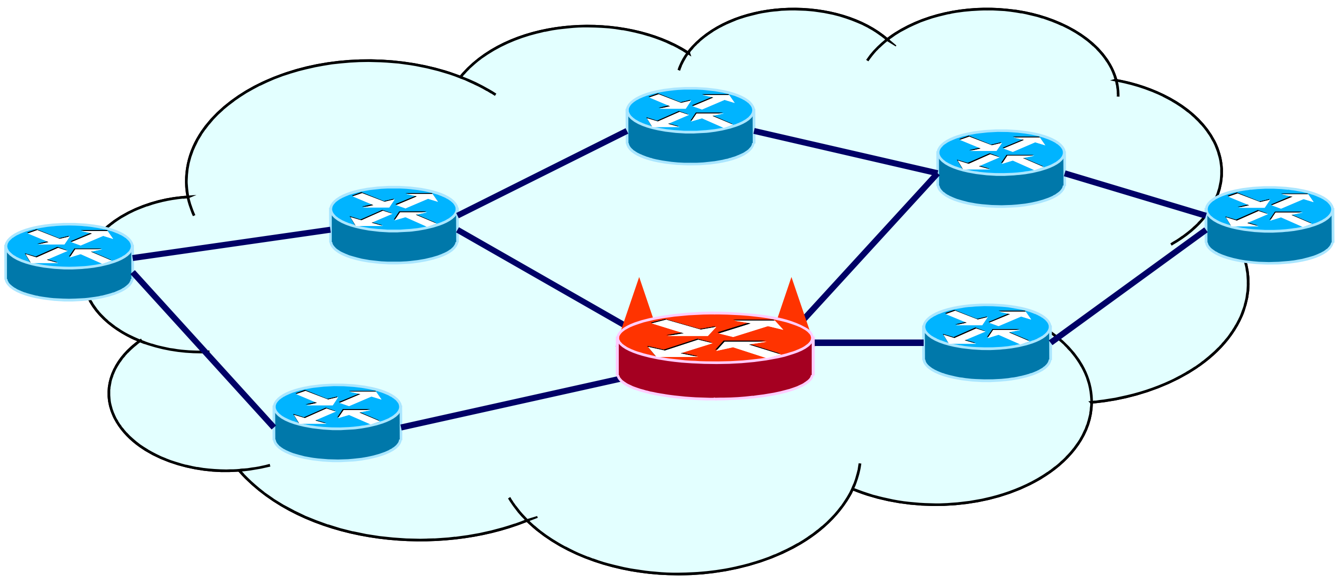
Internet Path-Quality Monitoring in the Presence of Adversaries



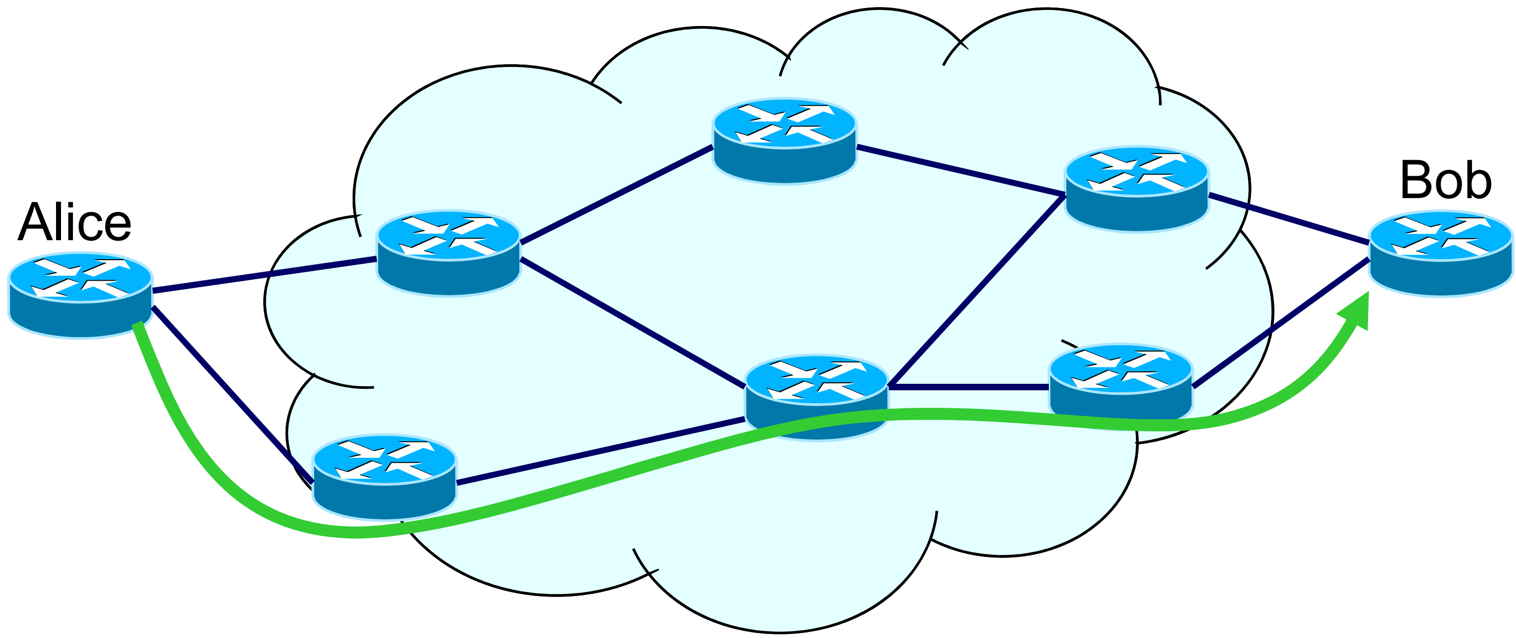
Sharon Goldberg

David Xiao, Eran Tromer, Boaz Barak, Jennifer Rexford

Princeton University

Excerpts of talks that have been presented in seminars at Penn State University, IBM Research, Cisco, Ben Gurion University, and the Weizmann Institute.

Internet 101



Packets routed from Alice to Bob via a path of intermediate routers

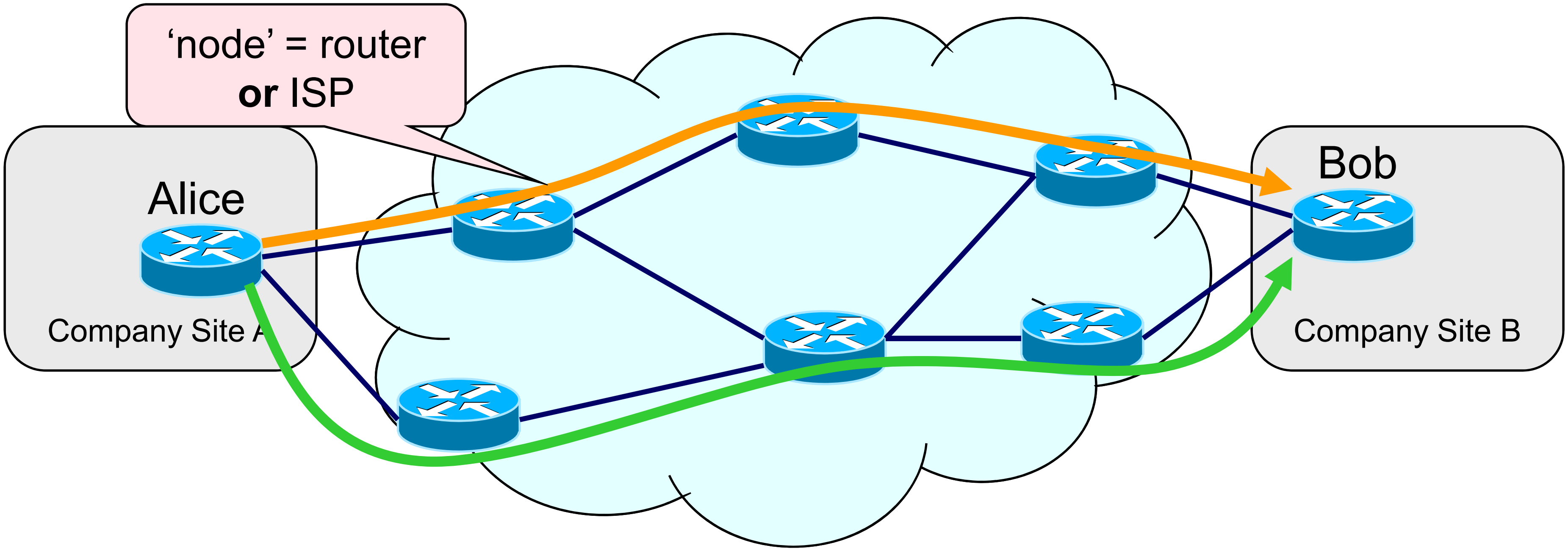
Routing protocols used to set up paths between routers

Today's focus

Packets forwarded along these paths with best-effort delivery

- No guarantees on packet arrival or integrity
- Congestion (random packet dropping) and reordering

Applications of path-quality monitoring



Routers need tools to detect unacceptably high packet loss rates.

<u>Performance Routing</u>

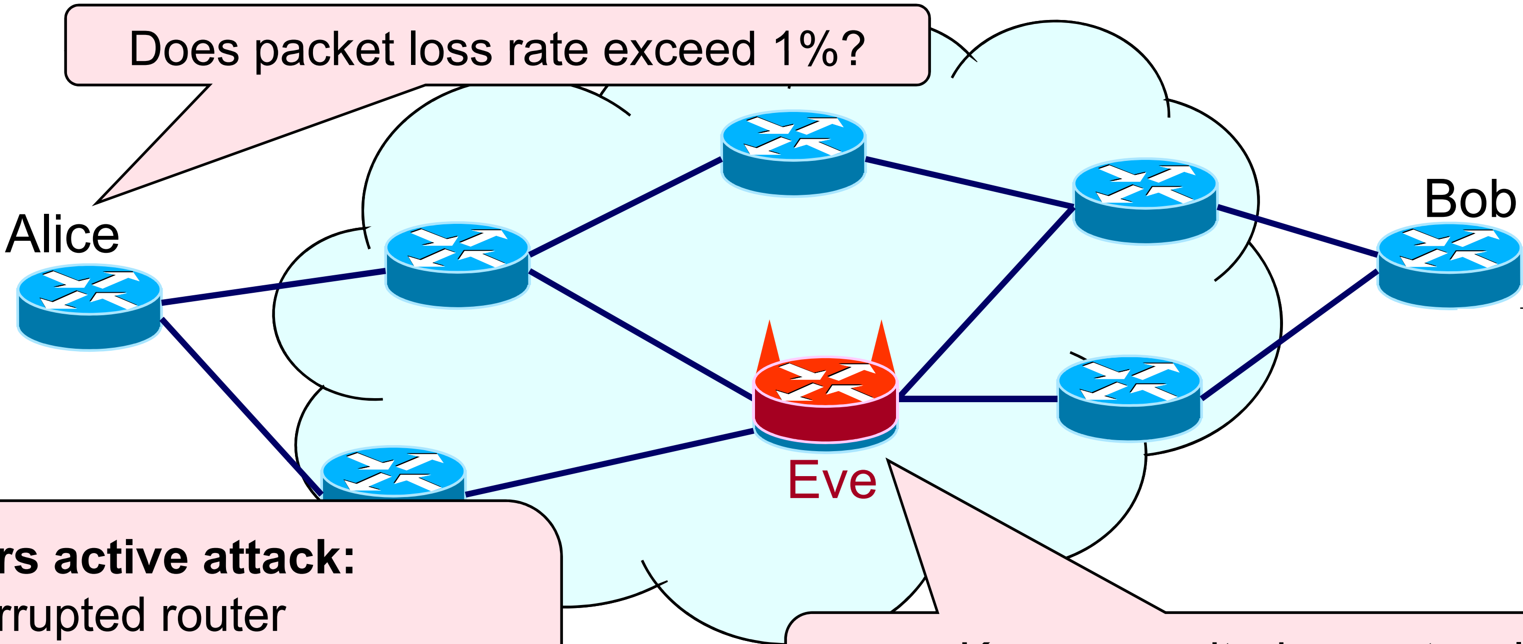
- Balancing loads between multiple paths (e.g. multihomed company sites)
- Quick response for avoiding blackholed routes and brownouts
- Avoiding "suspicious" paths (e.g. that drop Skype pkts, or corrupt traffic)

<u>SLA compliance monitoring</u>

- e.g. Cisco IP SLA's – detects end-to-end performance degradation

The presence of adversaries

Does packet loss rate exceed 1%?



