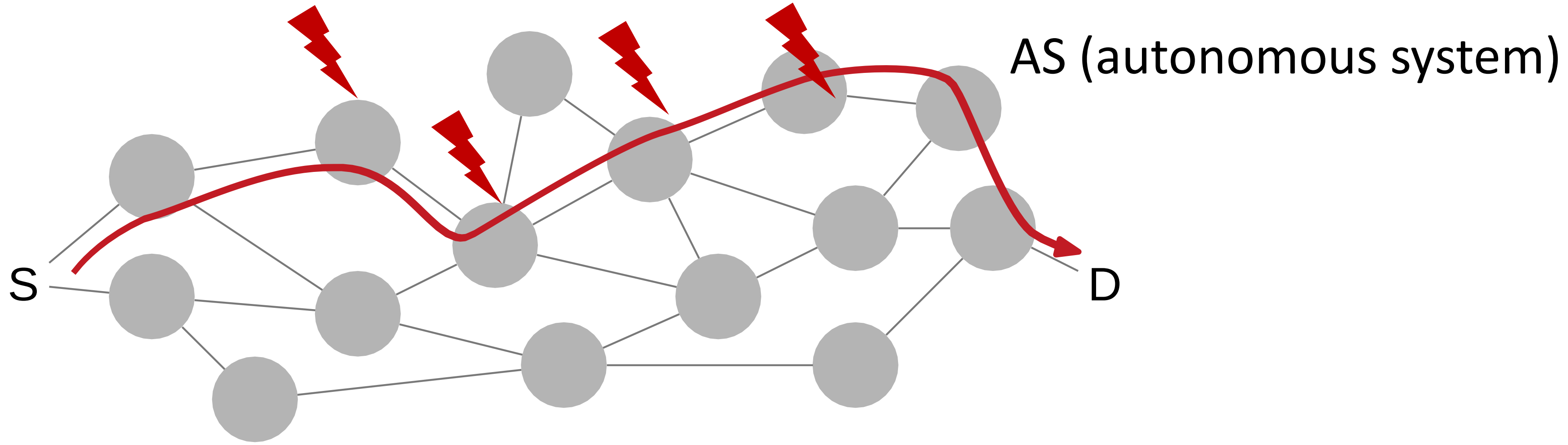
High-Speed Inter-Domain Fault Localization

Cristina Basescu, Yue-Hsun Lin, Haoming Zhang, Adrian Perrig

Presentation:
Huazhe Wang



Drop, delay, or modify packets:

