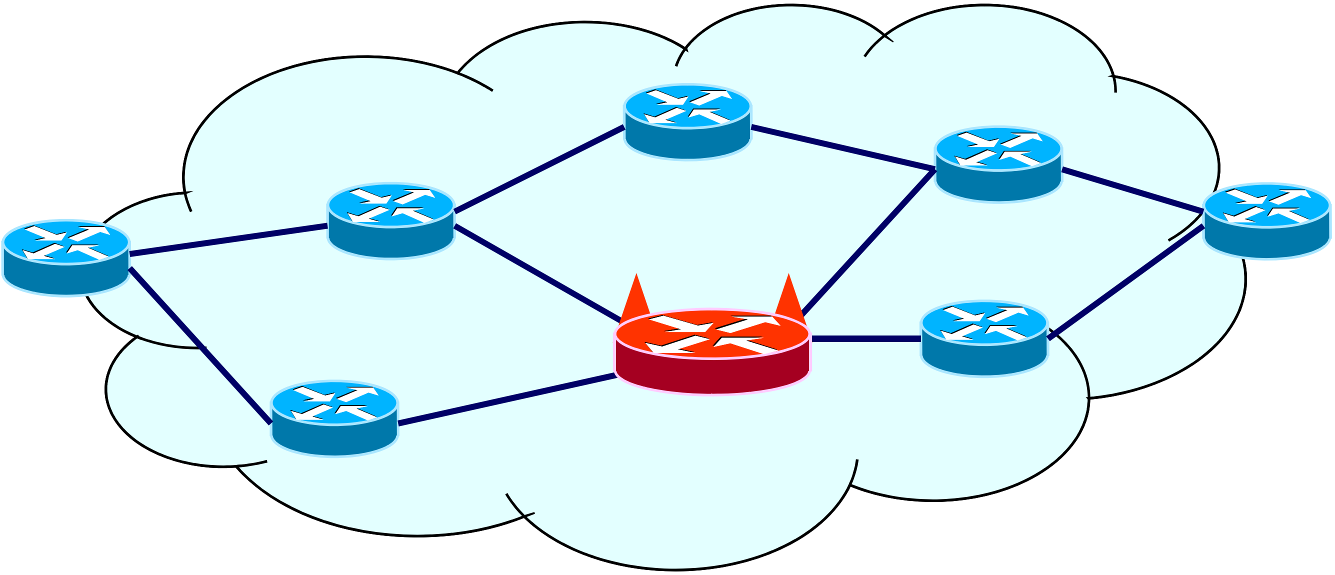
Failure Localization in the Internet



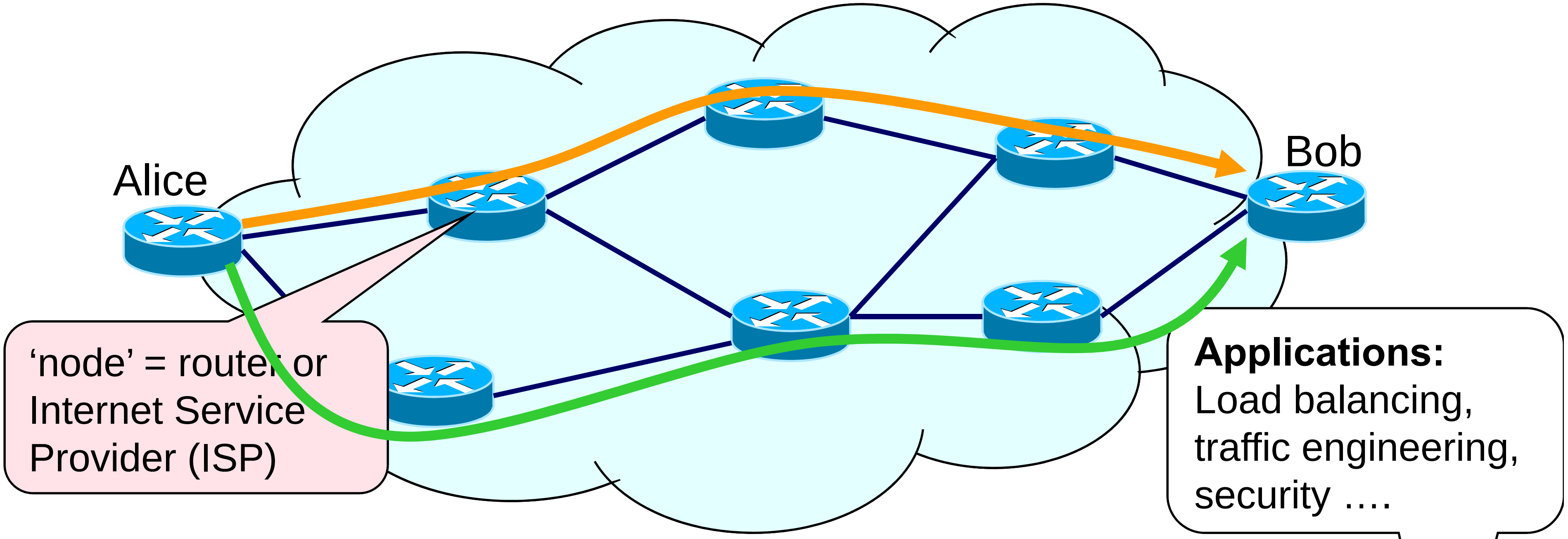
Boaz Barak, **Sharon Goldberg,** David Xiao

Princeton University

Excerpts of talks presented at Stanford, U Maryland, NYU.

Why use Internet path-quality monitoring?

Internet: Best-effort delivery, congestion, no integrity for traffic, *competition*



‘node’ = router or Internet Service Provider (ISP)

Applications: Load balancing, traffic engineering, security ….

Many (new) applications need Internet path quality monitoring….