Covers active attack:
- Corrupted router
- Botnet
- Greedy ISP

And *all* benign failures.

Knows monitoring protocol
Wants to hide packet loss from Alice

Can we have both?

Strong threat model - Eve can drop/delay/reorder/**add**/**modify** packets

Efficient protocols for high-speed routers

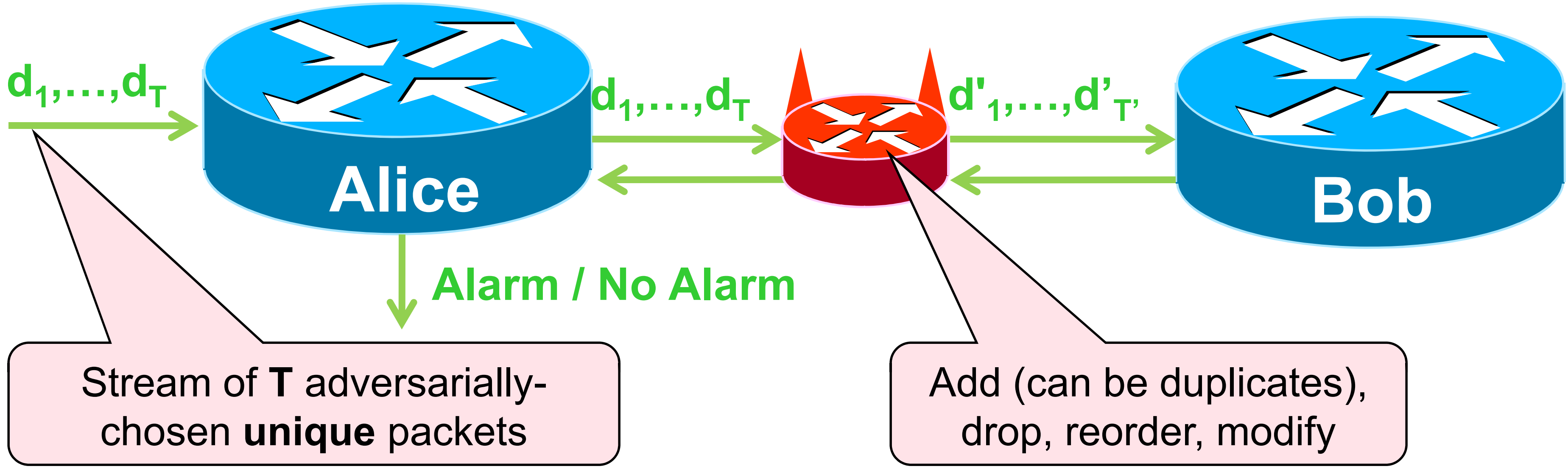
- Extremely limited storage, communication, computation
- No marking or encryption of existing traffic

This talk

1. Overview √
2. Defining secure PQM
3. Secure Sketch PQM
4. PQM and the adversarial sketch model
5. Public-Key / Client-Server PQM protocols
6. Conclusion

Formal Definition of PQM (1)

Malicious case



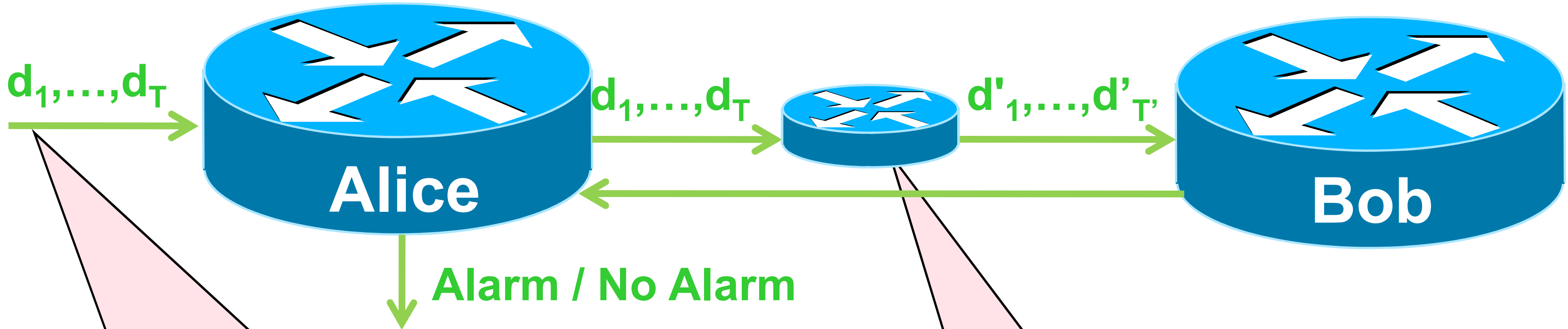
Secure path quality monitoring (PQM)

With probability **1- δ = 99%** ,

- Alice **alarms** if packet loss rate exceeds **β** *regardless of Eve's actions*

Formal Definition of PQM (2)

Benign case



Stream of **T** adversarially-chosen **unique** packets

Randomly drop **< αT** , reorder

Secure path quality monitoring (PQM)

With probability **1- δ = 99%** ,

- Alice **alarms** if packet loss rate exceeds **β** *regardless of Eve's actions*
- Alice **will not alarm** if packet loss rate is less than **α** *in benign case*

T > some function of **α,β**

Main result: For every **α <β <1** and security parameter **k** there exists a PQM protocol with **O(k+log(T))** communication and storage, one hash computation / packet and no packet marking.

Overview of (some of) our results

Secure path quality monitoring (PQM)

With probability **1- δ = 99%** ,

- Alice **alarms** if packet loss rate exceeds **β** *regardless of Eve's actions*
- Alice **will not alarm** if packet loss rate is less than **α** *in benign case*

T > some function of **α,β**

Main result: For every **α <β <1** and security parameter **k** there exists a PQM protocol with **O(k+log(T))** communication and storage, one hash computation / packet and no packet marking.

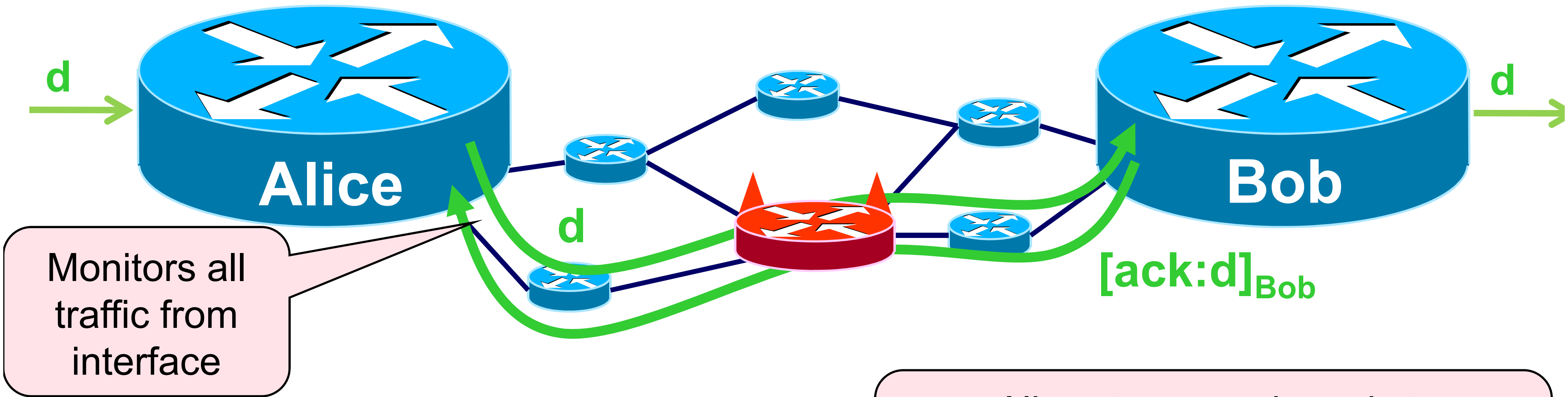
Analysis
α = 0.5% β = 1%
⇓
storage = 540 bytes
T = 10^9 packets

Simulations
α = 0.5% β = 1%
⇓
storage = 170 bytes
T = 10^6 packets

This talk

1. Overview √
2. Defining secure PQM √
3. Secure Sketch PQM
4. PQM and the adversarial sketch model
5. Public-Key / Client-Server PQM protocols
6. Conclusion

Background: Secure PQM



Trivial PQM:

Bob acks *each* packet.

Alice detects loss if a packet is not ack'd

Alice stores each packet.
100% communication overhead.
Not practical for network layer!

We want to avoid encrypting all traffic.