- Malicious AS
- Configuration errors

Fault localization enables localization of the problem

- malicious entities attempt to hide and interfere with localization

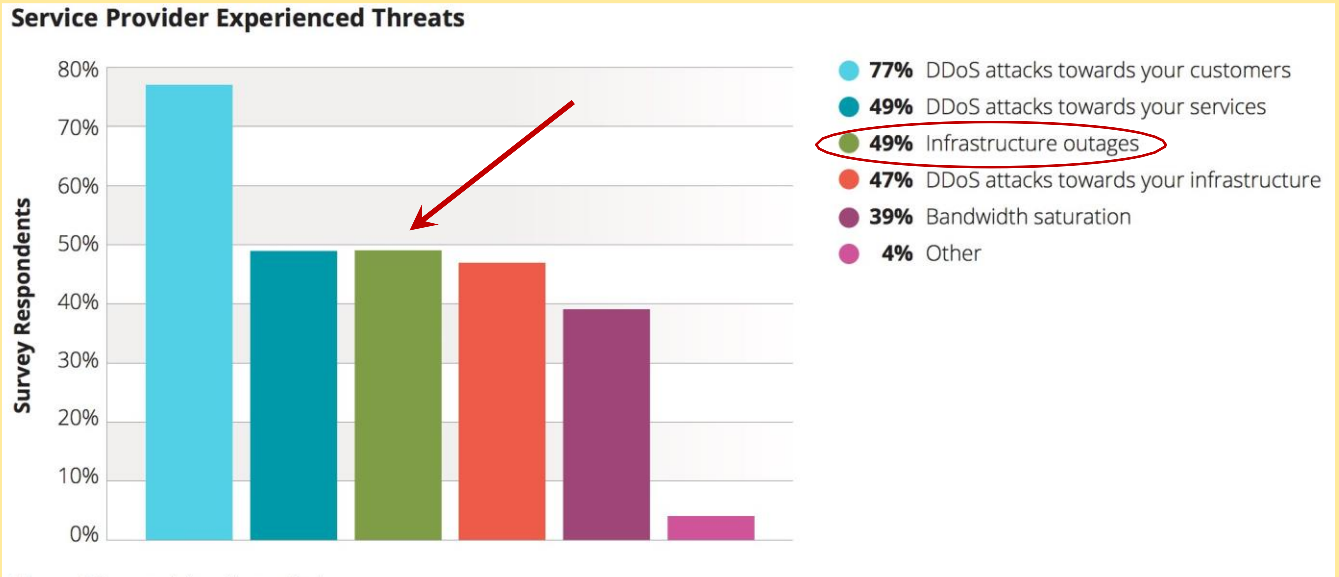
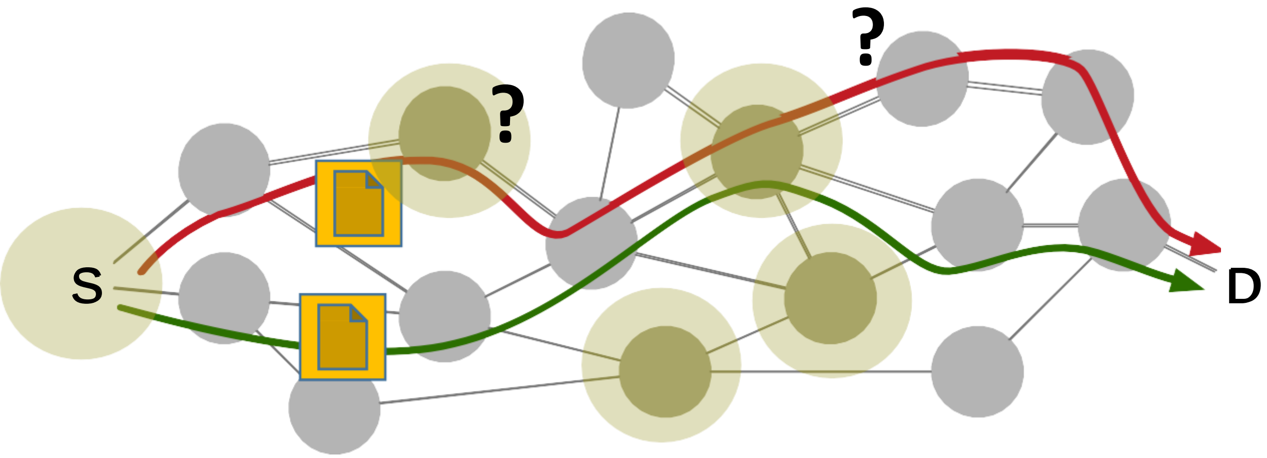


Figure 7 Source: Arbor Networks, Inc.

- **Fault localization problem statement**
  - Localize entities that **drop**, **delay**, or **modify traffic**
  - Practical for **inter-domain settings**



Who localizes faults?

Acceptable localization duration?

Acceptable communication overhead?

Storage overhead at nodes?

State of art:

❖ Each node stores a summary of observed packets and sends it to the source (path-based)
  - **per-source storage, share a key with each source**

❖ Sending summaries to fewer entities, such as an authenticated control
  - **Hard to deploy under inter domain setting**

Path based approaches: low comm. overhead, large memory cost due to per source or per flow storage

Neighborhood based approaches: low memory cost, but rely on trusted hardware or central entity

TABLE I: Comparison of the practicality of existing Fault Localization (FL) protocols.

FL scheme	Assumptions		Overhead		Practicality	
	No trusted central entity	No trusted hardware	Router storage per 10 Gbps link (FP: fast path, SP: slow path)	Comm. (extra %)	Max. eval. throughput	Localiz. delay for 99% accuracy (pkts)
Secure sketch FL	✓	✓	FP: 149.87*GB*[1]+*key* ∗ #*src* SP: *timer* ∗ #*slowpath_pkts*[2]	0.0002%	No eval	10^6
ShortMAC	✓	✓	FP: 21*B* ∗ #*flows*+*key* ∗ #*src* SP: *timer* ∗ #*slowpath_pkts*	0.01%	0.9Gbps	$3.8*10^4$
TrueNet	✓	X	FP: 512KB[3]+ *40B* ∗ #*neighbors*	0.0001%[4]	~1Gbps	10^4
DynaFL	X	✓	FP: *1.95MB*[5]∗ #*neighbors* + 1*key*	0.002% - 0.012%	No eval	$5*10^4$
Faultprints	✓	✓	FP: ~46.8MB SP: (*timer*+*ctrl_pkts*) ∗ #*slowpath_pkts*	3.3%	119.7Gbps	$4*10^3$

Adversary model

- An adversary can compromise any number of ASes.
- The ASes may drop, delay, modify or inject packets.
- The adversary cannot eavesdrop or influence traffic on links that are not adjacent to any of its routers.