Intelligent Routing: To inform routing decisions

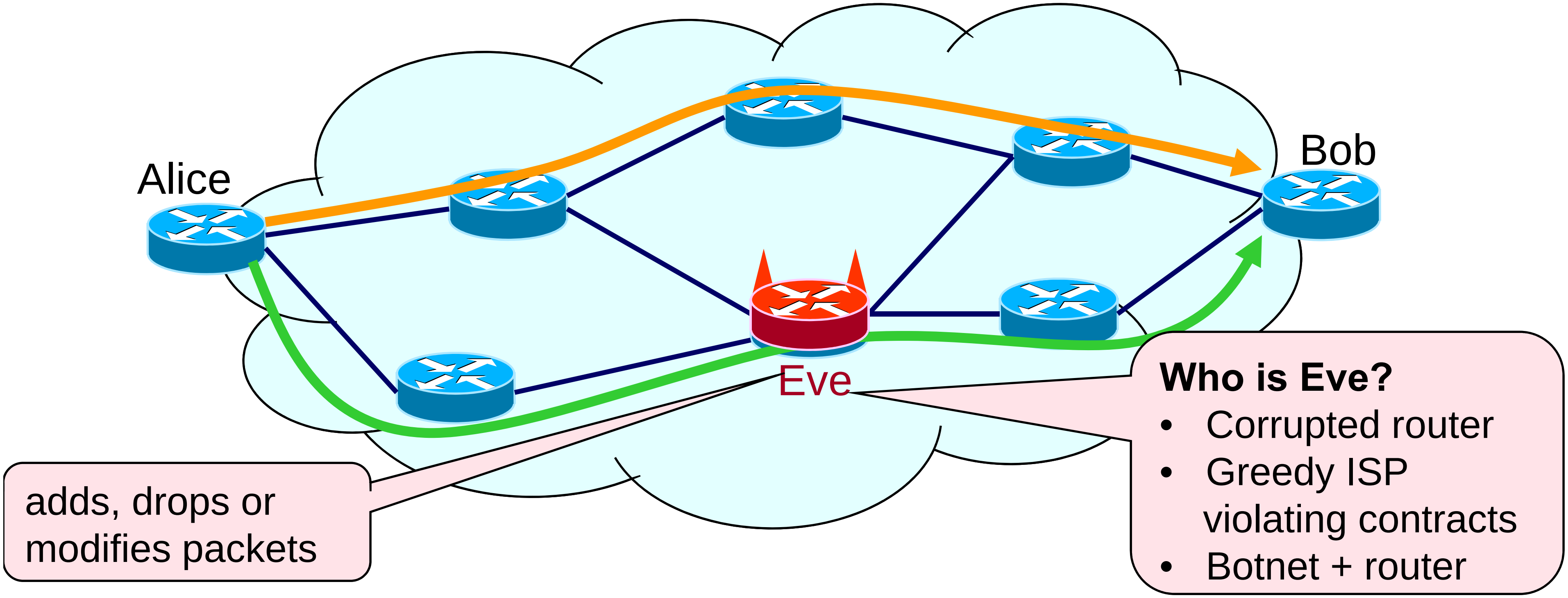
- Source routing: (Alice chooses nodes on path to Bob)
- Multipath routing: (Alice switches paths based on performance)

Network Accountability: To demand reimbursement from faulty ISPs

- Necessary to drive innovation! (game-theoretic study of [LC06])

The presence of adversaries

Internet: Best-effort delivery, congestion, no integrity for traffic, *competition*



adds, drops or modifies packets

Who is Eve?

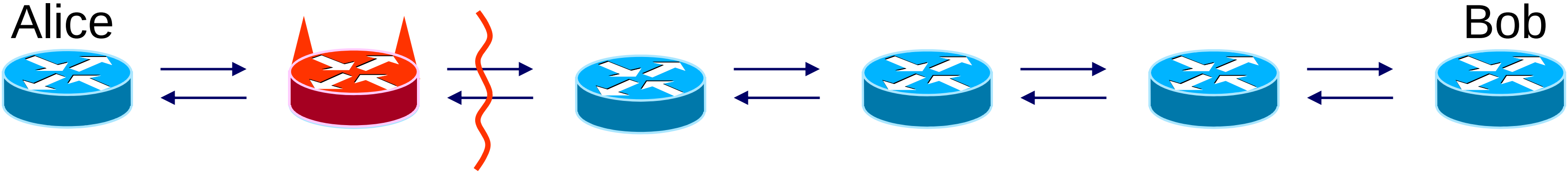
- Corrupted router
- Greedy ISP violating contracts
- Botnet + router

Failure Detection: Alice wants to know **if** her packets were dropped/modified.

Failure Localization: Alice wants to know **who** dropped/modified her packets.

We consider **benign** (congestion, link failure) and **malicious** (due to Eve) packet loss, but do not require Alice to distinguish between them.

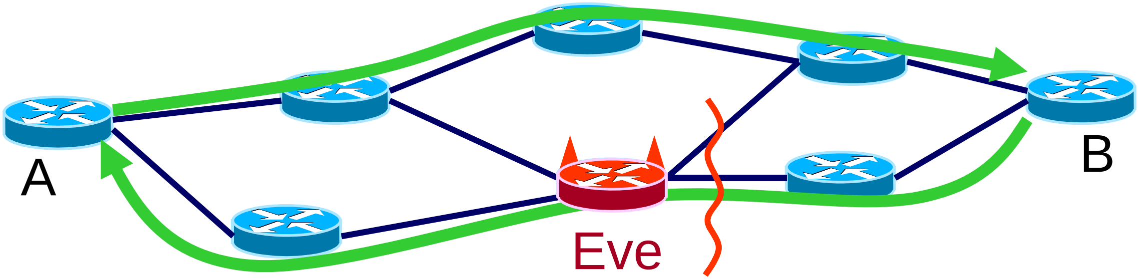
Failure Localization (FL)



We assume:

1. Alice knows identity of nodes on path.
2. Eve occupies node(s) on the path, and can add, drop, modify packets.
3. Alice doesn't know where Eve is.
4. Symmetric paths

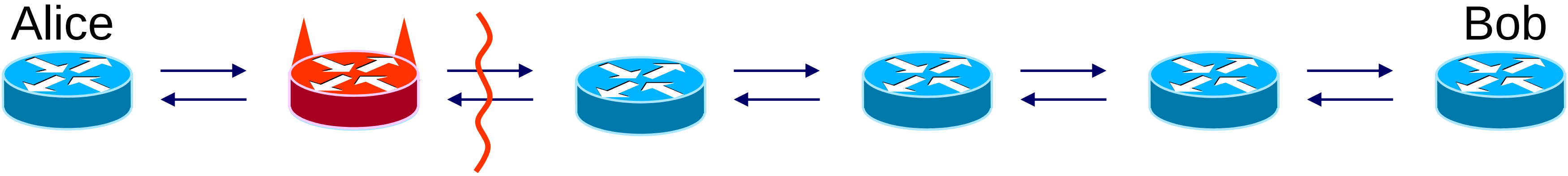
Need more assumptions about Eve for assymetric path setting (Eve occupies only 1 path ? – left for future work)



Need to model a path switching mechanism ?

