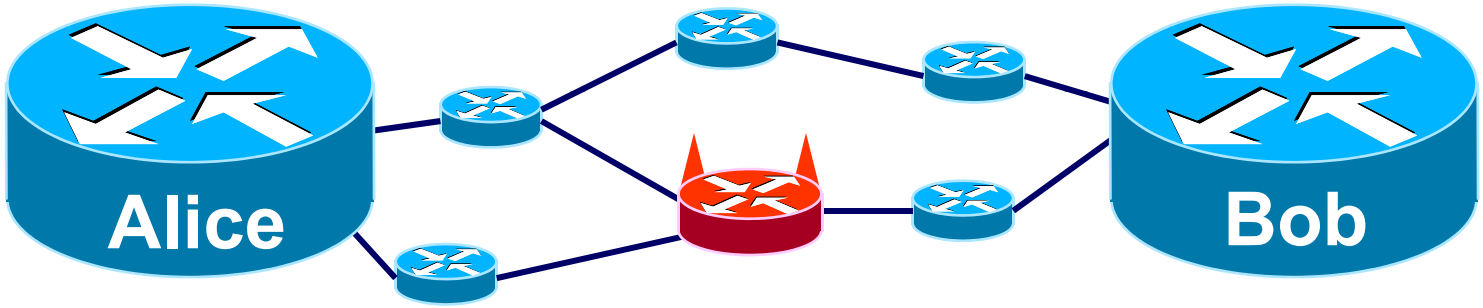
MS ALT-TAB
Nov 10, 2009

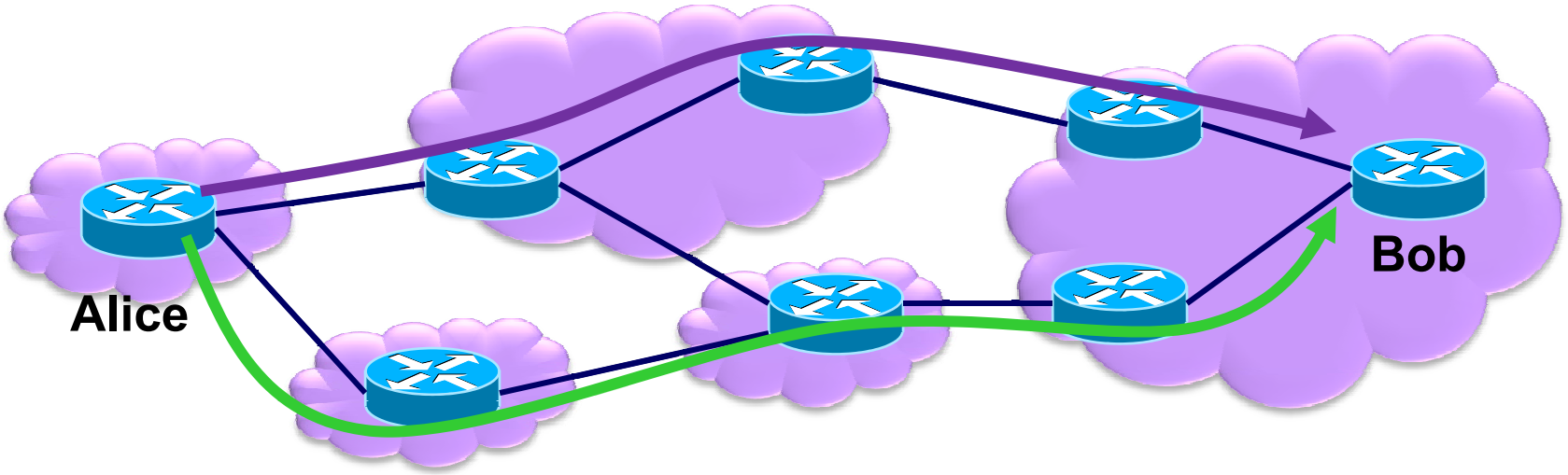
Path Quality Monitoring in the Presence of Adversaries



Sharon Goldberg
MSR New England

Based on work with:
Boaz Barak, Jennifer Rexford, Eran Tromer, David Xiao
Ramana Kompella, George Varghese