Assumptions

- Source knows the entire AS-level path
- Router-level symmetric paths
- Loosely time synchronized nodes
- S and D share a symmetric key K_{SD}
- Each AS has a public-private key pair

Overview

Setup

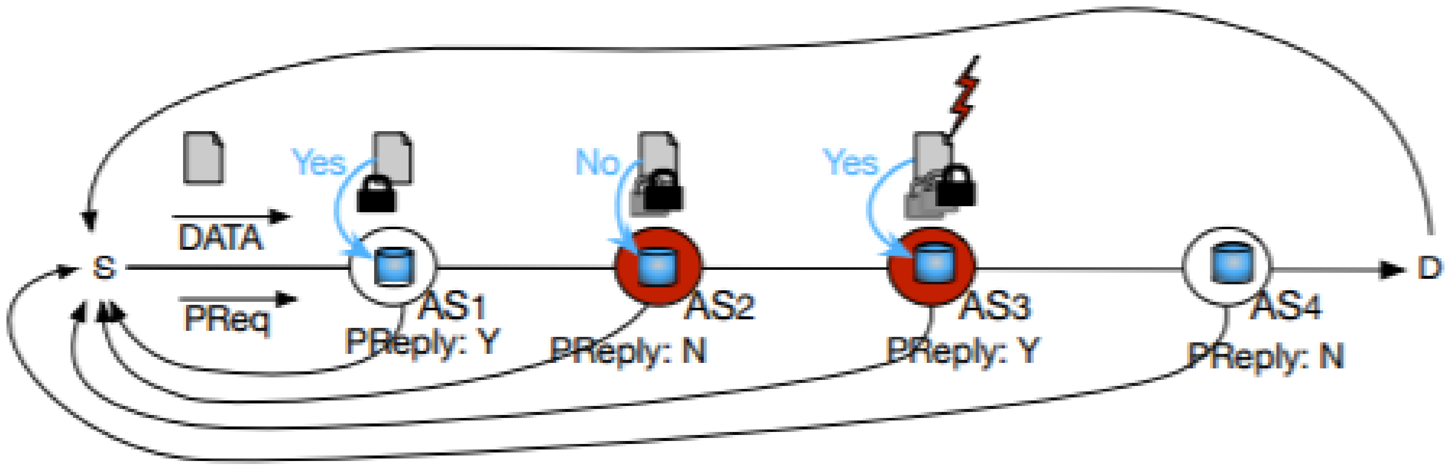
Each AS establishes with the source a secret key $K_{AS,S}$

DATA Sending

The source S sends our data. The destination D replies each packet with an ACK

Probing

S sends out probe request if an ACK is not received correctly. Each AS on the path reply to the source with a PReply message



Key setup

Source S, session σ, a public-private key pair $(PK_\sigma, PK_\sigma^{-1})$, cTimes

$$\textsc{SessionId} \leftarrow H(cTimes_S, PK_\sigma) \quad (1)$$

S generates a key setup packet (sessionID, cTimes, PK_σ)

At each AS, $$K_{AS_i,S} \leftarrow PRF_{SV_{AS_i}}(\textsc{SessionId}) \quad (2)$$

- Each AS derive a key on-the-fly based on a single secret value, so the internal node does not have to store per-host keys

Each AS replies S with

$$EncKEY_{AS_i,S} = Enc_{PK_\sigma}(K_{AS_i,S}) \quad (3)$$

$$SignKEY_{AS_i,S} = Sig_{AS_i}(EncKEY_{AS_i,S}, \textsc{SessionId}) \quad (4)$$

S learns the key without disclosing it to other entities.