Other related work:

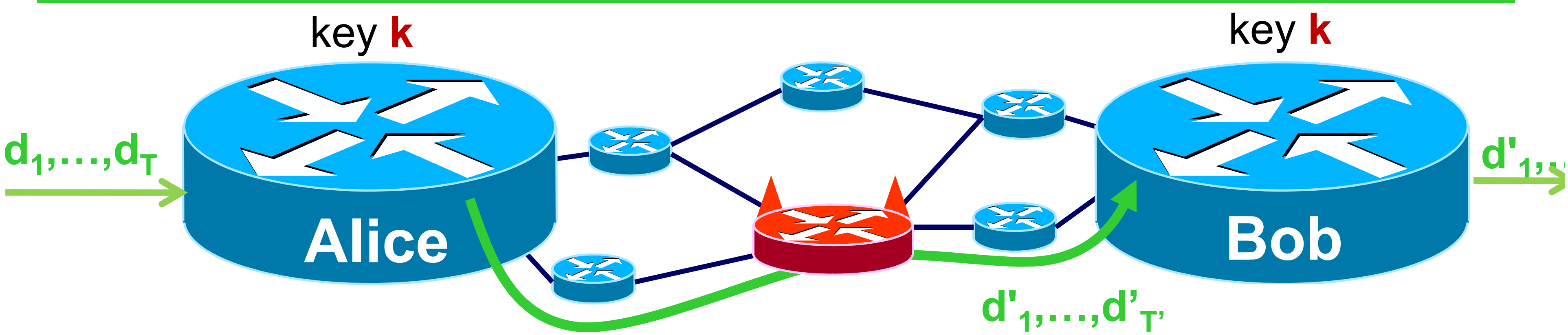
[IPsec] No acks. High overhead.

[AR06] Mark and monitor only a fraction of traffic . Encrypt to hide mark.

[MSMC05] Fatih and [SRSSK04] Listen, both insecure in our model.

Secure Sketch PQM: The Protocol

Uses ℓ_2-norm estimation sketches: [AMS96] [Ach01] [CCF2004] [TZ2004]



A

0	0	0	-2	1	0	1	0	3	0

B

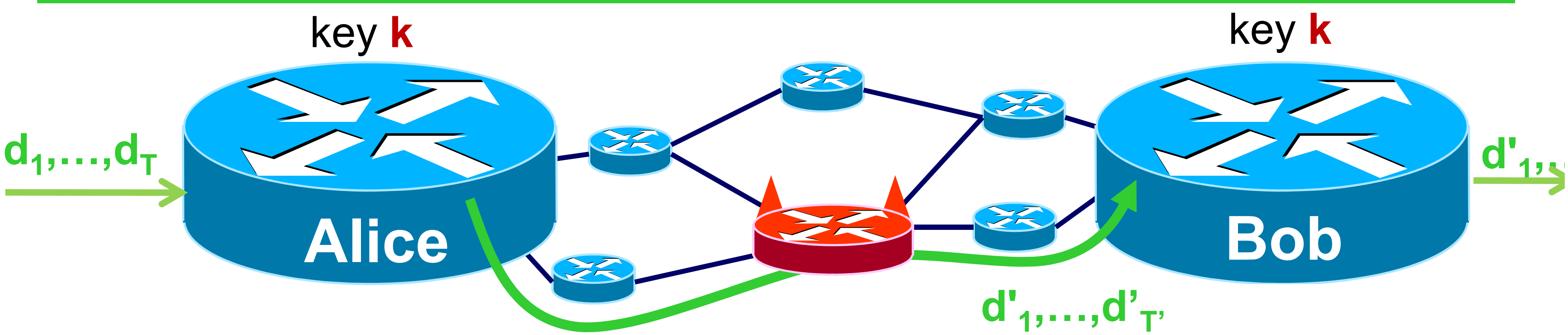
0	0	0	-1	1	0	-1	0	3	0

Hash each packet **$f_k(d)$ = index , bit**
Update sketch **A[index] += bit**

Hash each packet **$f_k(d)$ = index , bit**
Update sketch **B[index] += bit**

Secure Sketch PQM: The Protocol

Uses ℓ$_2$-norm estimation sketches: [AMS96] [Ach01] [CCF2004] [TZ2004]



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

PRF: $f_k()$: packets → [N]x{1,-1}

| -1 | 1 | 0 | -1 | 0 | 3 | 0 | **B**

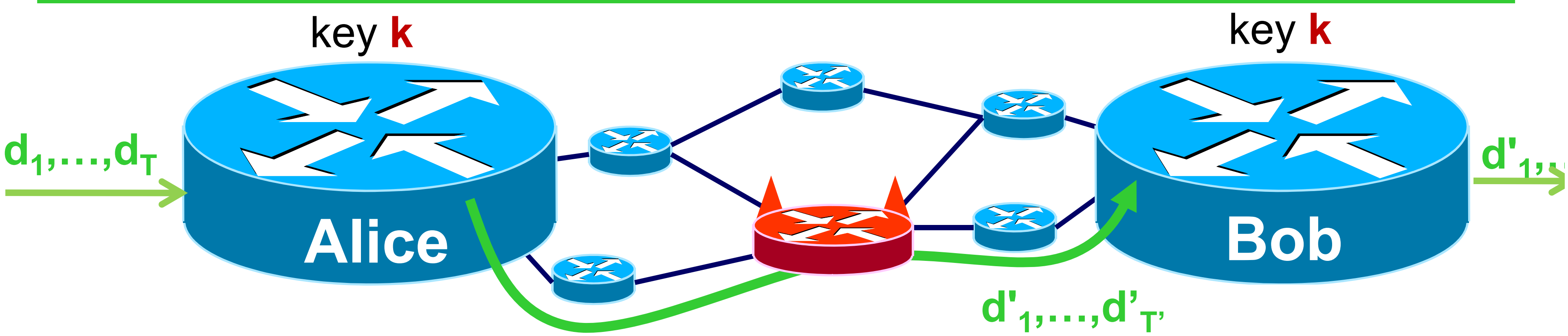
Hash each packet $\mathbf{f_k(d) = index\ ,\ bit}$
Update sketch $\mathbf{A[index] \mathrel{+}= bit}$

Hash each packet $\mathbf{f_k(d) = index\ ,\ bit}$
Update sketch $\mathbf{B[index] \mathrel{+}= bit}$

$[\text{report: B}]_{\text{Bob}}$

Secure Sketch PQM: The Protocol

Uses ℓ_2-norm estimation sketches: [AMS96] [Ach01] [CCF2004] [TZ2004]



A

0	0	0	-2	1	0	1	0	3	0

B

0	0	0	-1	1	0	-1	0	3	0

Hash each packet **$f_k(d)$ = index , bit**
Update sketch **A[index] += bit**

Hash each packet **$f_k(d)$ = index , bit**
Update sketch **B[index] += bit**

Send authenticated (MAC'd) sketch

To decide between loss rate **$< \alpha$** and **$> \beta$**:

- Take difference sketch **X = A-B**
- Compute its ℓ_2-norm **ΣX_i^2**
- Raise an alarm iff **$\Sigma X_i^2 / T > (\alpha+\beta)/2$**

Actually, we used a different threshold to optimize constants

Refresh hash key & Repeat

Refresh hash key & Repeat

Secure Sketch PQM: Analysis

Thm (Simplified): Alice can use (CCF-based) secure sketch PQM protocol to decide between cases where packet loss rate is **$< \alpha$** and **$>$ $\beta = 2\alpha$**, with **$1-\delta$** success probability if the sketch has

$$N > 65 \ln (100 / 99\delta) \quad \text{bins}$$

and $$T > 867 N (\ln (100 N / \delta)) / \alpha \quad \text{packets monitored per interval.}$$

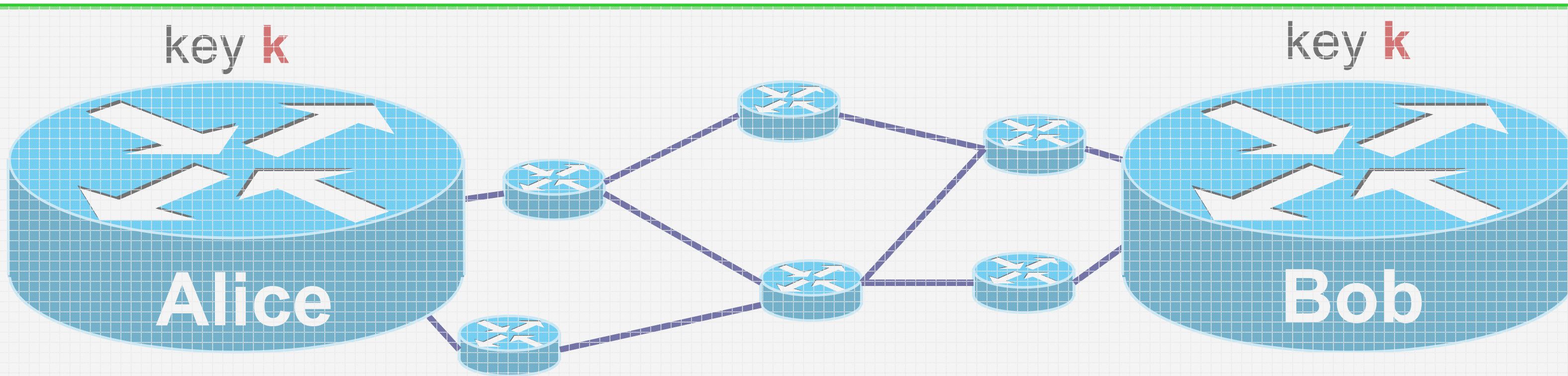
Analysis
$\alpha = 0.5\%$ $\delta = 1\%$
$\Downarrow$
$N = 300$ $T = 10^9$

Simulations
$\alpha = 0.5\%$ $\delta = 1\%$
$\Downarrow$
$N = 150$ $T \geq 10^6$

I'll show the proof for PQM using classic ℓ_2-norm sketches [AMS96] [Ach01]

Secure Sketch PQM: The "Classic" Version

Uses ℓ_2-norm estimation sketches: [AMS96] [Ach01] [CCF2004] [TZ2004]



A

0	0	0	-2	1	0	1	0	3	0

PRF: $f_k()$: packets $\rightarrow \{1,-1\}^N$

-1	1	0	-1	0	3	0

B

Hash each packet $f_k(d) = [b_1 \ldots b_N]$
Update sketch $B \mathrel{+}= [b_1 \ldots b_N]$

Hash each packet $f_k(d) = [b_1 \ldots b_N]$
Update sketch $B \mathrel{+}= [b_1 \ldots b_N]$