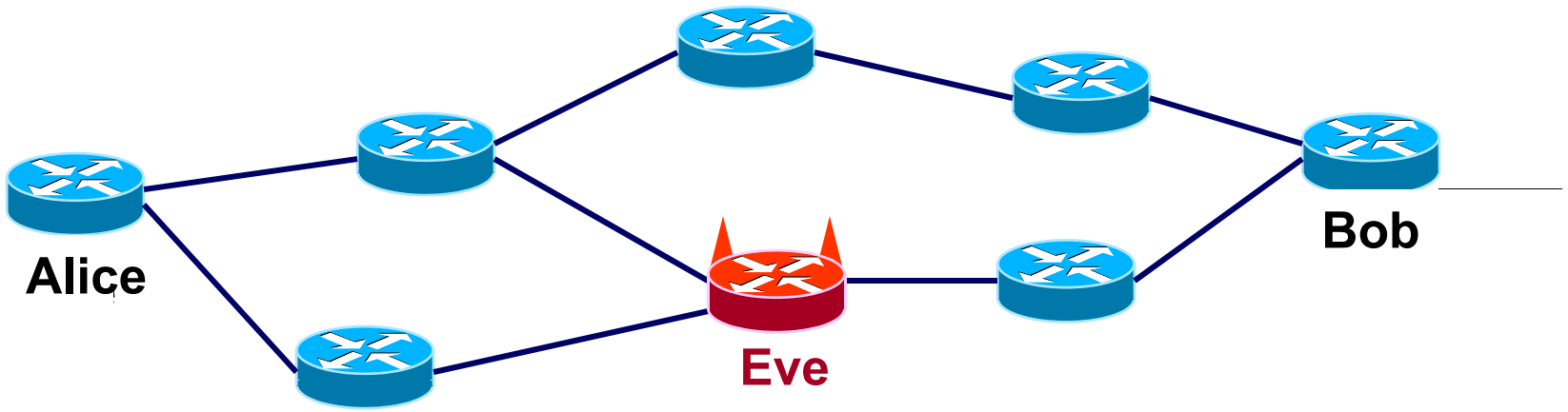
Path Quality Monitoring …



Routers need tools to measure packet loss and delay.

- Performance routing protocols
- Enforcing contracts - Service Level Agreements (SLAs)

… in the Presence of Adversaries



Knows monitoring protocol

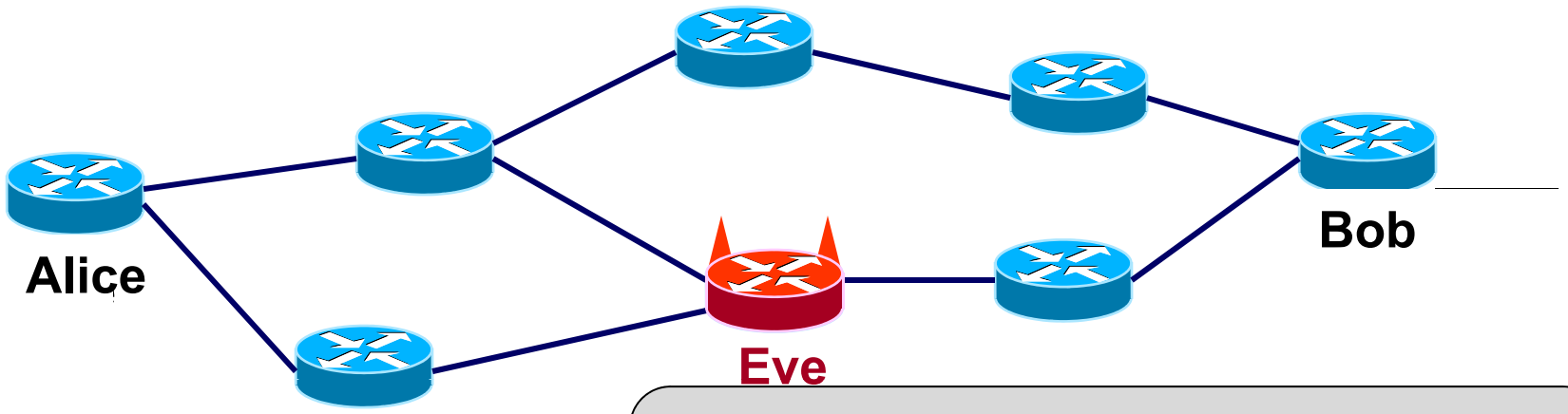
Add / **drop** / **modify** / **reorder** packets