Maybe we should consider the whole graph, not just a path?

Two flavours of Failure Localization (FL)



We assume:

1. Alice knows identity of nodes on path.
2. Eve occupies node(s) on the path, and can add, drop, modify packets.
3. Alice doesn't know where Eve is.
4. Symmetric paths

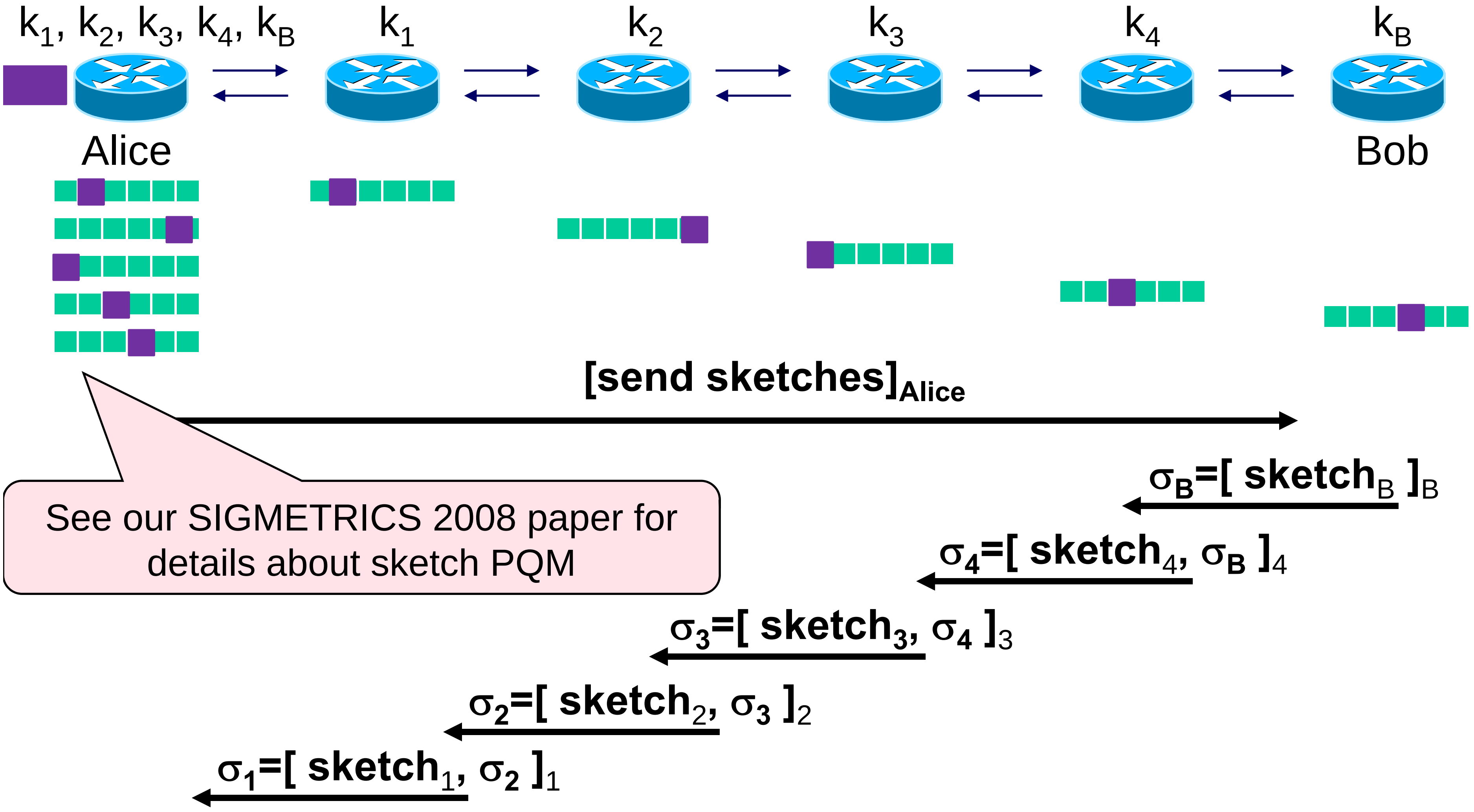
Secure per-packet failure localization (FL):
For each packet dropped or modifies on a link , the Alice **outputs that link.**

Secure statistical fault localization (FL):
If the packet loss rate on a link exceeds **β**, Alice **outputs that link** (or a link adjacent to Eve) *regardless of Eve's behavior*
Alice **will not alarm** if packet loss rate on the path is less than **α**

Contributions of our work

1. Per-packet failure localization protocols
2. Statistical failure localization protocols
3. Lower bounds:
   - FL needs keys and crypto at **each node on path**
4. Implications of our work
   - FL protocols necessarily require the participation of **every node on the path**
     - And, thus, is expensive to deploy
     - Can deploying FL be compatible with node incentives?
   - FL is good for highly secure networks / important traffic

Statistical FL by composition of Sketch PQM



$[\text{send sketches}]_{\text{Alice}}$

$\sigma_B = [\text{ sketch}_B\ ]_B$

$\sigma_4 = [\text{ sketch}_4, \sigma_B\ ]_4$

$\sigma_3 = [\text{ sketch}_3, \sigma_4\ ]_3$

$\sigma_2 = [\text{ sketch}_2, \sigma_3\ ]_2$

$\sigma_1 = [\text{ sketch}_1, \sigma_2\ ]_1$

See our SIGMETRICS 2008 paper for details about sketch PQM

‘Onionizing’ the reports prevents Eve selectively dropping reports for an innocent node.