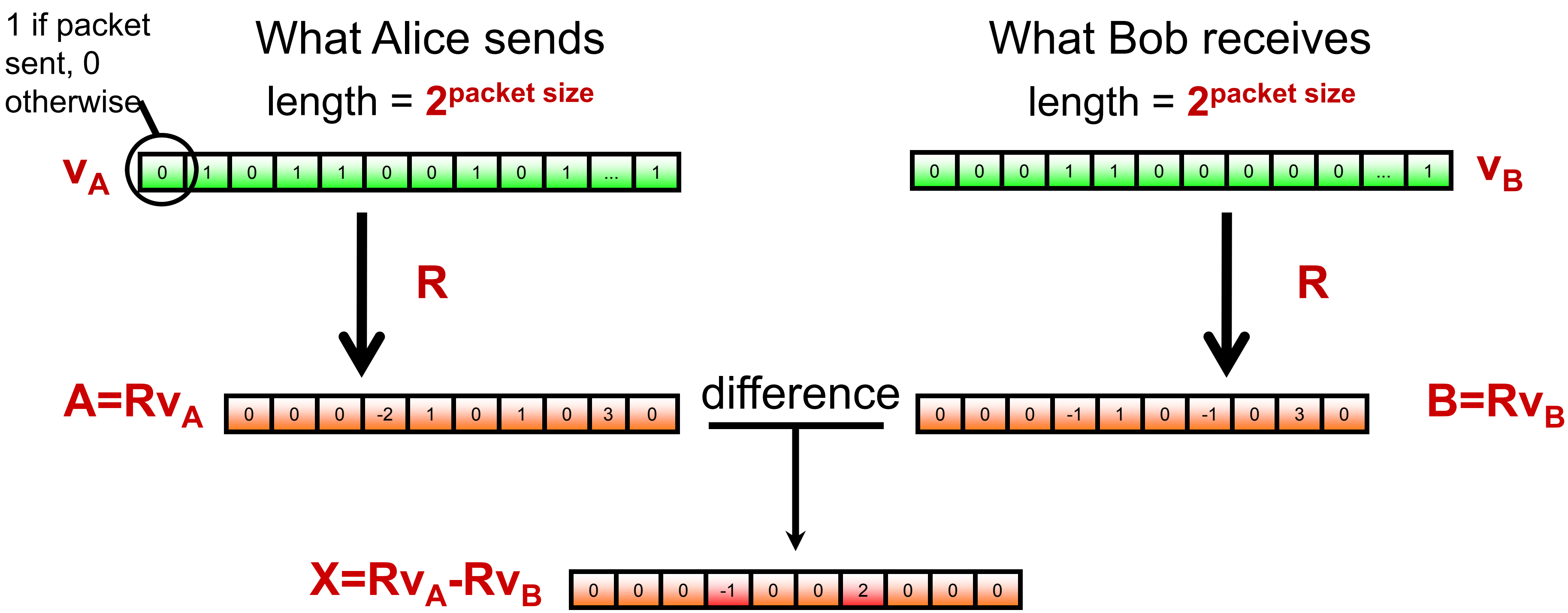
Send authenticated (MAC'd) sketch

To decide between loss rate $< \alpha$ and $> \beta$:

- Take difference sketch $X = A - B$
- Compute its ℓ_2-norm $\Sigma X_i^2 / N$
- Raise an alarm iff $\Sigma X_i^2 / NT > 2\,\alpha\beta / (\alpha+\beta)$

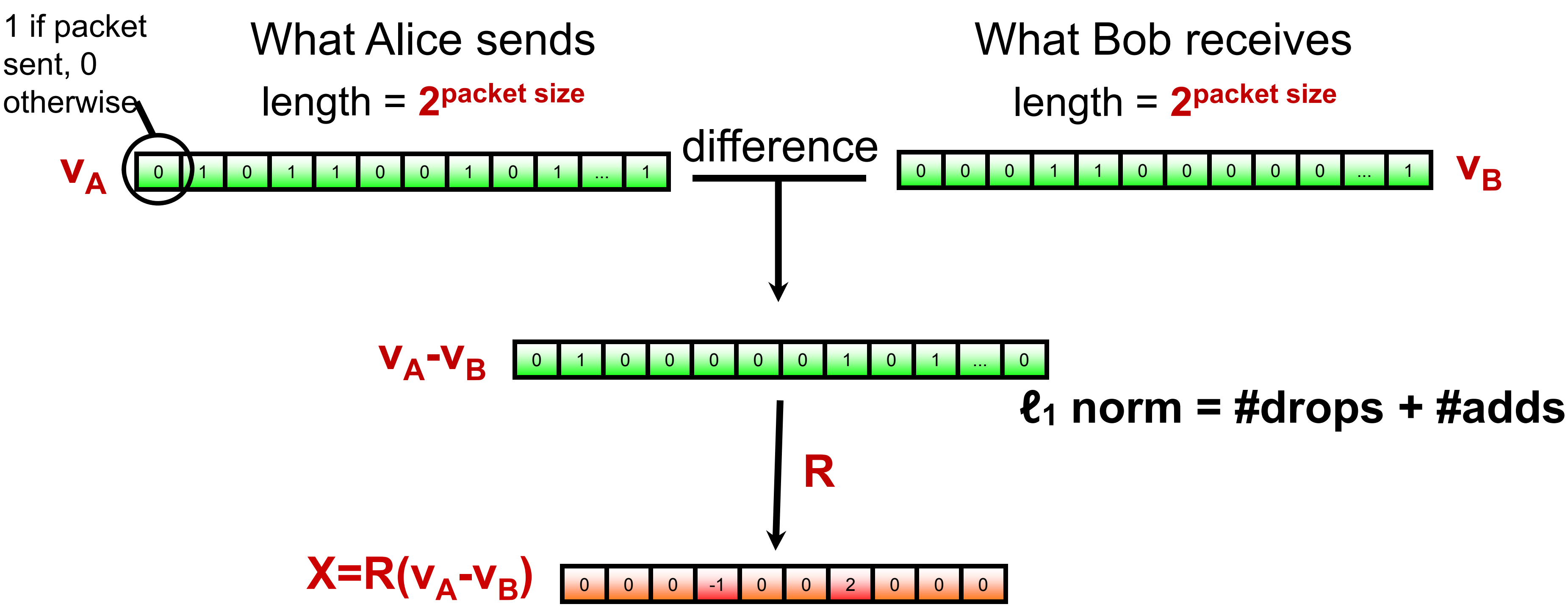
Analysis with “classic” sketching (1)

PRF → random function
Packet Stream → long vector
Hashing packet → multiplication by random **R** in $\{-1,1\}^{N \times 2^{\text{packet size}}}$



Analysis with “classic” sketching (2)

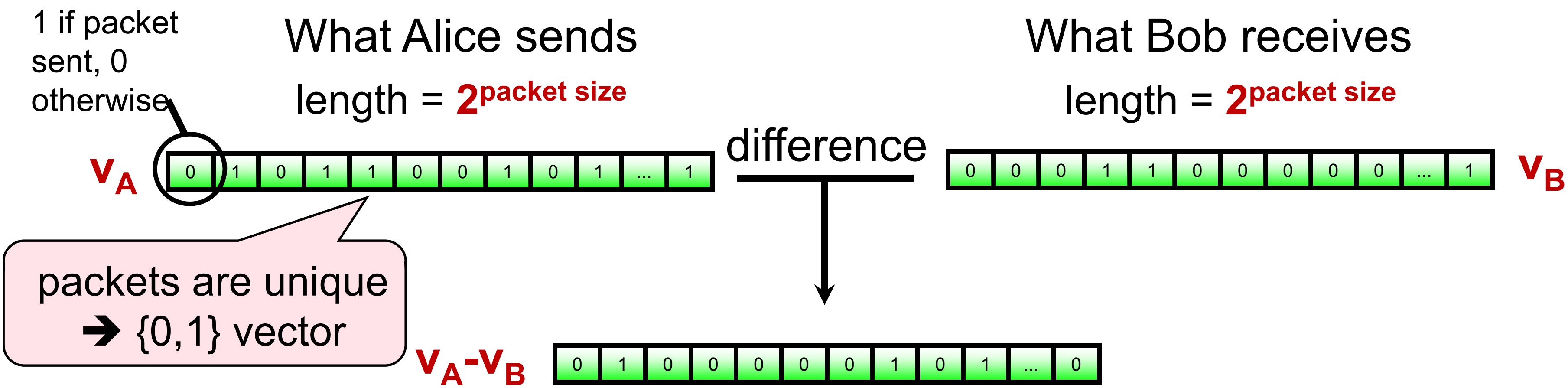
PRF → random function
Packet Stream → long vector
Hashing packet → multiplication by random **R** in $\{-1,1\}^{N \times 2^{\text{packet size}}}$



If we used a good ℓ_1 norm estimation sketch, we’d be done. But we use (more efficient) ℓ_2 norm estimation

Analysis with "classic" sketching (3)

PRF → random function
Packet Stream → long vector
Hashing packet → multiplication by random **R** in $\{-1,1\}^{N\times 2^{\text{packet size}}}$



ℓ_1 norm = #drops + #adds

Benign case:

- no adds → v_B subset v_A
- $\mathbf{v_A - v_B}$ is {0, 1} vector
- $\boldsymbol{\ell_1 = \ell_2^2}$ for {0, 1} vectors
- $\boldsymbol{\ell_2^2}$ sketch estimates #drops

Malicious case:

- can have duplicate adds
- $\mathbf{v_A - v_B}$ is {0, 1, -1,-2,…} vector
- $\boldsymbol{\ell_1 \leq \ell_2^2}$ (= unique adds)
- $\boldsymbol{\ell_2^2}$ sketch (overestimates) **#drops**
- duplicates increase Pr[alarm]

Analysis with "classic" sketching (4)

$v_A - v_B$

0	1	0	0	0	0	0	1	0	1	...	0

Multipy by random $\mathbf{R}$ in $\{-1,1\}^{N\text{x}2\ \text{packet size}}$

$X = R(v_A - v_B)$

0	0	0	-1	0	0	2	0	0	0

"JL-Theorem" [Ach01]: For any (long) vector **v** and random **{-1,1}**-matrix mapping **v** to **N** dimensions, then w.p. $\exp(-O(N\varepsilon^2))$

$$(1-\varepsilon)\,\|v\|_2^2 < \|Rv\|_2^2 / N < (1+\varepsilon)\,\|v\|_2^2$$

Corollary: For error δ take a sketch of size $N = O(\log(1/\delta)1/\varepsilon^2)$

PQM Decision Rule: To decide between drop rate $< \alpha$ and $> \beta = 2\alpha$ with confidence $1-\delta$ alarm iff

$$\|R(v_A - v_B)\|_2^2 / N > 2\alpha\beta/(\alpha+\beta)\ T$$

and use sketch length $N = O(\log(1/\delta)\,(\beta+\alpha)^2/(\beta-\alpha)^2)$

Secure Sketch PQM: Analysis with CCF

Thm (Simplified): Alice can use (CCF-based) secure sketch PQM protocol to decide between cases where packet loss rate is **$< \alpha$** and **$>$ $\beta = 2\alpha$**, with **$1-\delta$** success probability if the sketch has

$N > 65 \ln (100 / 99\delta)$ bins

and **$T > 867 N (\ln (100 N / \delta)) / \alpha$** packets monitored per interval.