Wants to bias Alice's measurements

Today's approaches cannot withstand active attack

(**ping**, traceroute, active probing, marked diagnostic packets)

… in the Presence of Adversaries



Knows monitoring protocol

Add / **drop** / **modify** / **reorder** packets

Wants to bias Alice's measurements

Today's approaches cannot withstand active attack

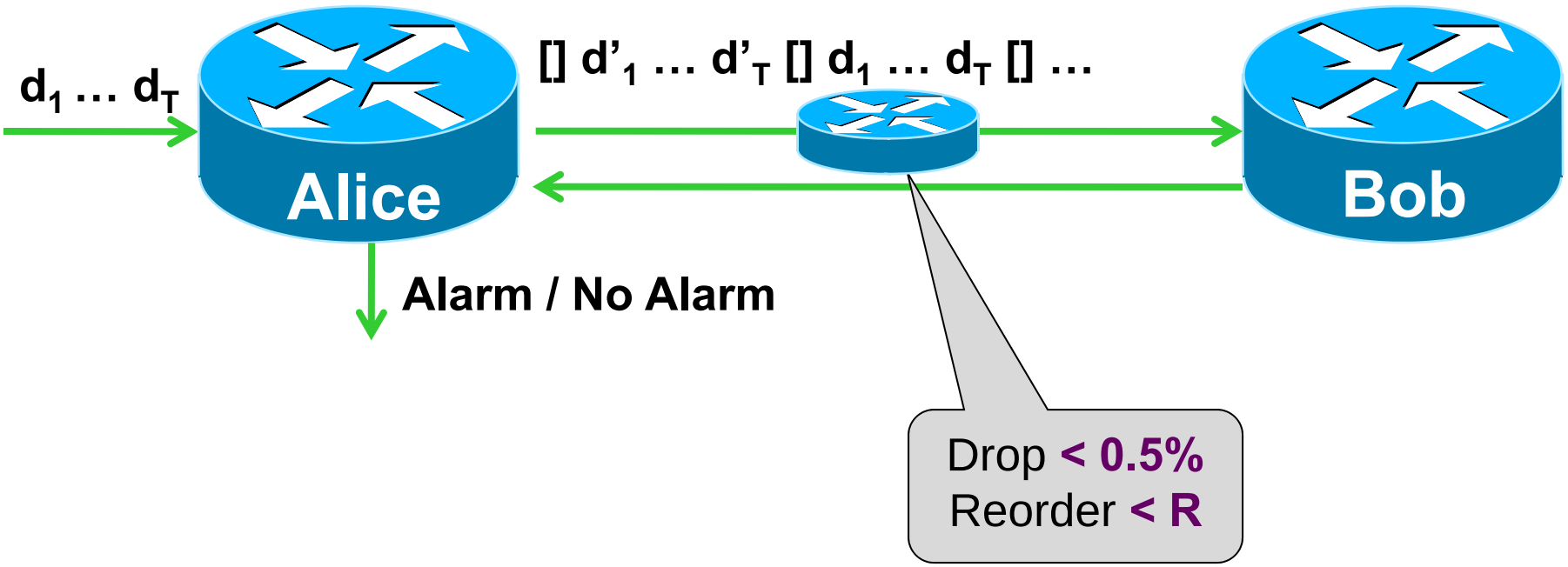
(**ping**, traceroute, active probing, marked diagnostic packets)

We want efficient protocols for high-speed routers

- Extremely limited storage, computation, communication
- No marking or encryption of existing traffic

Path Quality Monitoring : Setting (1)