Contributions of our work

1. Per-packet failure localization protocols
2. Statistical failure localization protocols
3. Lower bounds:
    - FL needs keys and crypto at **each node on path**
4. Implications of our work
    - FL protocols necessarily require the participation of **every node on the path**
        - And, thus, is expensive to deploy
        - Can deploying FL be compatible with node incentives?
    - FL is good for highly secure networks / important traffic

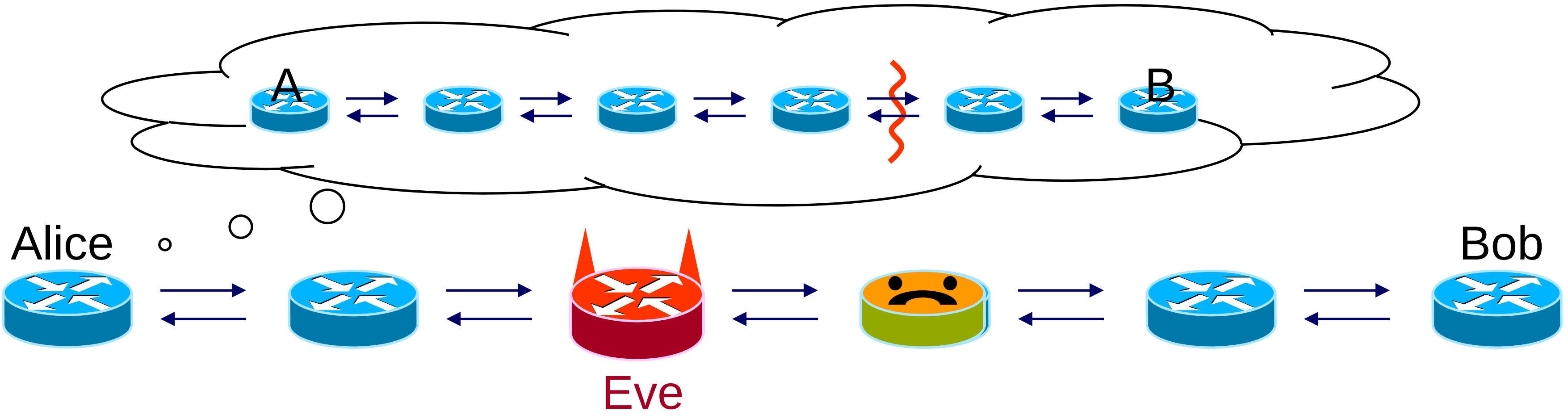
Lower Bounds for per-packet Fault Localization

Proof idea: If a node **i** lacks a resource

Eve at node **i-1** can trick Alice into thinking node **i+1** failed

Alice implicates link **(i,i+1)**

Eve breaks security



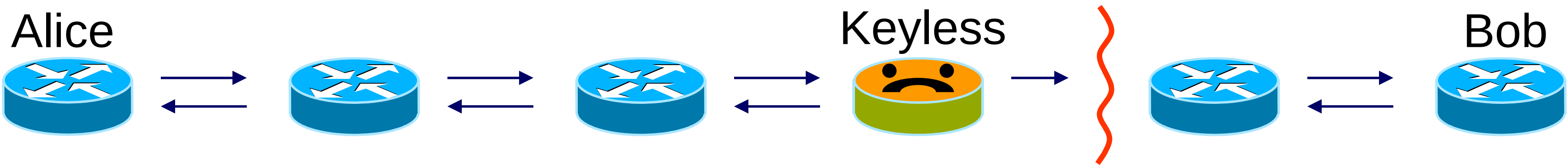
Each node must: 1) Share keys with Alice

2) Use cryptographic operations

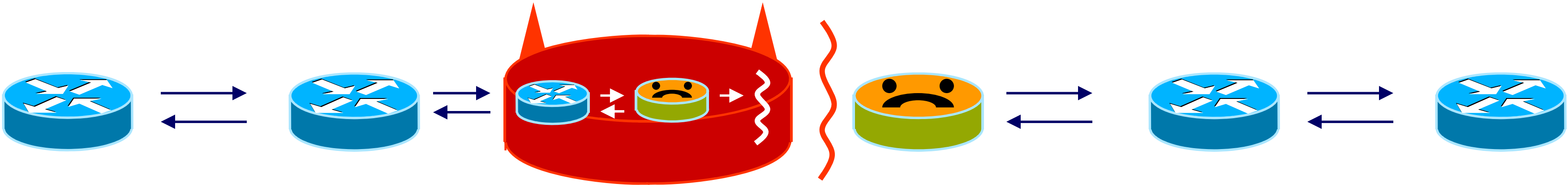
Fault Localization needs keys at each node

Theorem: Each node needs a shared secret with Alice

Proof: Suppose node *i* does not a share secret with any upstream node:



Case (a): Node *i+1* is unreachable



Case (b): Eve drops packets and impersonates keyless node *i*

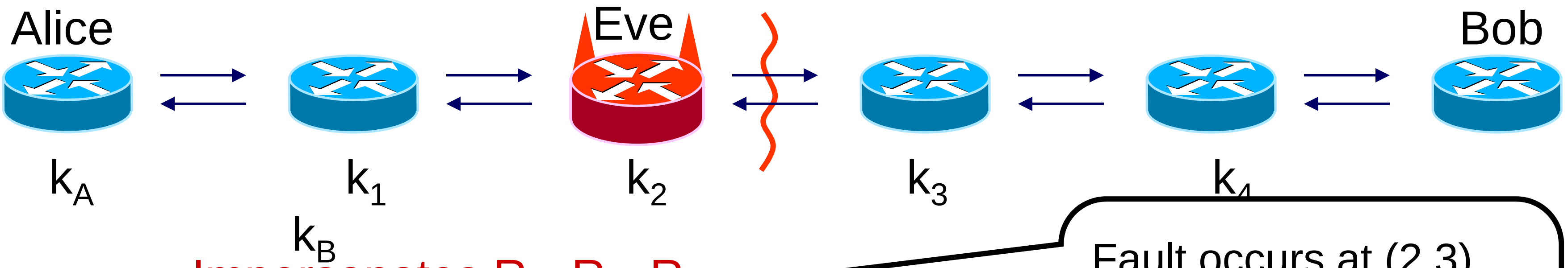
Case (a) and case (b) are indistinguishable to Alice

⇒ In case (b) Eve drops packets while making innocent link (i, i+1) look guilty.

⇒ The FL protocol cannot be secure.

Can we use reductions to prove FL needs crypto?

Proof idea [IL89]**:** Existence of a secure FL protocol ⇒ existence of a OWF



Impersonates R_3, R_4, R_B

Fault occurs at (2,3) but if Eve successfully impersonates, Alice sees no fault!