Analysis
$\alpha = 0.5\%$ $\delta = 1\%$
$\Downarrow$
$N = 300$ $T = 10^9$

Simulations
$\alpha = 0.5\%$ $\delta = 1\%$
$\Downarrow$
$N = 150$ $T \geq 10^6$

Simulations with CCF (1)

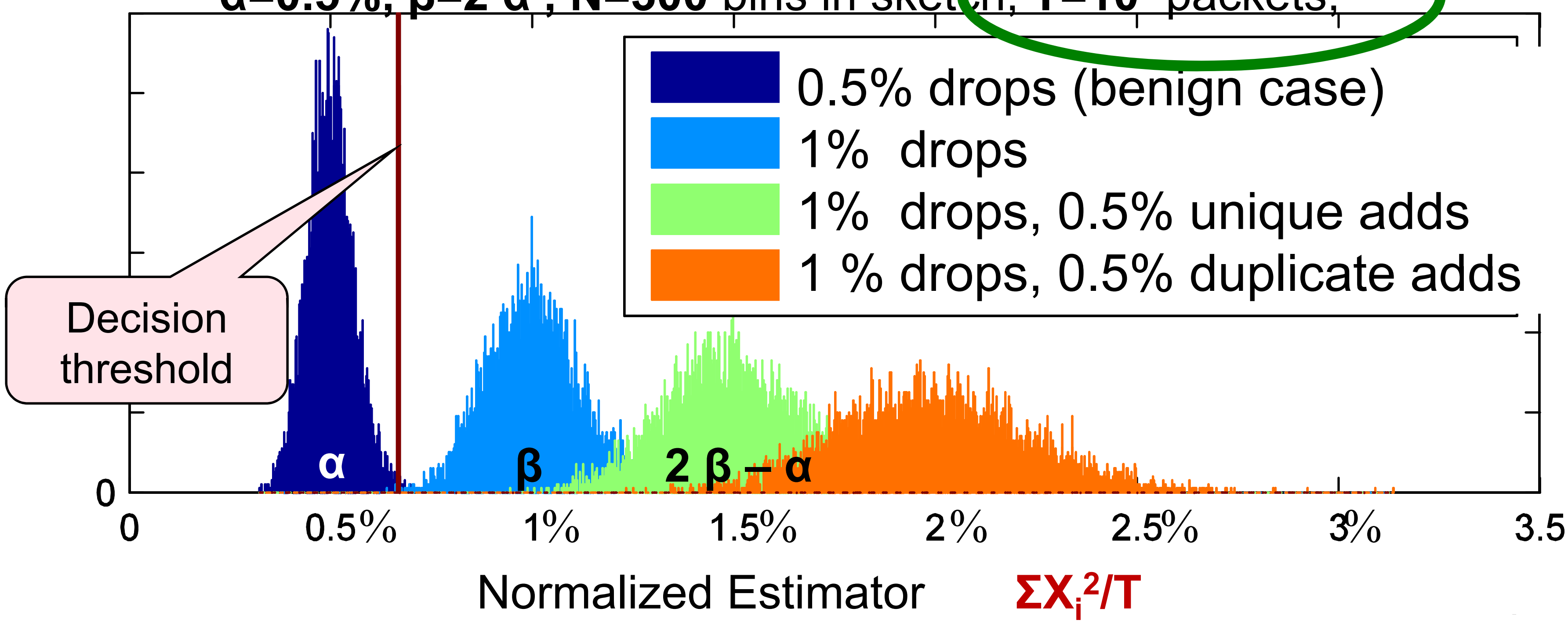
Analysis
$\alpha = 0.5\%$ $\delta = 1\%$
$\Downarrow$
$N = 300$ $T = 10^9$

Simulations
$\alpha = 0.5\%$ $\delta = 1\%$
$\Downarrow$
$N = 150$ $T \geq 10^6$

Recall that **$||R(v_A - v_B)||_2 = ||A-B||_2 \geq$ drops + adds**

Histogram of Estimator $||A-B||_2$

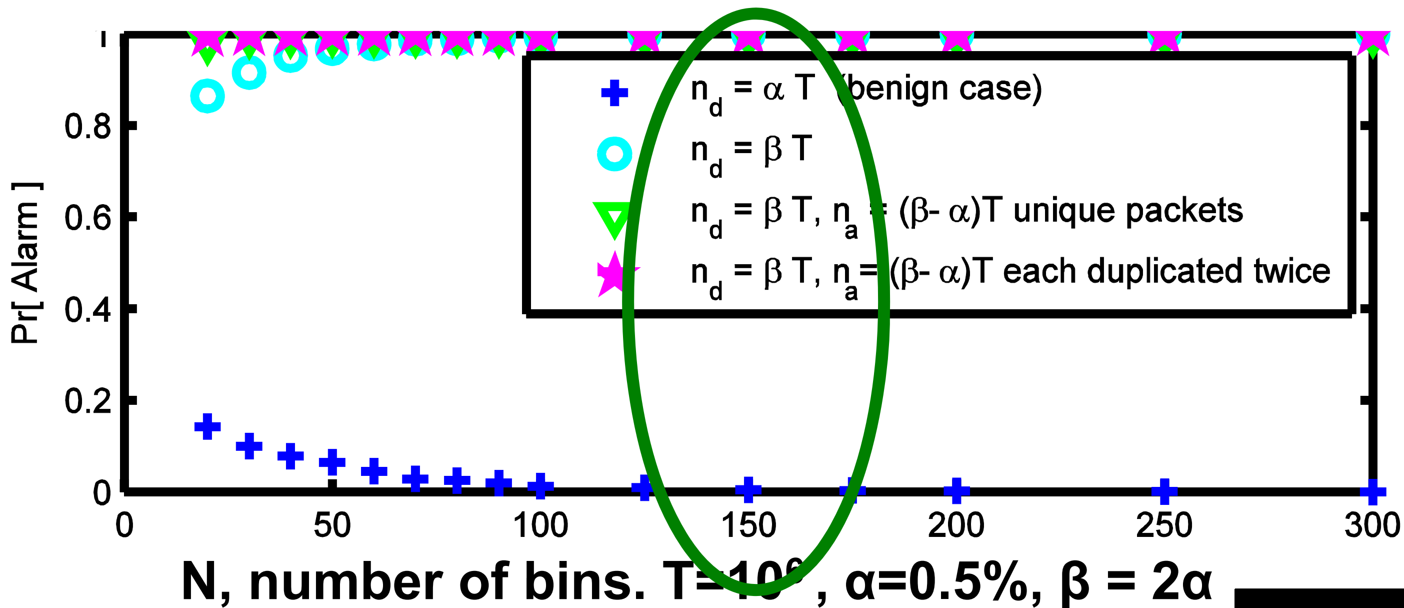
α=0.5%, β=2 α , N=300 bins in sketch, **$T=10^6$** packets,



Simulations with CCF (2)

Analysis
$\alpha = 0.5\%$ $\delta = 1\%$
$\Downarrow$
$N = 300$ $T = 10^9$

Simulations
$\alpha = 0.5\%$ $\delta = 1\%$
$\Downarrow$
$N = 150$ $T \geq 10^6$



If **N=150** ➡

T	Sketch Size
10^6	170 bytes
10^7	200 bytes
10^8	235 bytes
10^9	270 bytes

Secure Sketch PQM Summary

T	Sketch Size
10^6	170 bytes
10^7	200 bytes
10^8	235 bytes
10^9	270 bytes

Low storage overhead

Low communication overhead

- 1 report packet / T regular packets
- Report contains sketch and authenticator

No packet marking

- Protocol is backward compatible.
- Can be implemented *off* the fast path of the router

One cryptographic hash computation per packet

- *Online setting* so we can use fast hash functions
  - Even universal hash functions work!
- High-throughput
- Do not modify packets, so can compute hash after packet sent

Shared keys at Alice and Bob

- Can be derived from public key infrastructure via key exchange

Secure Sketch PQM Summary

- Low storage overhead
- Low communication overhead
  - **1** report packet / **T** regular packets
  - Report contains sketch and authenticator

T	Sketch Size
10^6	170 bytes
10^7	200 bytes
10^8	235 bytes
10^9	270 bytes

- No packet marking
  - Protocol is backward compatible.
  - Can be implemented *off* the fast path (

No information leaked until the report released, and by then the key is refreshed

- One cryptographic hash computation per pack
  - *Online setting* so we can use fast hash fu ctions
    - Even universal hash functions work!
  - High-throughput
  - Do not modify packets, so can compute hash after packet sent
- Shared keys at Alice and Bob

Thm [GXTBR08]: **Any secure PQM protocol** robust to adversarial nodes on the path that can **add/drop** packets, needs a key infrastructure and crypto.

This talk

1. Overview √
2. Secure Sketch PQM √
3. $\boxed{\text{Public-Key / Client-Server PQM protocol}}$
4. PQM and the adversarial sketch model
5. Conclusion

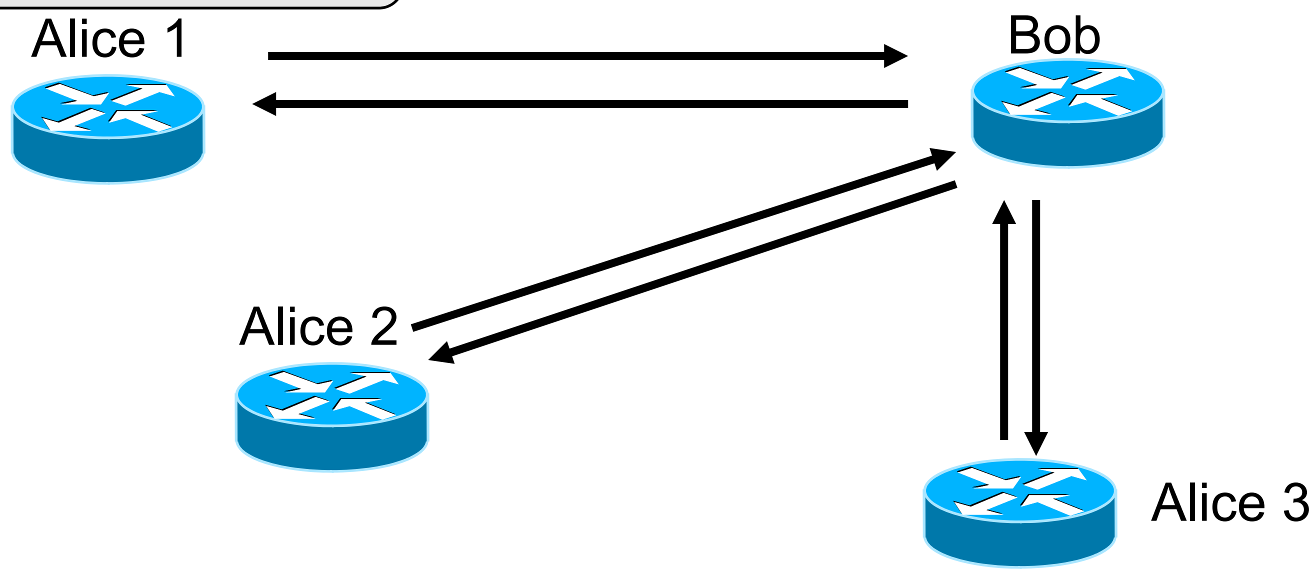
A Public-Key / Client-Server PQM Protocol (1)