Data sending at S

Source S inserts into the packet header sessionID, time, AS index,

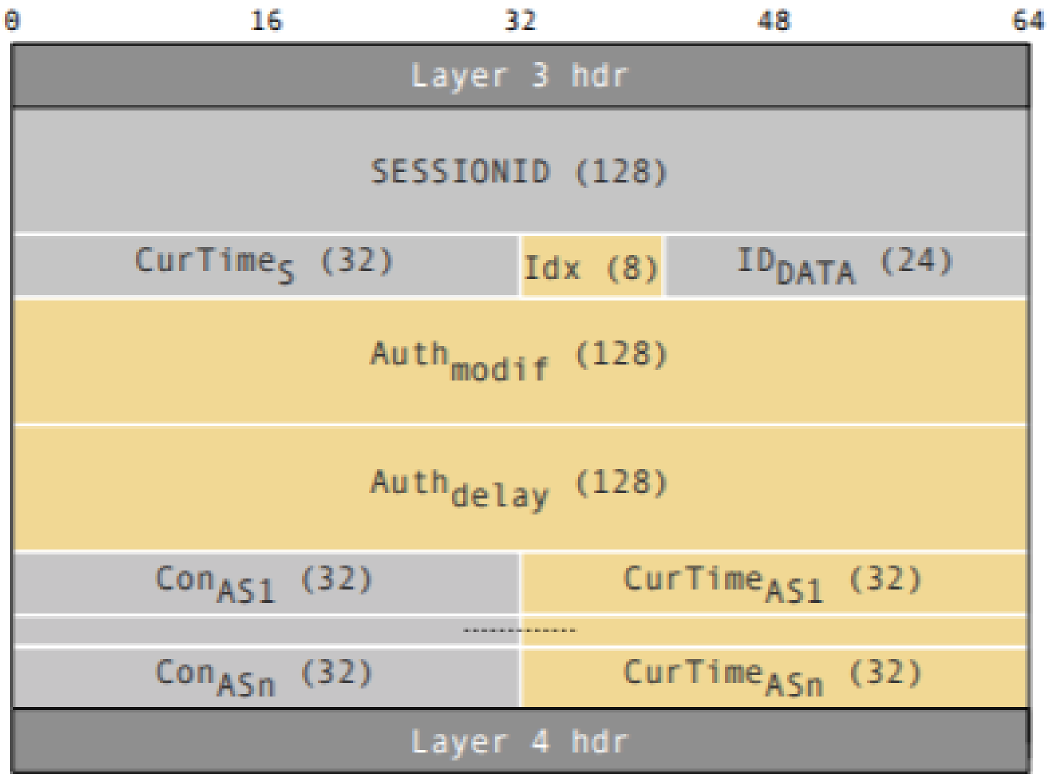
- Con_{ASi} : enable ASes to authenticate packets contents.

$$Con_1 \leftarrow MAC_{K_{AS_1,S}}(Cst(\text{DATA})),$$
$$Con_i \leftarrow MAC_{K_{AS_i,S}}(Cst(\text{DATA})||Con_{i-1}) \quad (5)$$

- ID_{DATA} : computed from DATA, used to match acknowledgements generated by D.

$$ID_{\text{DATA}} = MAC_{K_{SD}}(Cst(\text{DATA})) \quad (6)$$

- $Auth_{modif}$: to enable localization of problem.



Data sending at intermediate ASes

AS_i computes over the constant part of the packet using $K_{AS_i,S}$ and a pseudo-random function (PRF)

↓ If larger than P_{sample}

The packet is sampled and its fingerprint is stored in a local Bloom filter

⬇

The whole packet needs to be included in probing!

⬇

Sample and storage on a much smaller fingerprint $Auth_{modif}$

Data sending at intermediate ASes

AS_i computes fingerprint

$$Auth_{modif,0} \leftarrow ID_{\text{DATA}} || cTimes_S, \quad (7)$$
$$Auth_{modif,i} \leftarrow MAC_{K_{AS_i,S}}(Con_i || Auth_{modif,i-1})$$

$$PRF_{K_{AS_i,S}}(Auth_{modif,i}) \geq P_{Sample} \quad (8)$$

↓

Update $Auth_{modif}$

↓

Verify Con_i $$Con_i == MAC_{K_{AS_i,S}}(Cst(\text{DATA}) || Con_{i-1}) \quad (9)$$

↓ If true

Store the fingerprint in the Bloom filter

Similar operations with time delays.

Data sending at D

D computes ID_{DATA}

↓ If the value is correct

Create a D_{ACK} packet

$$Ack_{info} = ID_{\text{DATA}} || Auth_{delay} || cTime_{AS_1} || .. || cTime_{AS_n} \quad (11)$$

$$\text{DACK}[\text{DATA}] = Ack_{info} || MAC_{K_{SD}}(Ack_{info}) \quad (12)$$

To prevent the ACK is tampered by malicious nodes, D_{ACK} packets are also sampled on the reverse path.

Probing

The source decides with probability P_{probe} whether to probe an unacknowledged DATA packet and D_{ACK}.