Define: OWF **$f(k_A, k_1, \ldots, k_N, k_B, Á) = FL\_Conversation(R_2,R_3)$**

The reduction: ∃ Ivan that inverts the OWF ⇒ ∃ Eve that breaks FL security

Very Nice!

But we only proved that **someone** needs to do crypto.

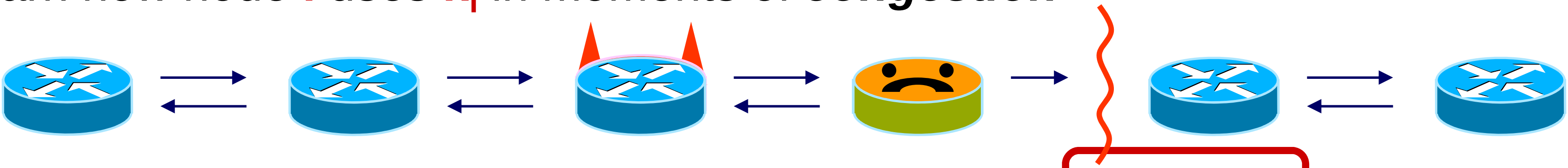
We want to show that **each node** needs to do crypto.

Fault Localization needs crypto at each node

Theorem: Each node needs to perform cryptographic operations.

Proof (sketch, for the per-packet case): Suppose node **i** has a shares key **k_i** with Alice but does not do crypto.

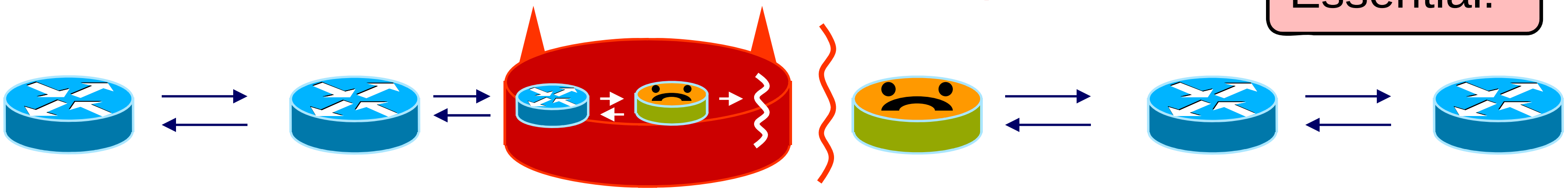
Since **i** doesn't do crypto, Eve can observe messages she gets from **i** and learn how node **i** uses **k_i** in moments of ***congestion***



Case (a): Node *i+1* is unreachable due to congestion

Essential!

Now Eve can impersonate node **i** by using the **k_i** that she lear

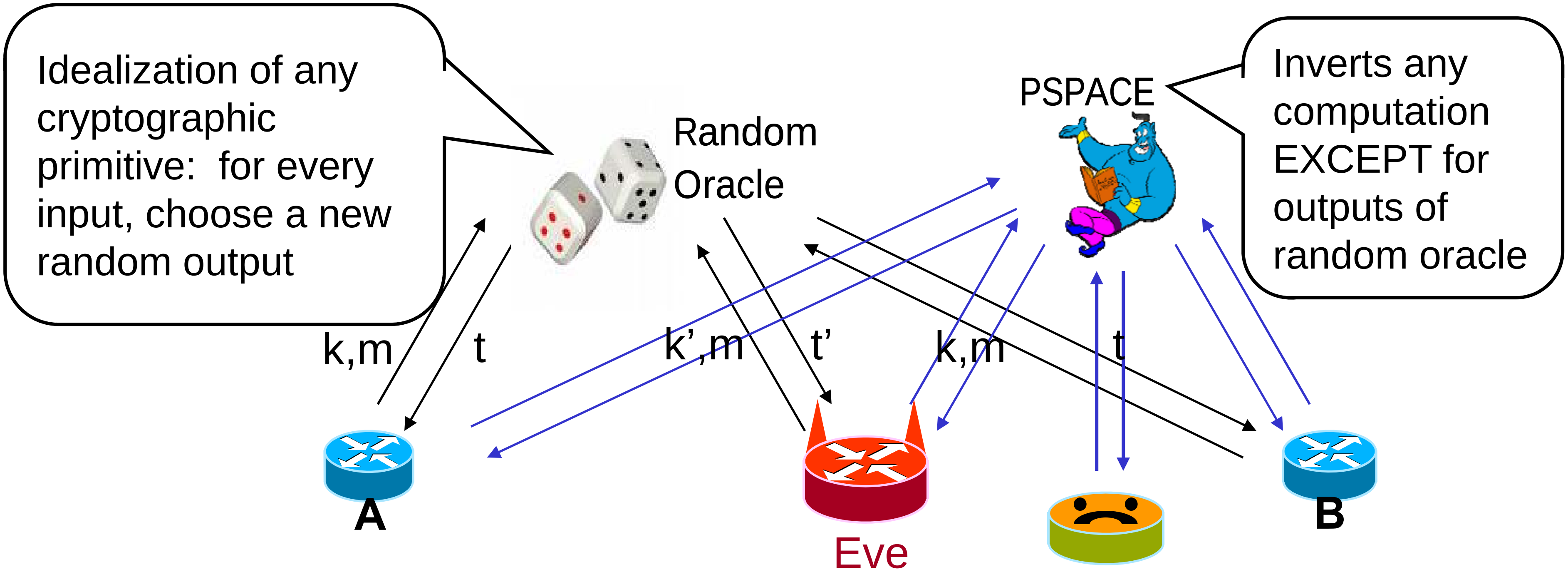


Case (b): Eve drops packets and impersonates crypto-less node *i*

⇒ Eve can drop packets while making innocent link *(i,i+1)* look guilty, and the FL protocol is not secure!

FL needs crypto. Proof tool: oracle separation

Black-box constructions: Use only input / output properties of the primitive



Idealization of any cryptographic primitive: for every input, choose a new random output

Inverts any computation EXCEPT for outputs of random oracle

Security of the protocol is based on the security of the cryptographic primitive

[IR, BGS]: Any secure black-box protocol must remain secure even when the PRF is replaced by a Random Oracle and all parties have a PSPACE oracle

In our proof, the node that "doesn't do crypto" can't call the random oracle!