Want: Server uses the same key for each client

Challenge: Client can be malicious

Public key operations are expensive

Can't share same symmetric key with many senders

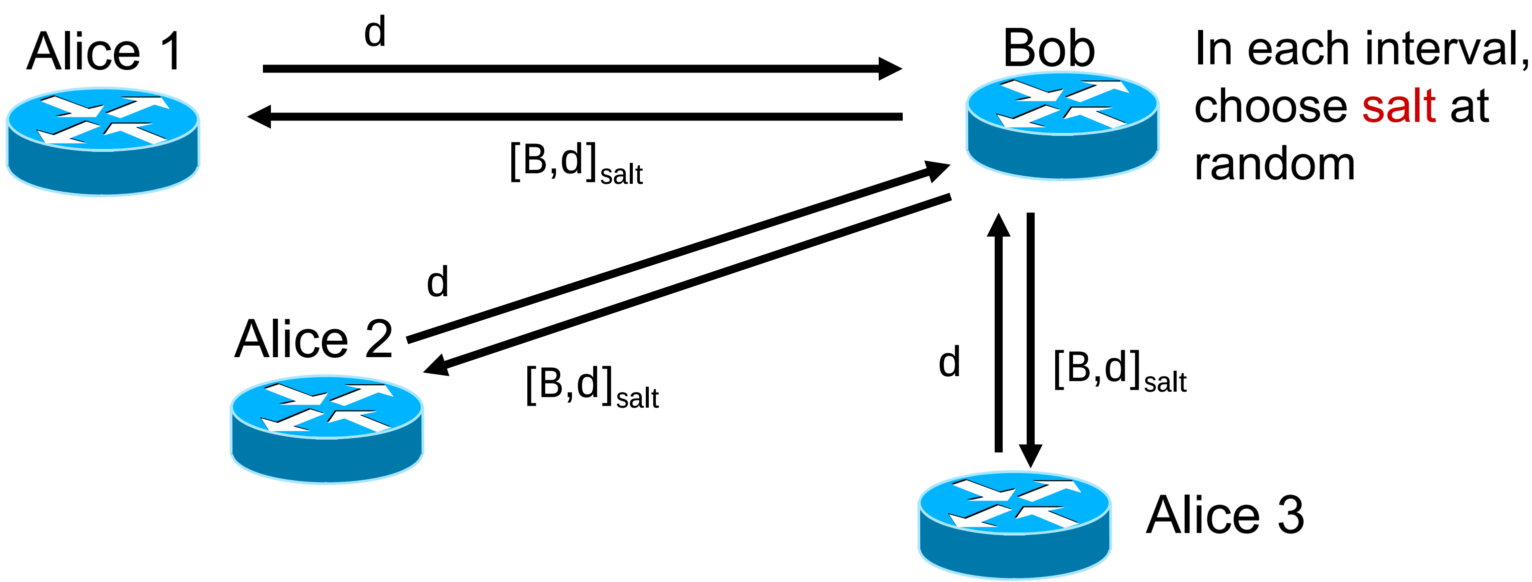


Solution: Bob uses a temporary key (**salt**) that is revealed after use

Run a **secure sampling** protocol using the salt

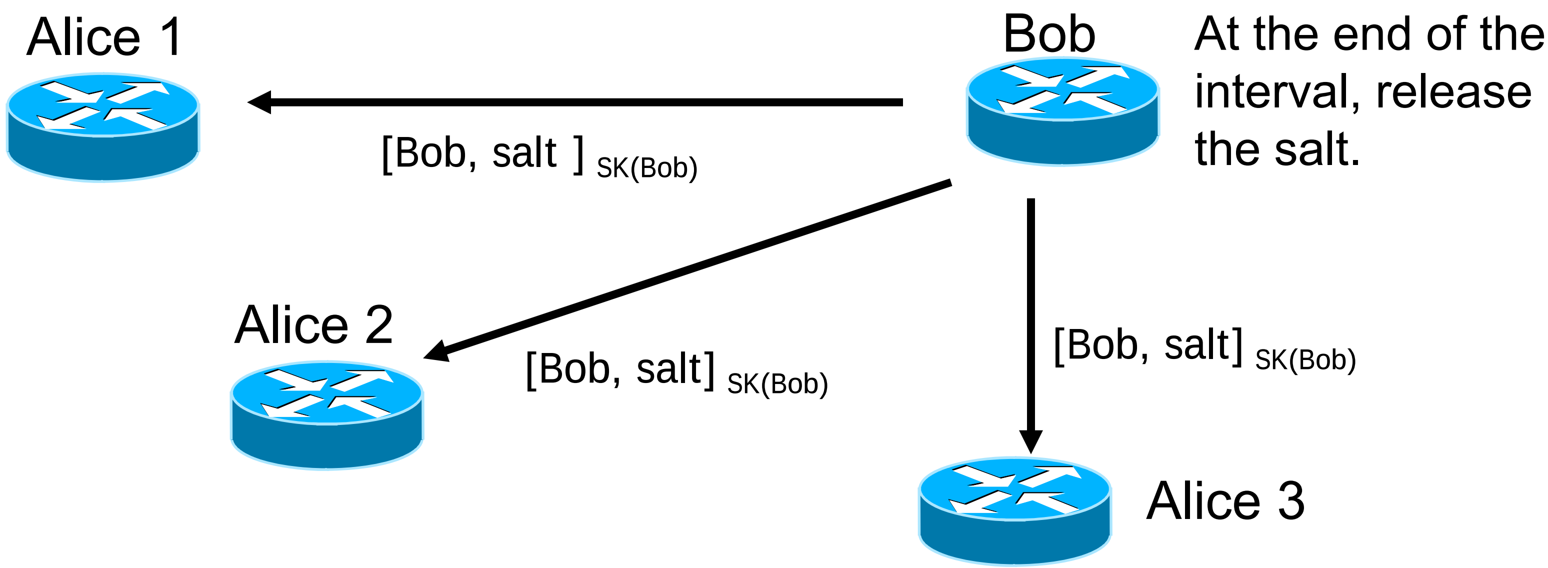
A Public-Key / Client-Server PQM Protocol (2a)

Receiver can respond to many senders with same salt and PK



A Public-Key / Client-Server PQM Protocol (2b)

Receiver can respond to many senders with same salt and PK



Similar to TESLA multi-cast signatures [PCST]

Client-Server Secure Sampling

Sampling rate is pq

Choose salt at random

With probability q
Store d (d is a probe)

If $f_{salt}(d) < p$
Send $[d]_{salt}$
Else send nothing

Use PK(Bob) to verify salt

$[\text{released: salt}]_{SK(Bob)}$

Use salt to check which probes needed and received valid ack

Alice stores: O(T) Communication: O(T)
Can we get **O(log T)** with sketching?

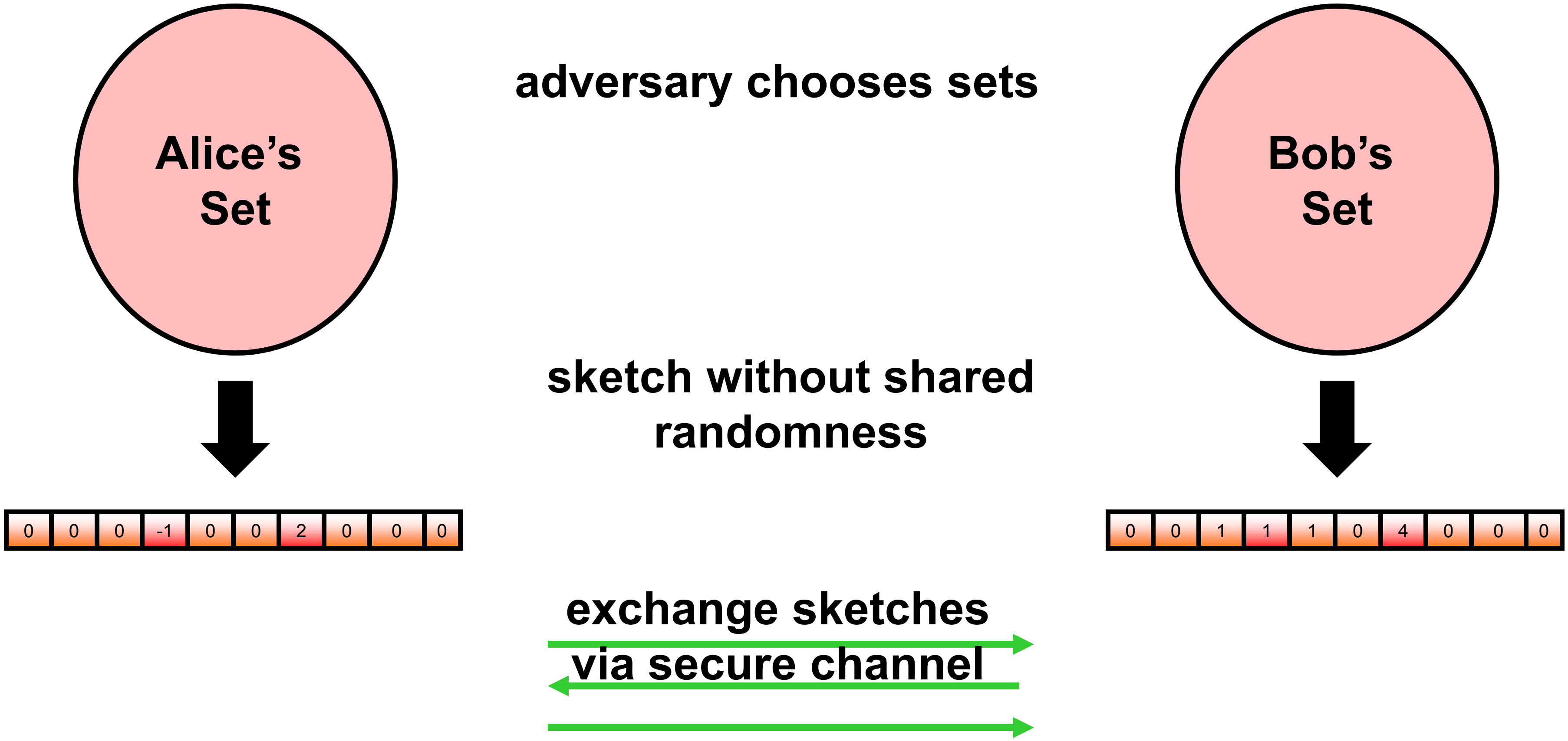
No!