S assembles a P_{REQ} packet:

$$\text{PREQ}[\text{DATA}] = \text{SESSIONID}||cTimes_S||IndexAS||Ctr|| \\ ||Con_1||..||Con_n||Auth_{modif}||ReplyTiming \quad (13)$$

↓

An AS derives the key, update $Auth_{modif}$,checks if the queried packet is sampled

↓ If sampled

ASes reply S separately with a bit indicating whether is queried packet is stored.

$$\text{PREPLY}_{AS_i}[\text{DATA}] = Enc_{K_{AS_i,S}}(bit_{Auth_{modif}})|| \\ ||MAC_{K_{AS_i,S}}(Enc_{K_{AS_i,S}}(bit_{Auth_{modif}})||\text{PREQ}[\text{DATA}]) \quad (14)$$

Probing

Reply packet indistinguishability:

- To prevent malicious ASes to launch framing attacks.
- Modified IP.

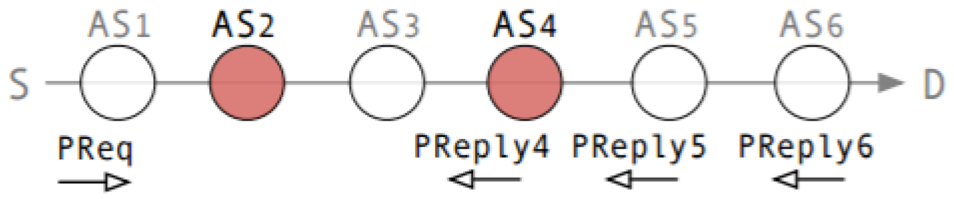


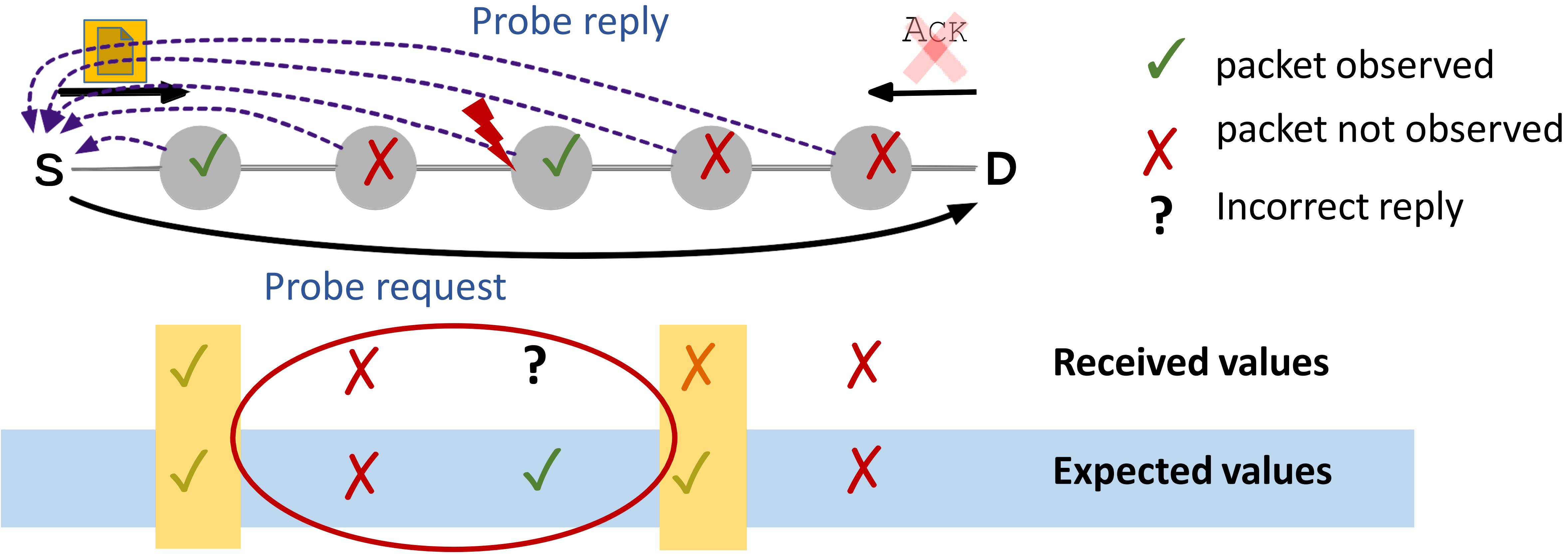
Fig. 4: Colluder nodes can track PREPLY packets, but targeted PREPLY damage is localized.

Delayed reply:

- Attackers could use the timing between P_{REQ} and P_{REPLY} to infer the number of hops from the AS that sent the reply.
- Relay time uniformly distributed from 100ms to 350 ms.