Oracles

Random Oracle

x

RO(x) = Uniformly random string in $\{0,1\}^r$

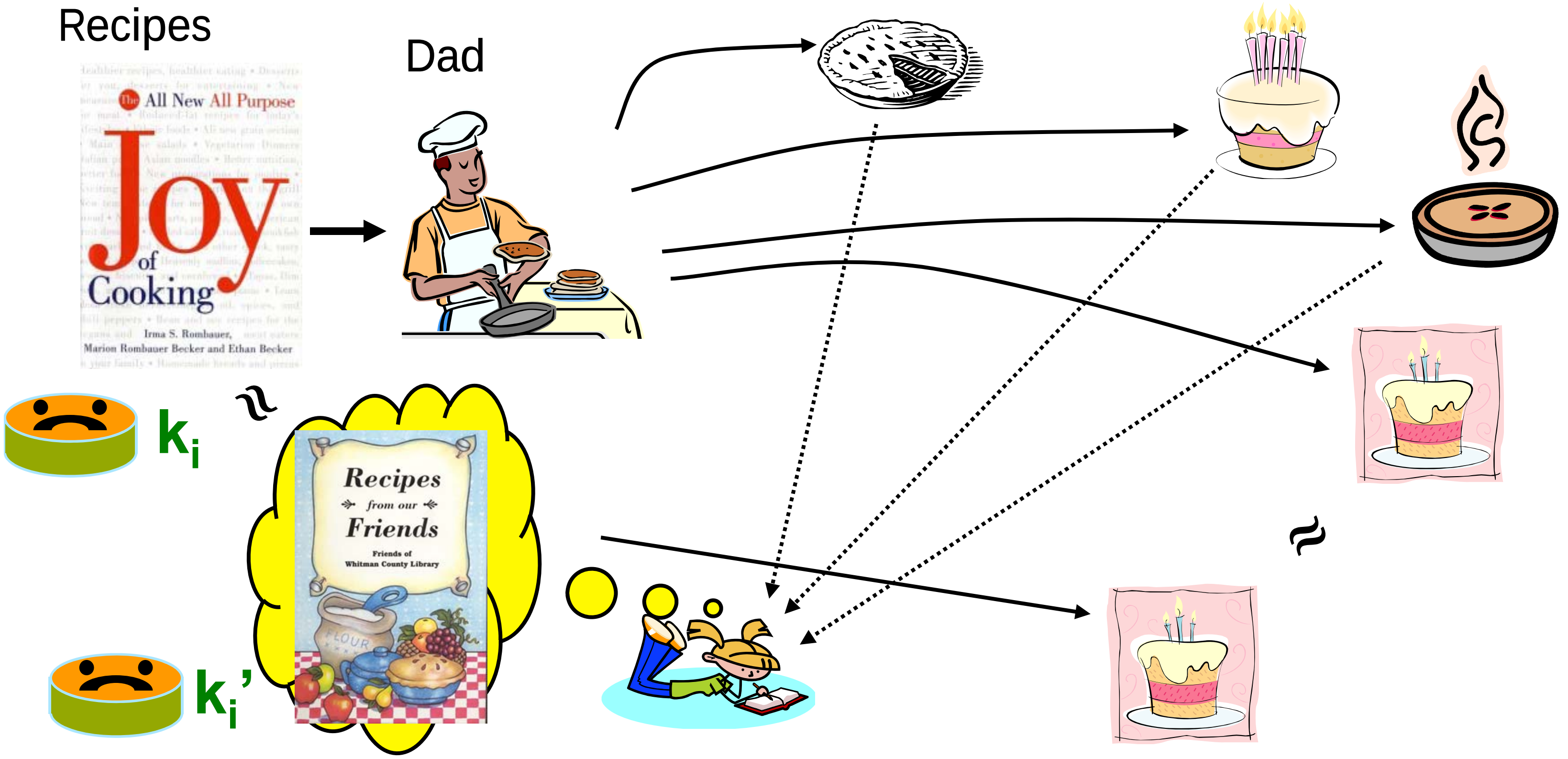
PSPACE Oracle

X in L?

(L in PSPACE)

Yes/No

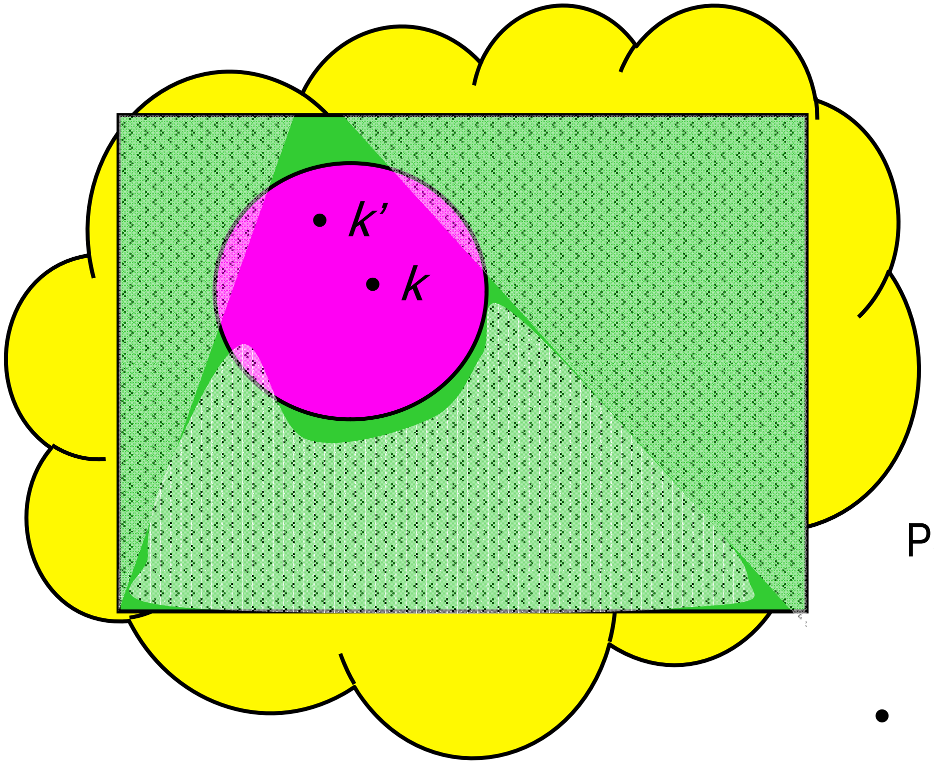
FL needs crypto. Proof tool: learning algorithm