This is the Adversarial Sketch Model

This talk

1. Overview √
2. Secure Sketch PQM √
3. Public-Key / Client-Server PQM protocol √
4. $\boxed{\text{PQM and the adversarial sketch model}}$
5. Conclusion

The Adversarial Sketch Model [MNS08]



Lower bound for norm of symmetric difference

| Alice Sketch | x | Bob Sketch | = O(|size of sets|)

Via reduction to equality testing in simultaneous communication model **[BK97]**

Symmetric-key PQM in Adversarial Sketch Model

Sent packets

adversary chooses sets
Received Packets

Received Packets

with private
sketch ~~without~~ shared randomness

0	0	0	-1	0	0	2	0	0	0

0	0	1	1	1	0	4	0	0	0

exchange sketches
via secure channel

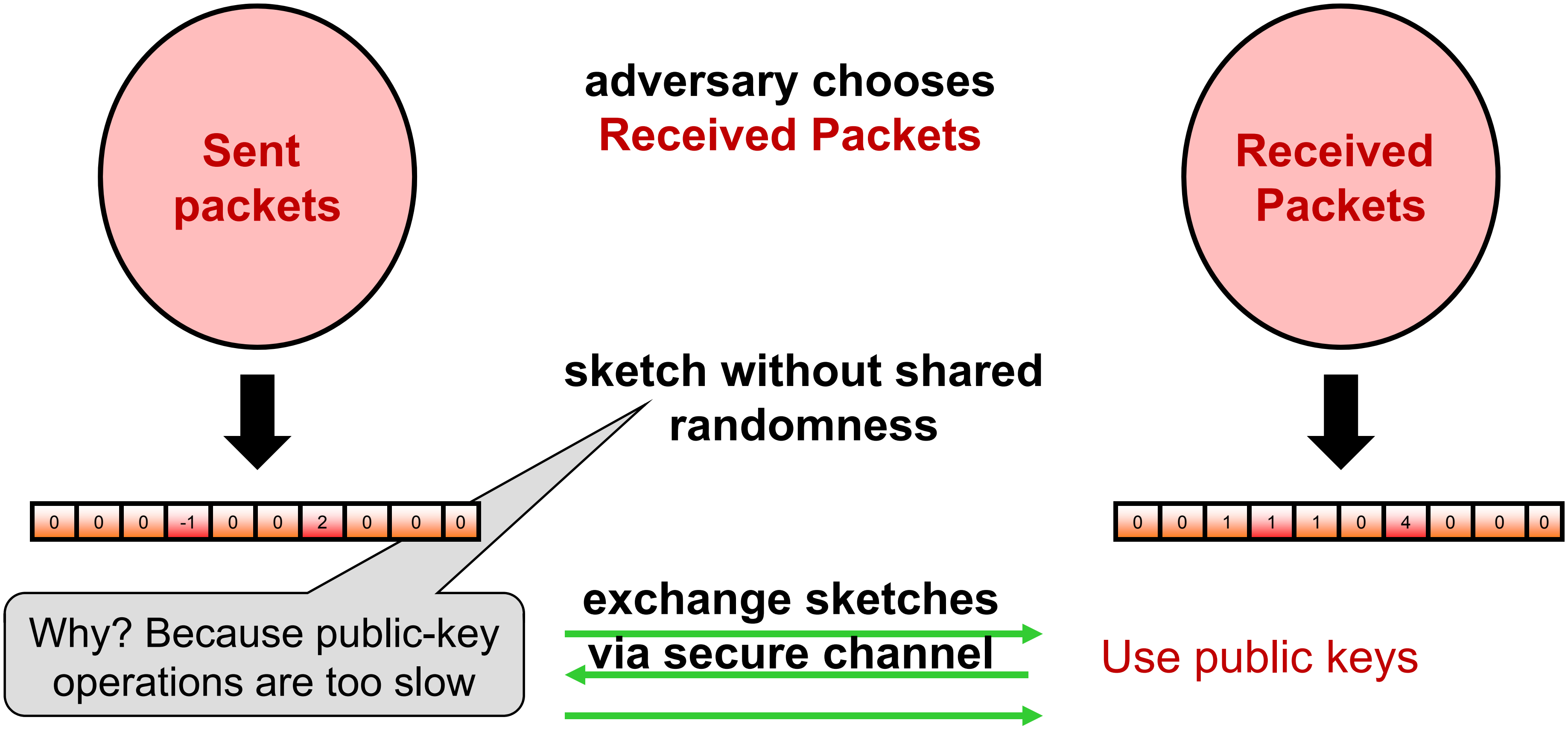
Use symmetric keys

Lower bound for norm of symmetric difference

| Alice Sketch | x | Bob Sketch | = Ω(|size of sets|)

Via reduction to equality testing in simultaneous communication model [BK97]

Public-key PQM in Adversarial Sketch Model



Lower bound for norm of symmetric difference

| Alice Sketch | x | Bob Sketch | = O(|size of sets|) =O(T)

Via reduction to equality testing in simultaneous communication model **[BK97]**

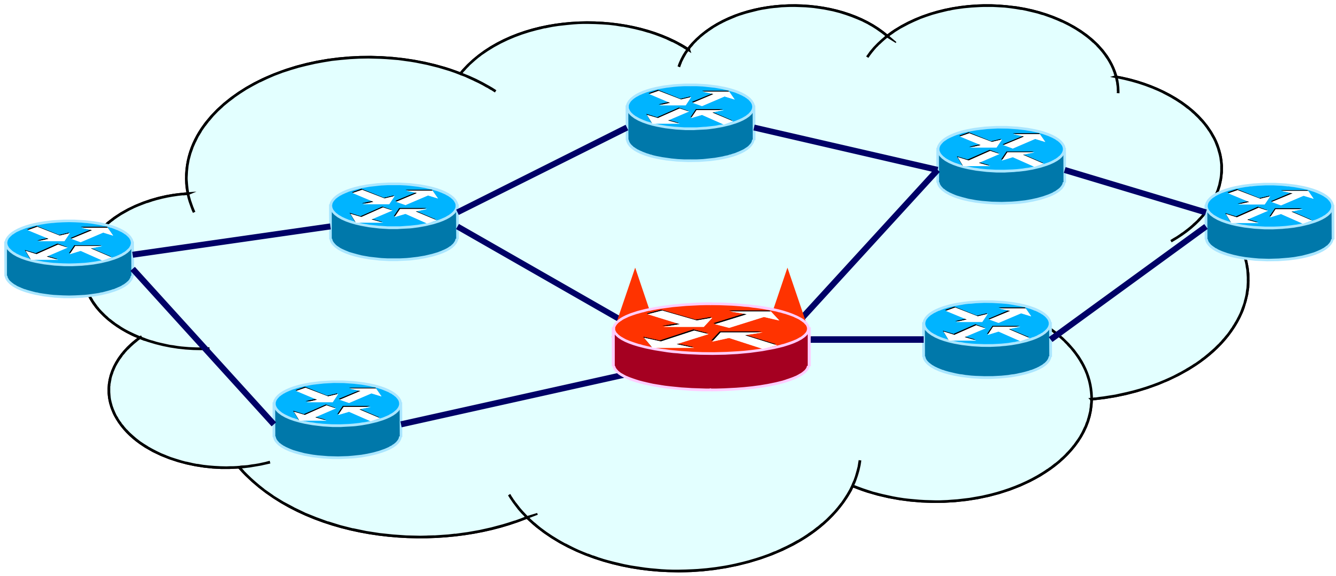
Conclusions

Sometimes we don't have to give up security for the sake of efficiency

1. .Efficient and secure path-quality monitoring is possible
   - ❑ Combining cryptography and sketching
   - ❑ Can monitor billions of packets using ~200 bytes of storage
   - ❑ No packet marking
   - ❑ Can use faster (and weaker) hash functions

2. PQM can be seen as an application of adversarial sketch model
   - ❑ And, sadly, sometimes subject to same lower bounds

Thanks!



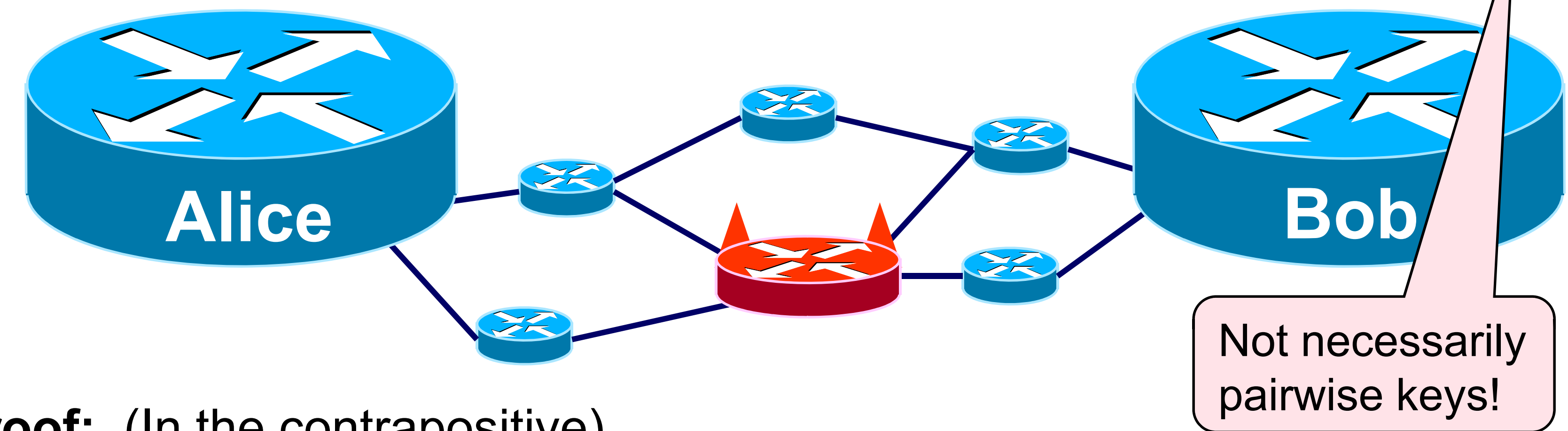
[Goldberg, Xiao, Tromer, Barak, Rexford, “Path-Quality Monitoring in the Presence of Adversaries”, to appear at SIGMETRICS 2008.]

www.princeton.edu/~goldbe

Secure PQM needs keys

Our protocol requires a key infrastructure between Alice and Bob.

Thm: **Any secure PQM protocol** that is robust adversaries on the path that can **add** and **drop** packets requires a key infrastructure.