Benign case

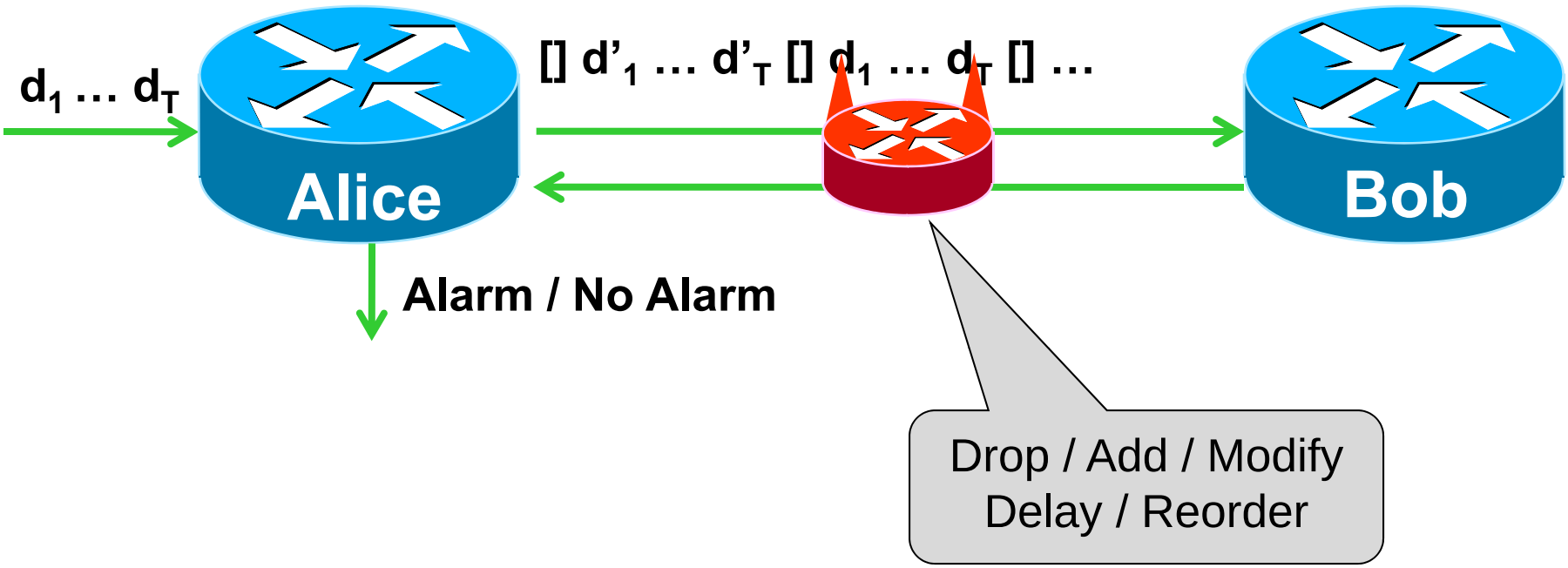


In each interval, with prob **1- δ = 99%**:

- **No Alarm** if **< 0.5%** packets are dropped in the *benign case*

Path Quality Monitoring : Setting (2)

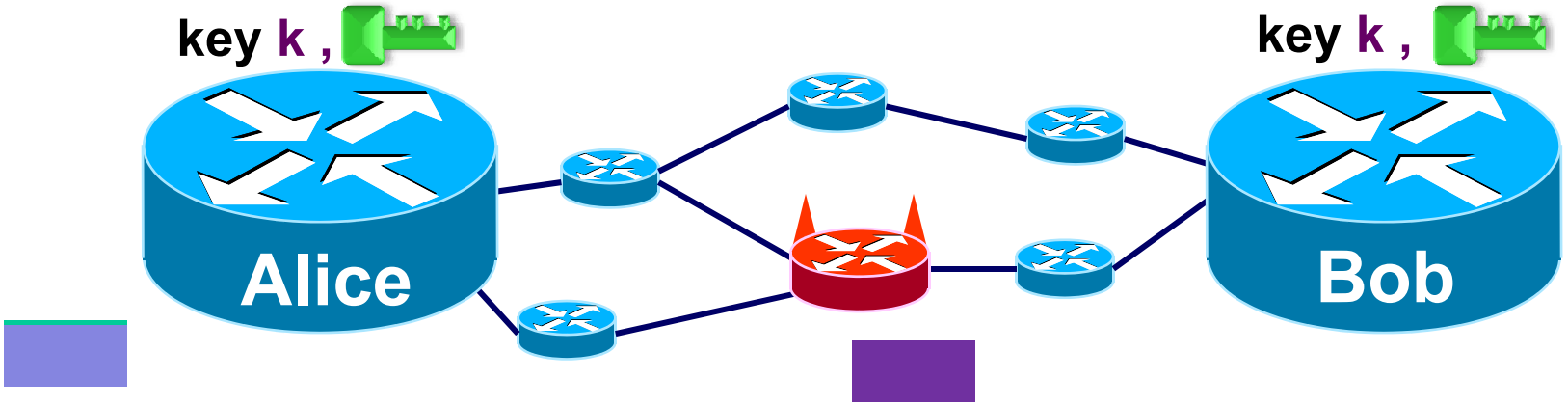
Malicious case



In each interval, with prob **1- δ = 99%**:

- **No Alarm** if **< 0.5%** packets are dropped in the *benign case*
- **Alarm** if **> 1%** packets do not arrive correctly at Bob, *regardless of Eve's actions*

Packet Loss & Modification : Protocol



A 0 0 1 -2 1 0 1 0 3 1

PRF each packet $f_k(d)$ = **index**
Update sketch **A[index] += 1**

Send authenticated (MAC'd) sketch

B 0 0 1 -1 1 0 -1 0 4 0

PRF each packet $f_k(d)$ = **index**
Update sketch **B[index] += 1**

Take difference sketch **X = A-B**
MAC and send

Decide btwn **> 1%** and **< 0.5%** loss:

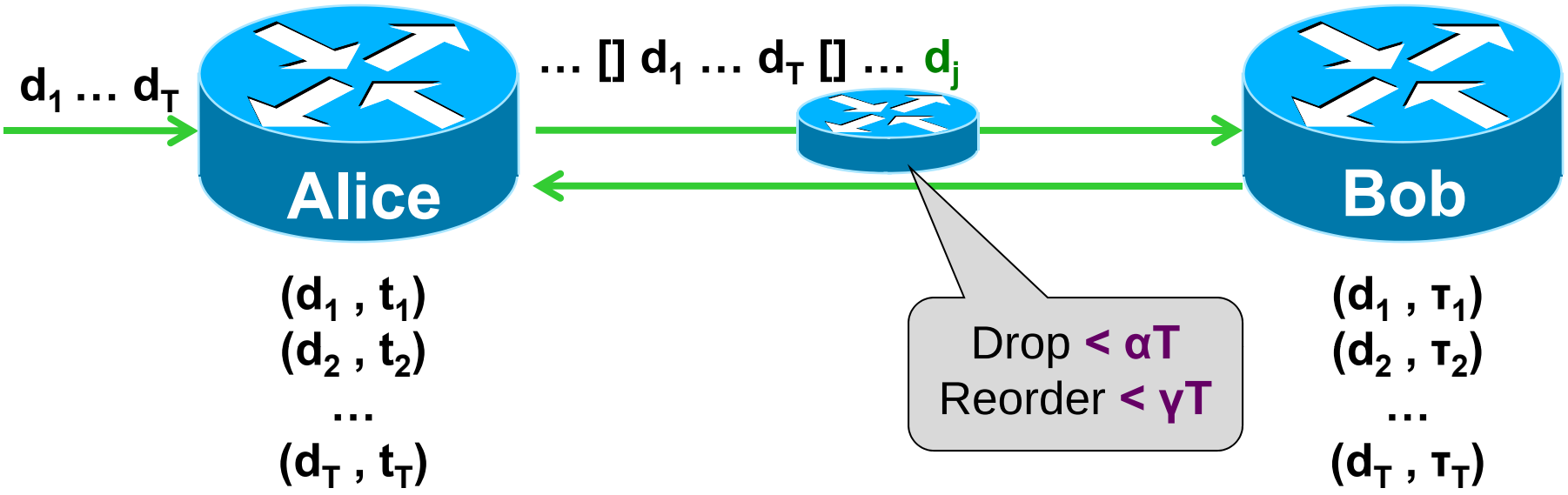
- Compute the ℓ_2-norm ΣX_i^2
- Raise an alarm iff norm **> 0.66%**