Fault localization

The source proceeds with localizing an adversarial AS only after it detects packet loss, unusual delay, or modification.



S compute corruption scores for correct replies. Compare corruption scores of AS neighbors can flag malicious links.

Fault localization

For incorrect replies (dropped, modified, delayed), S compute misbehavior probability for all ASes on the path.

S keeps tracking per epoch counters of damaged reply packets from each ASes on the forwarding path. At the end of the epoch, the source localizes as malicious the AS which maximizes the probability:

$$P(AS_i \; malicious | dmg_1, dmg_2, .., dmg_n) \overset{notation}{=} \mathscr{P}_i \qquad (16)$$

Fault localization

On the reverse path

$$B = \begin{array}{c} \\ \text{DMG} \\ \text{CORR} \end{array} \begin{array}{c} \begin{array}{cc} \text{DMG} & \text{CORR} \end{array} \\ \begin{pmatrix} 1 & 0 \\ \rho & 1-\rho \end{pmatrix} \end{array}, \quad D = \begin{array}{c} \\ \text{DMG} \\ \text{CORR} \end{array} \begin{array}{c} \begin{array}{cc} \text{DMG} & \text{CORR} \end{array} \\ \begin{pmatrix} 1 & 0 \\ P_D & 1-P_D \end{pmatrix} \end{array} \tag{17}$$

The probability of a correct reply packet to be damaged after traversing t Ases on the return path, out of which r are malicious.

$$P(t,r) = 1-(1-\rho)^{t-r}(1-P_D)^r \tag{18}$$

Both forward and reverse path

$$P(t,r,f) = 1-(1-\rho)^{t-r}(1-P_D)^r(1-P_Q)^f \tag{19}$$

Fault localization

$$\mathscr{P}_i = \frac{P(AS_i\ mal)}{P(dmg_1,\ldots,dmg_k)} * P(dmg_1,\ldots,dmg_k|AS_i\ mal) \quad (20)$$

$$P(dmg_1,\ldots,dmg_k|AS_i\ mal) = \prod_{j=1}^{k}[\binom{n}{dmg_j} P(t_j, r_{j,i}, f_{j,i})^{dmg_j} * \\ * (1 - P(t_j, r_{j,i}, f_{j,i})^{n-dmg_j})] \quad (21)$$

Simulation

Setup: forwarding path consists of 10 Ases, one malicious node at random location. Natural packet loss rate 0.001.

End-to-end maximum corruption rate

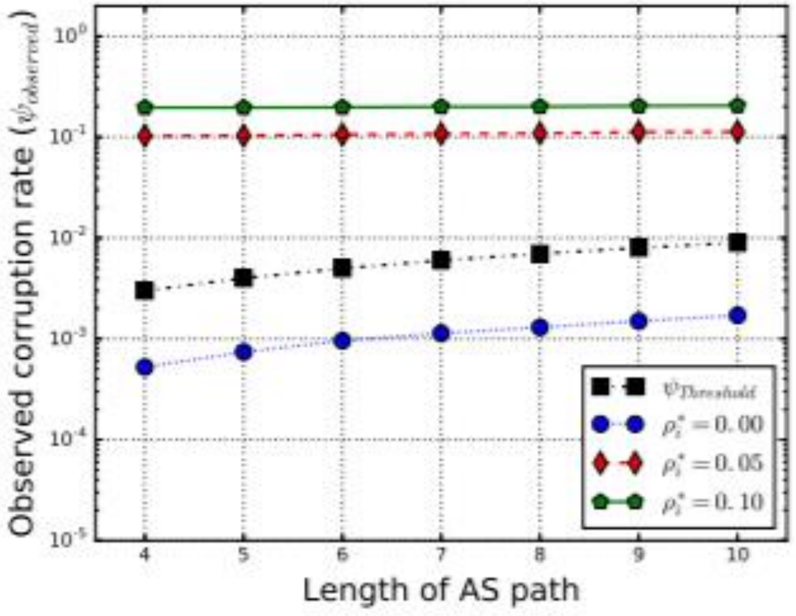


Fig. 10: Theoretical bound rate $\psi_{threshold}$ and observed rate $\psi_{observed}$ for varying malicious link corruption rates ρ_i^* and path lengths.

Path with adversaries with higher corruption rate always results in higher e2e corruption rate.