__Proof:__ (In the contrapositive)

Assume Alice and Bob **do not** have a shared key

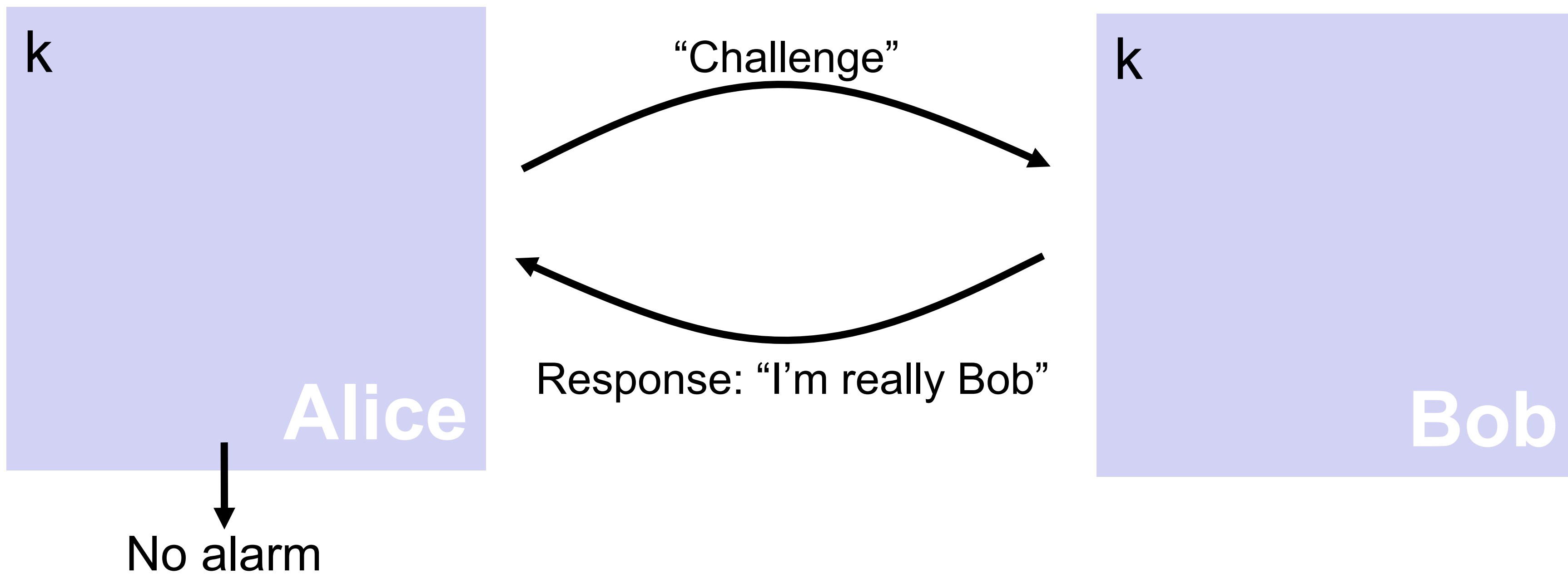
- All the packets that Alice sends to Bob pass thru Eve
- Then Eve knows everything Bob knows
- Eve drops all packets
- Eve impersonates Bob's reverse path messages (e.g. report)
- Alice won't detect packet loss, so Eve breaks security.

Secure PQM needs crypto (1)

Our protocol requires a key infrastructure between Alice and Bob.

Thm: **Any secure PQM protocol** that is robust adversaries on the path that can **add/drop** packets must invoke cryptographic operations.

<u>Proof:</u> (By **reduction** to keyed identification schemes (KIS))

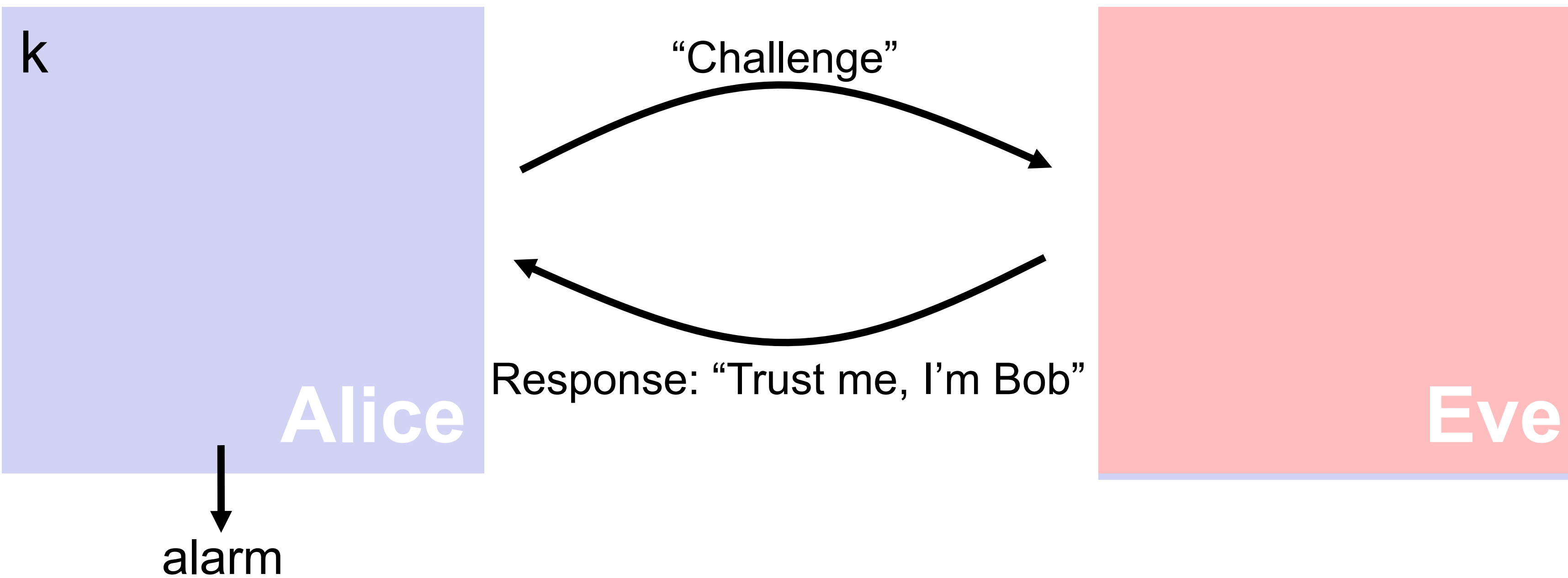


Secure PQM needs crypto (2)

Our protocol requires a key infrastructure between Alice and Bob.

Thm: **Any secure PQM protocol** that is robust adversaries on the path that can **add/drop** packets must invoke cryptographic operations.

Proof: (By **reduction** to keyed identification schemes (KIS))

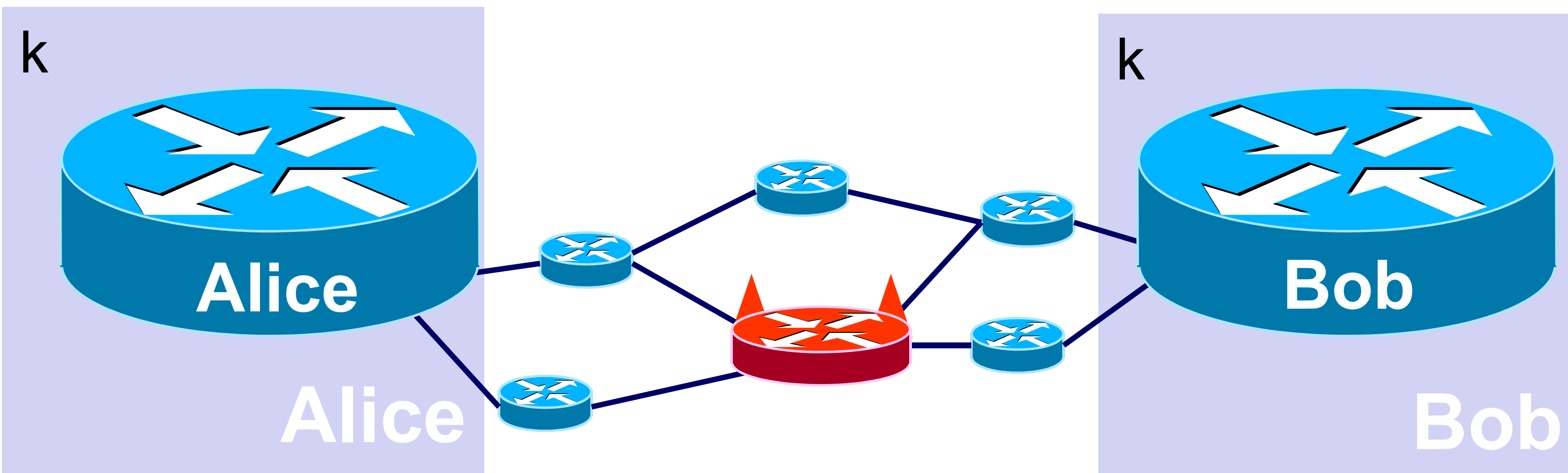


Secure PQM needs crypto (3)

Our protocol requires a key infrastructure between Alice and Bob.

Thm: **Any secure PQM protocol** that is robust adversaries on the path that can **add/drop** packets must invoke cryptographic operations.

<u>Proof:</u> (By **reduction** to keyed identification schemes (KIS))



Challenge: Traffic that Alice sends on the forward path

Response: Reverse path messages, *i.e.* report.

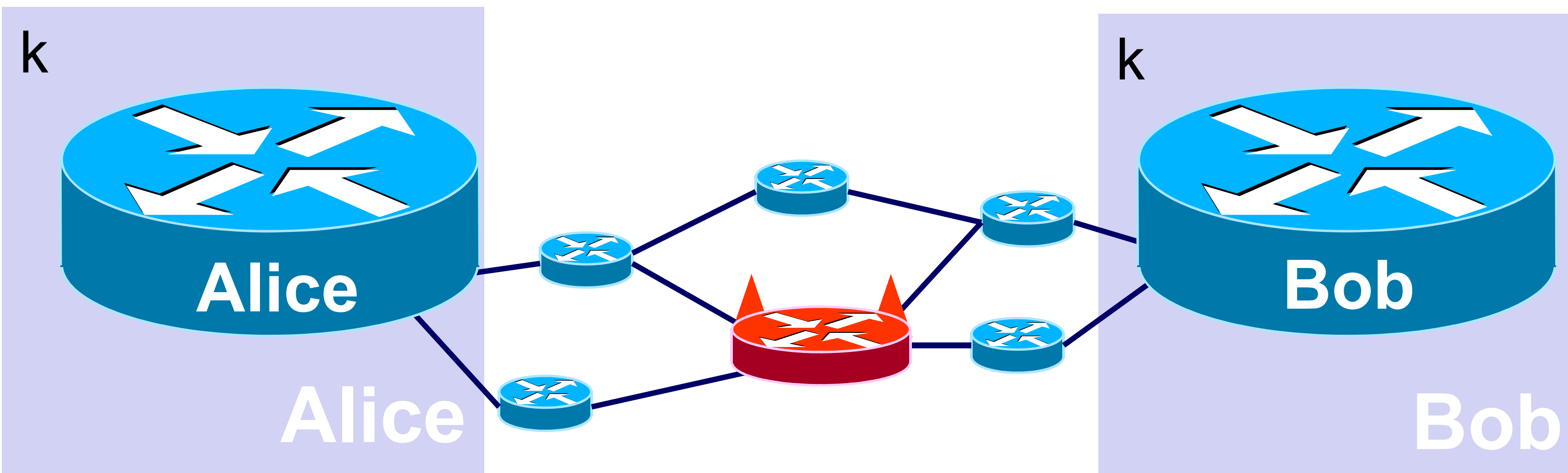
Alarm if report is invalid.

Secure PQM needs crypto (4)

Our protocol requires a key infrastructure between Alice and Bob.

Thm: **Any secure PQM protocol** that is robust adversaries on the path that can **add/drop** packets must invoke cryptographic operations.

Proof: (By **reduction** to keyed identification schemes (KIS))



KIS are at least as computationally complex as symmetric cryptographic primitives (e.g. encryption, MAC)

➔ Secure PQM needs crypto