Refresh hash key & Repeat

Refresh hash key & Repeat

Delay : Accumulating Timestamps

Benign case

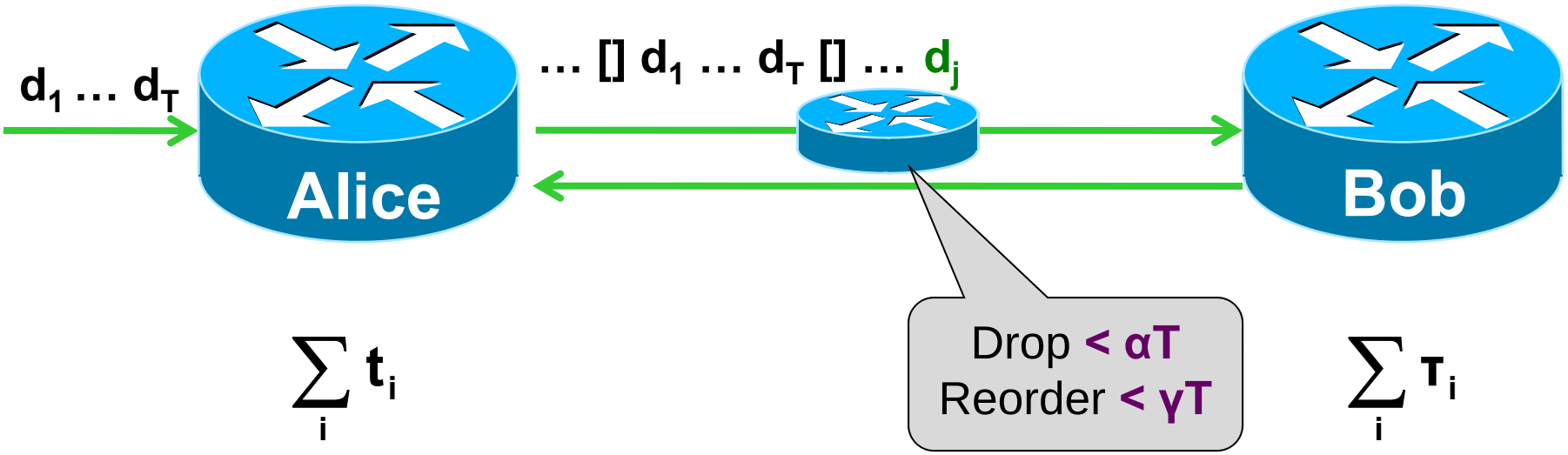


$$\text{Average delay} = \frac{1}{T}\left(\sum_i t_i - \sum_i \tau_i\right)$$

Need to store T packets and timestamps. ☹

Delay : Accumulating Timestamps

Benign case



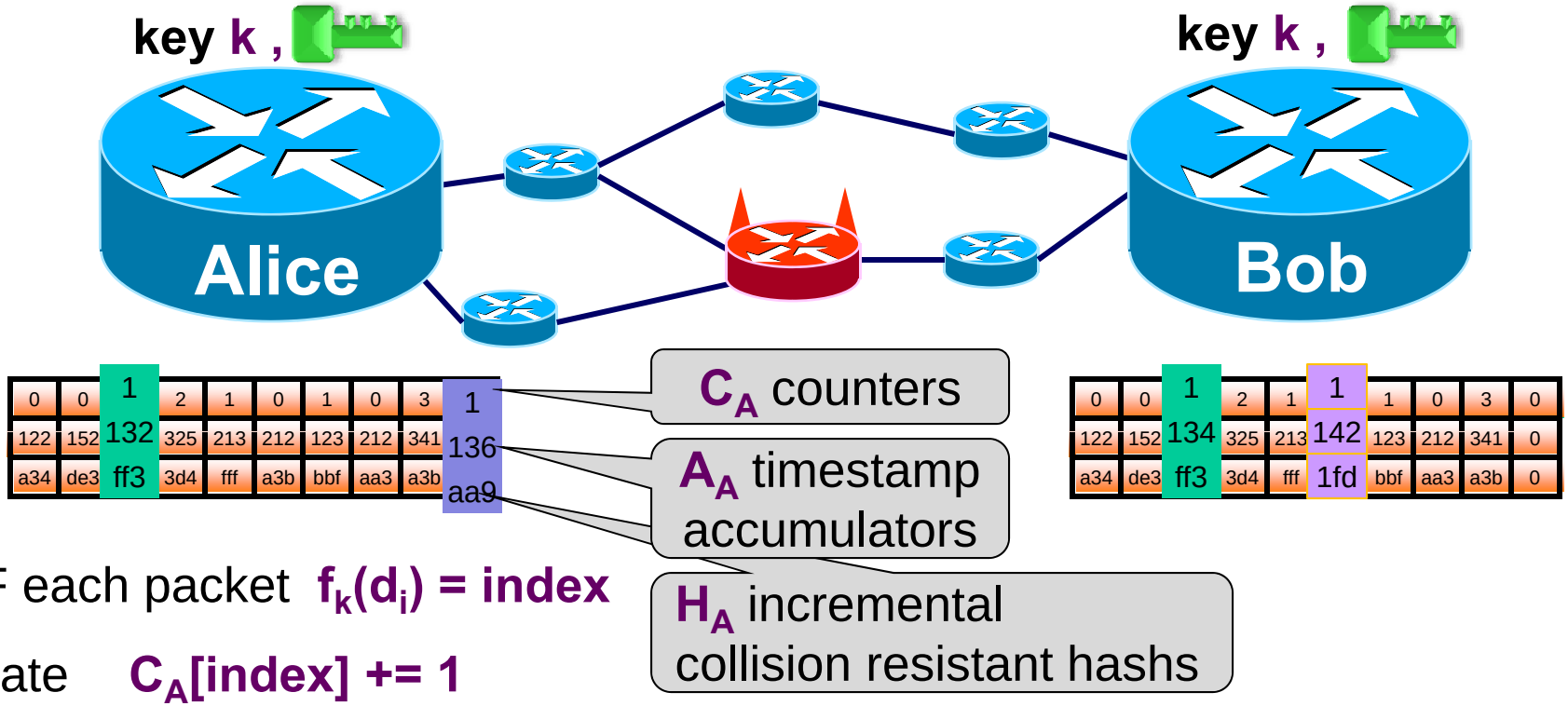
$$\text{Average delay} = \frac{1}{T}\left(\sum_i t_i - \sum_i \tau_i\right)$$