Simulation

Localization accuracy

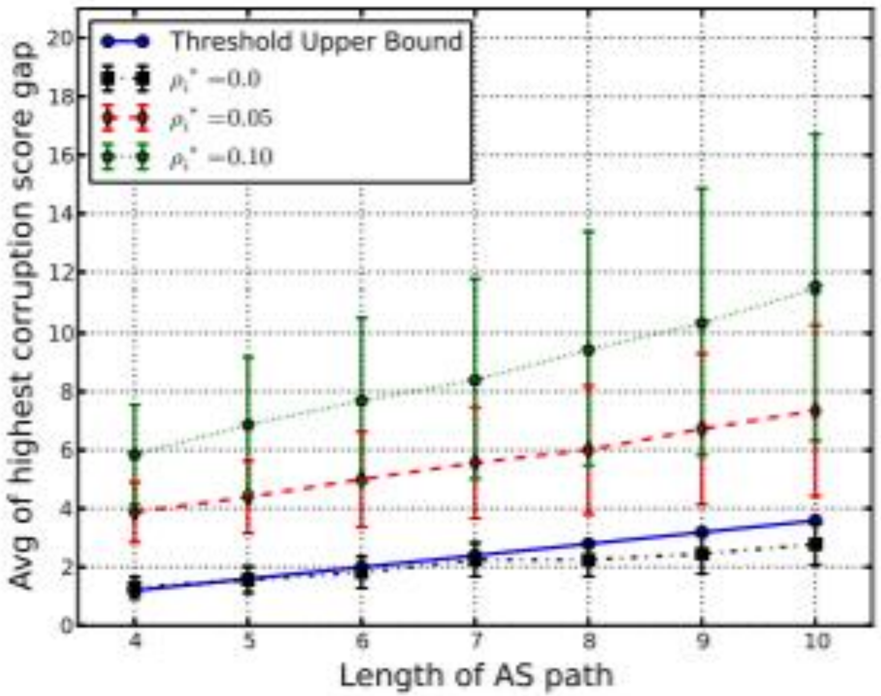


Fig. 11: Average and deviation of highest corruption score gaps computed by source, for varying malicious link corruption rates ρ_i^* and varying path lengths. AS parameters are P_{Probe} = 0.1 and FP_{Bf} = 0.02.

As path length increases, the source still correctly identifies adversarial activity.

Simulation

Localization accuracy

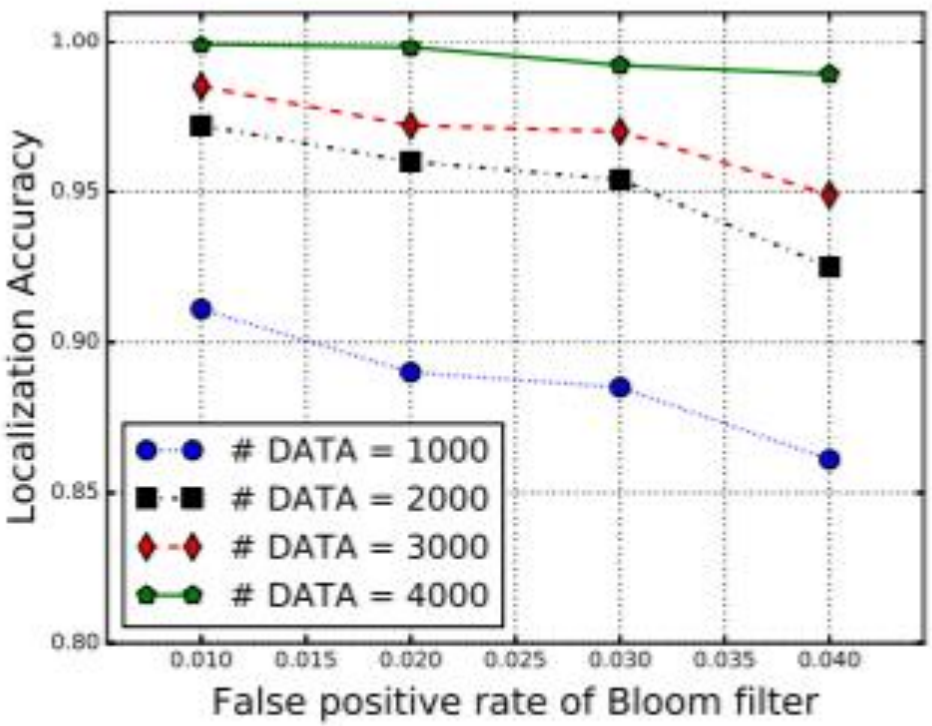


Fig. 12: Localization accuracy of corruption scores, with varying sending rates of DATA packets and false positive rate of Bloom filter.