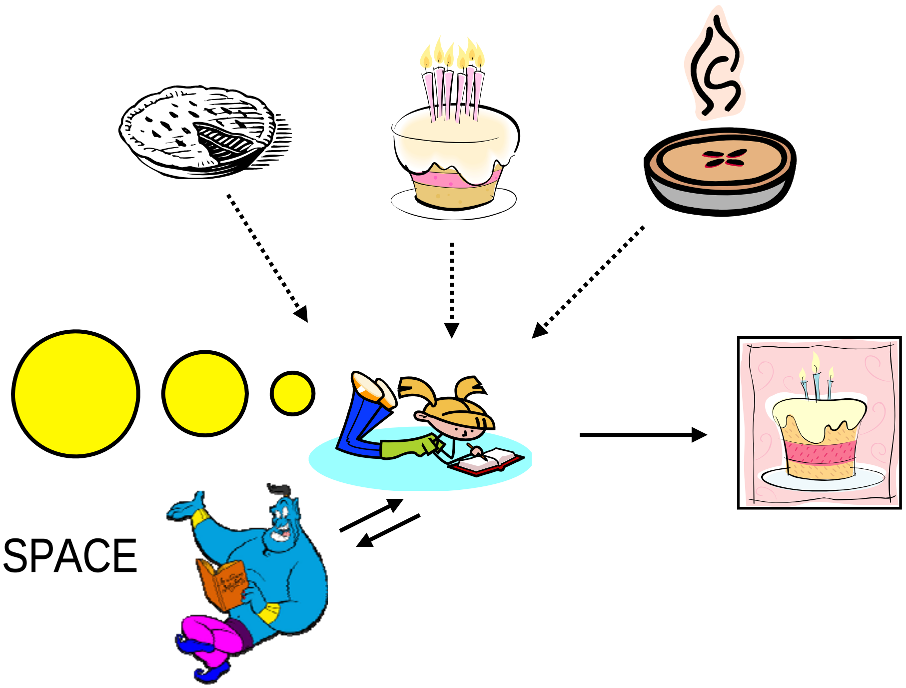


[Naor-Rothblum]: **Inefficient** algorithm that learns on **$O(n/(\varepsilon^2\ \delta^2))$** samples and outputs sample that is statistically ε-close to true distribution, w.p. **$> 1-\delta$**.

Inefficient step can be done by PSPACE oracle

Sketch of Learning Algorithm





🟩 All possible keys

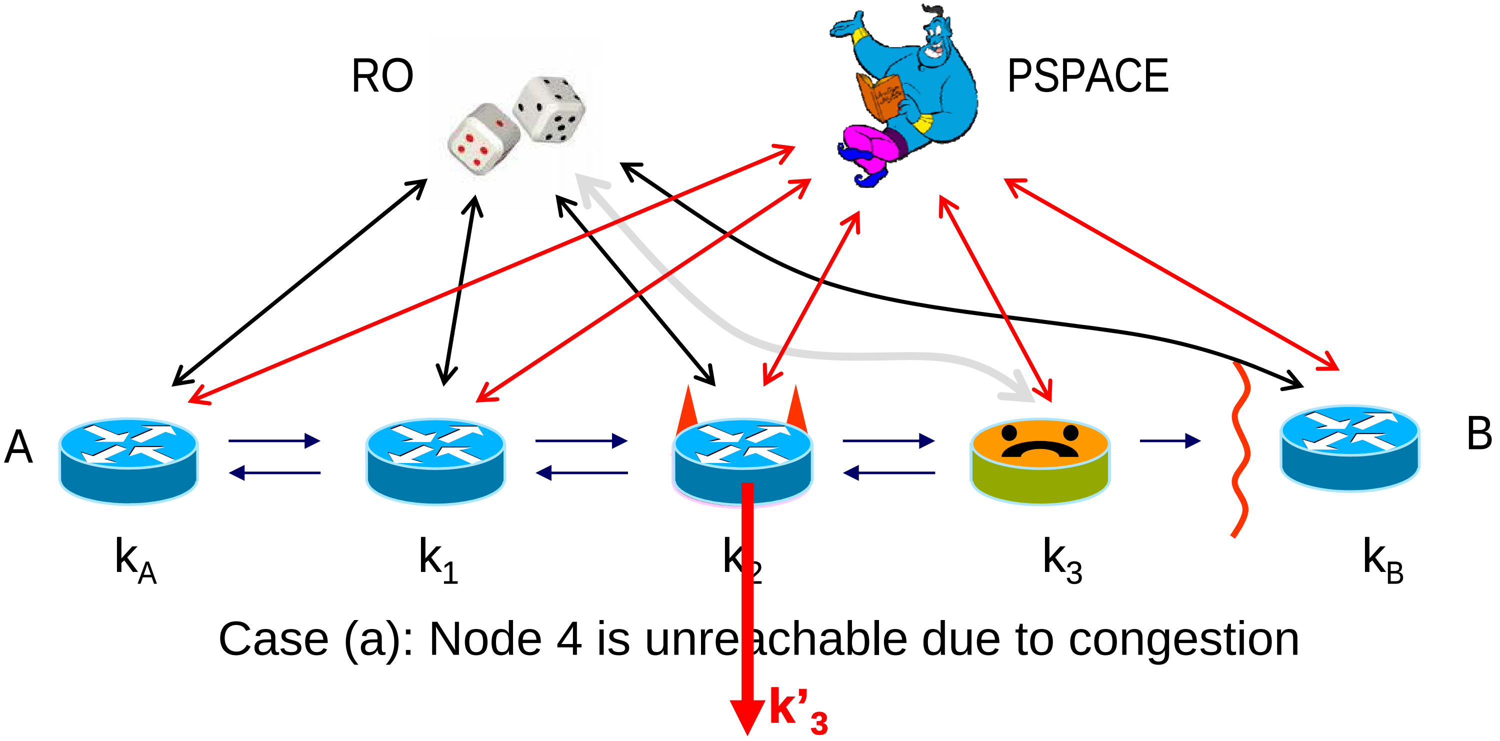
🟪 Behaves ε-close to *k*

▨ Inconsistent with samples

- [NR] show that after each sample either:
  - Can get good *k*' w.p. $1 - \delta$
  - Or entropy of *k* decreases by $\Omega(\varepsilon^2 \delta)$
- Algorithm succeeds after at most $O(n / (\varepsilon^2 \delta^2))$ samples w.p. $1 - \delta$