Can store only 1 timestamp! ☺

Bottom line: This trick works only if Alice & Bob agree on the same set of packets.

What happens if d_j is dropped / reordered?

Delay : Robust Lossy Difference Aggregator



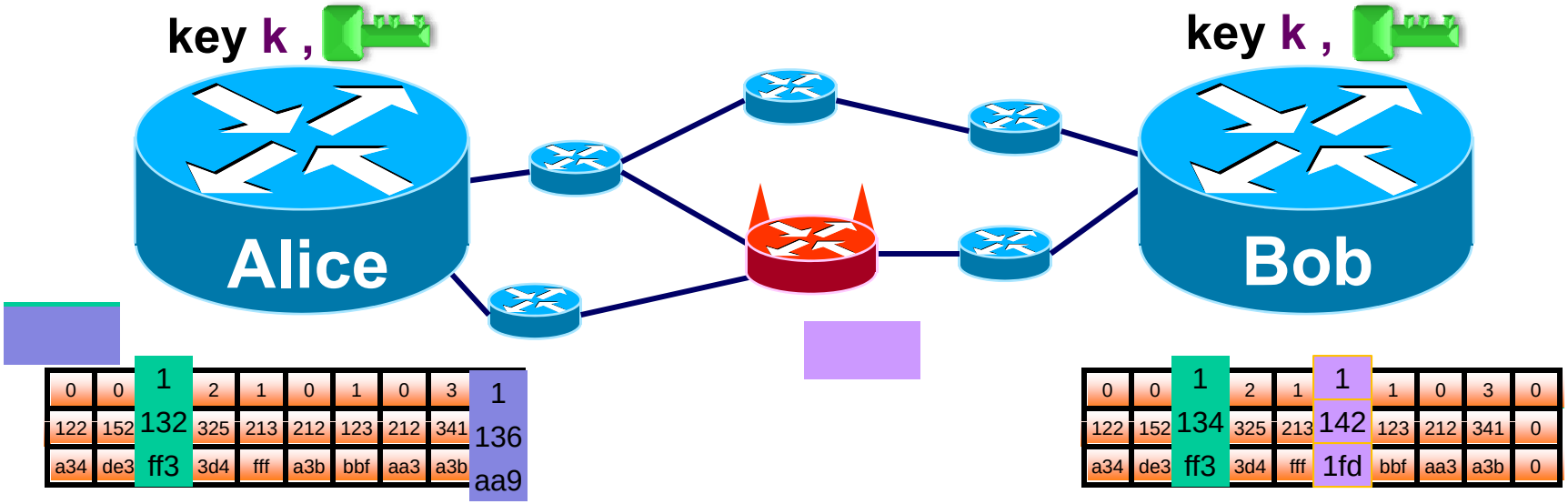
PRF each packet $f_k(d_i) = index$

Update $C_A[index] \mathrel{+}= 1$

$A_A[index] \mathrel{+}= t_i$

$H_A[index] \mathrel{*}= h(d_i)$

Delay : Robust Lossy Difference Aggregator



PRF each packet $f_k(d_i) = index$

Update $C_A[index] \mathrel{+}= 1$
$A_A[index] \mathrel{+}= t_i$
$H_A[index] \mathrel{*}= h(d_i)$

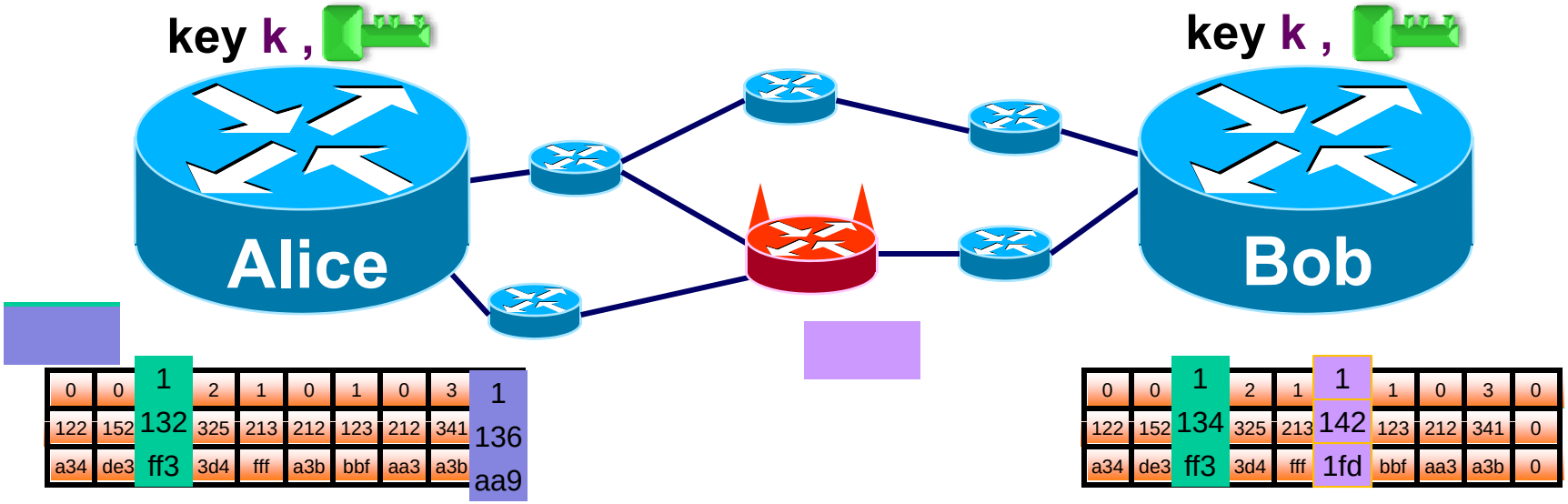
PRF each packet $f_k(d_i) = index$

Update $C_B[index] \mathrel{+}= 1$
$A_B[index] \mathrel{+}= \tau_i$
$H_B[index] \mathrel{*}= h(d_i)$

[interval end: A_A, H_A, C_A]

[Report: …]

Delay : Robust Lossy Difference Aggregator



PRF each packet $f_k(d_i) = index$

Update $C_A[index] \mathrel{+}= 1$
$A_A[index] \mathrel{+}= t_i$
$H_A[index] \mathrel{*}= h(d_i)$

PRF each packet $f_k(d_i) = index$

Update $C_B[index] \mathrel{+}= 1$
$A_B[index] \mathrel{+}= \tau_i$
$H_B[index] \mathrel{*}= h(d_i)$

[interval e

For every **index** s.t. $H_A[index] = H_B[index]$ take

$$\textbf{Avg Delay} = \frac{A_A[index] - A_B[index]}{C_A[index]}$$