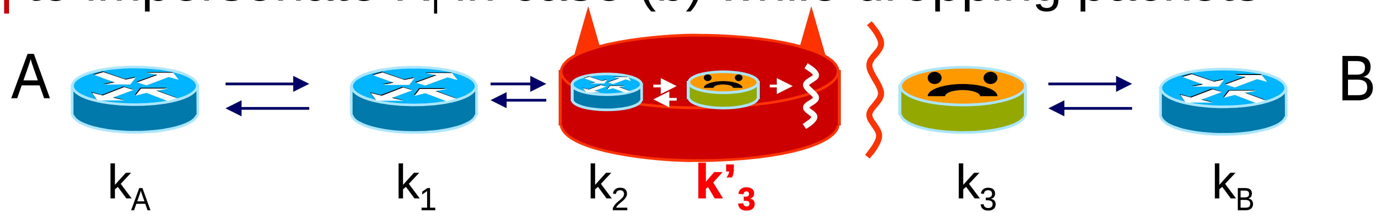
Black-box FL needs **crypto** at each node (1)

Eve can learn **k_i** in case (a) using [NR] algorithm with PSPACE oracle



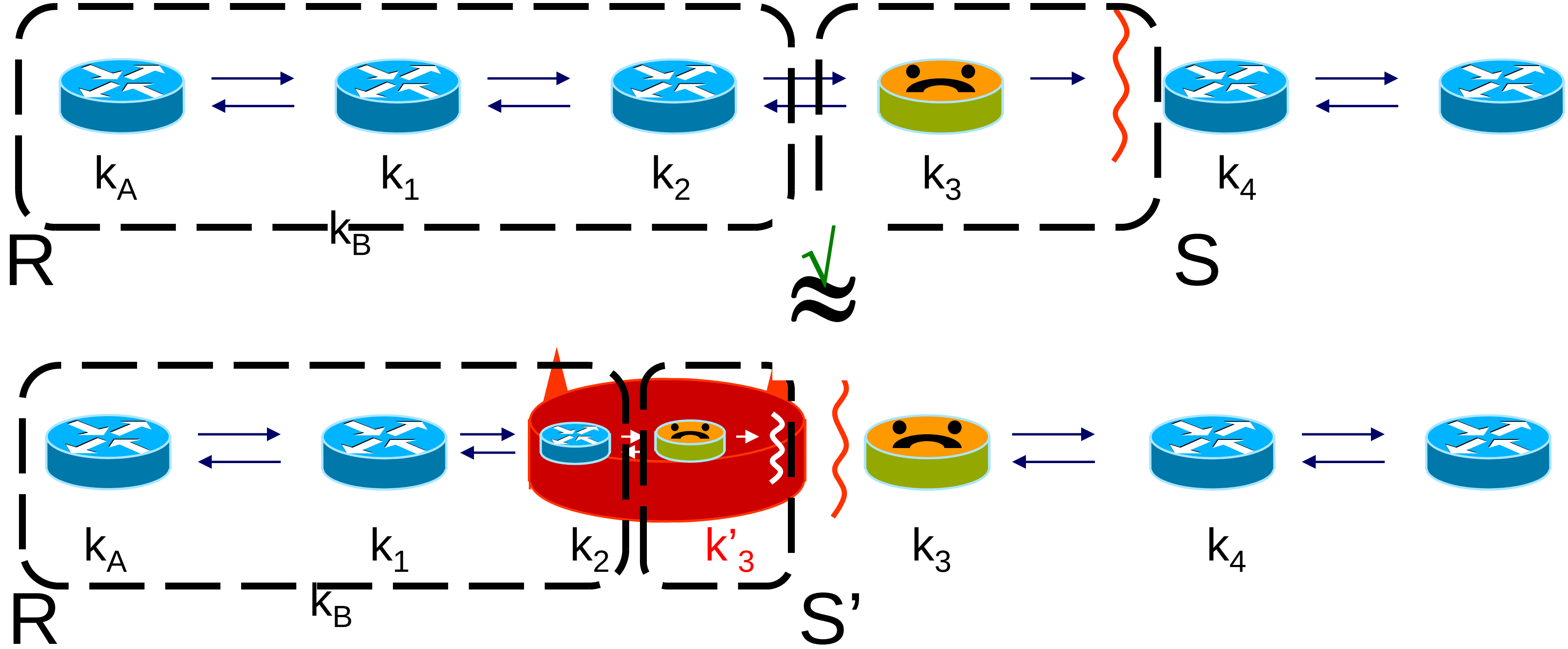
Case (a): Node 4 is unreachable due to congestion

Eve uses **k'_i** to impersonate R_i in case (b) while dropping packets



Does Eve fool Alice?

Black-box FL needs crypto at each node (2)



Lemma: If **$\Delta[(R, S), (R', S')] < \varepsilon$** and each pair **(R, S), (R', S') independent**

then **$\Delta[(R, S), (R, S')] < r\,\varepsilon$**

where **r** is the number of rounds of protocol.

PSPACE

+ [NR]

Eve wins because node *i* does not call Random Oracle and is therefore independent of other nodes

Contributions of our work

1. Per-packet failure localization protocols
2. Statistical failure localization protocols
3. Lower bounds:
    - FL needs keys and crypto at **each node on path**
4. Implications of our work
    - FL protocols necessarily require the participation of **every node on the path**
        - And, thus, is expensive to deploy
        - Can deploying FL be compatible with node incentives?
    - FL is good for highly secure networks / important traffic

Note (April 2008)

1. This talk contains an older version of our lower bounds – please see the full version of our paper, [Barak, Goldberg, Xiao., "Protocols and Lower Bounds for Fault Localization in the Internet", EUROCRYPT 2008] for the full details
2. See also the companion paper to this work [Goldberg, Xiao, Tromer, Barak, Rexford, "Path-Quality Monitoring in the Presence of Adversaries", to appear at SIGMETRICS 2008.]