Simulation

Localization accuracy

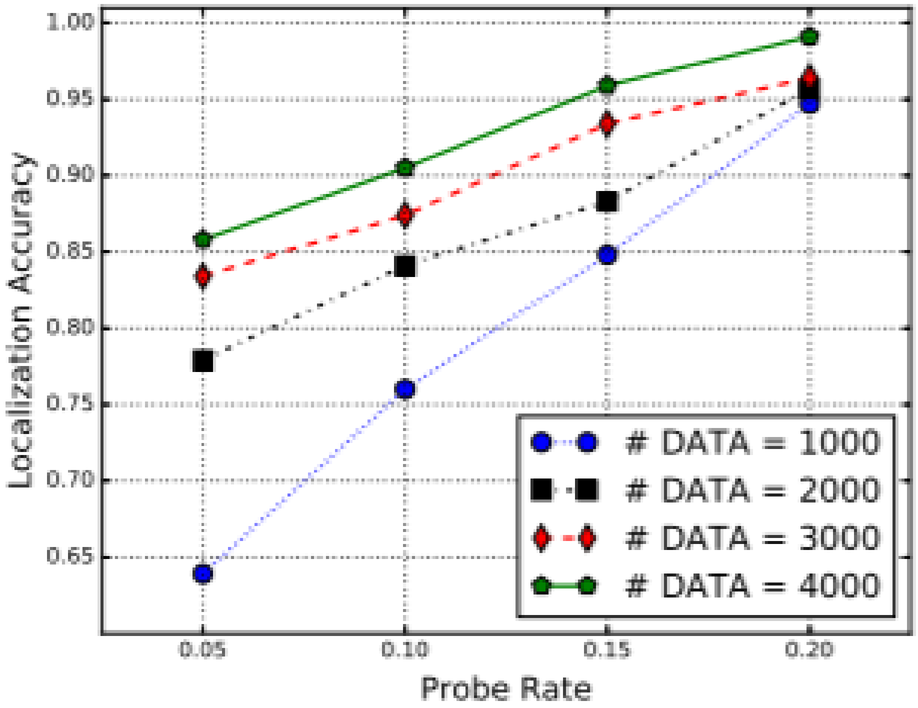


Fig. 13: Localization accuracy of misbehavior probabilities, with varying sending rates of DATA packets and probe rate P_{Probe}.

Works better when the source either sends enough data packets or perform more aggressive probing.

Simulation

Probing overhead

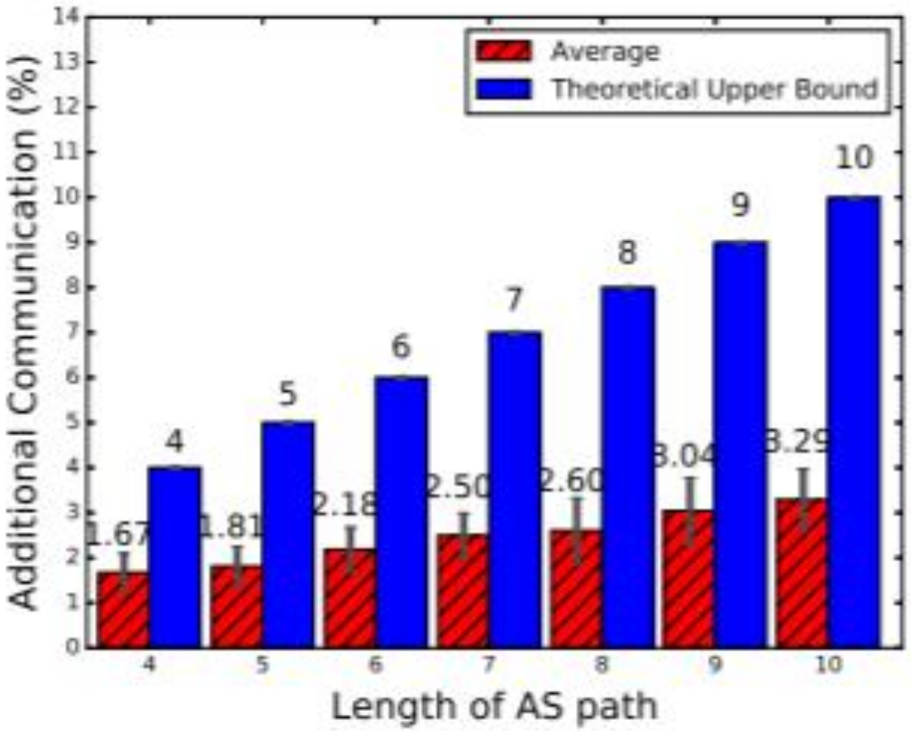
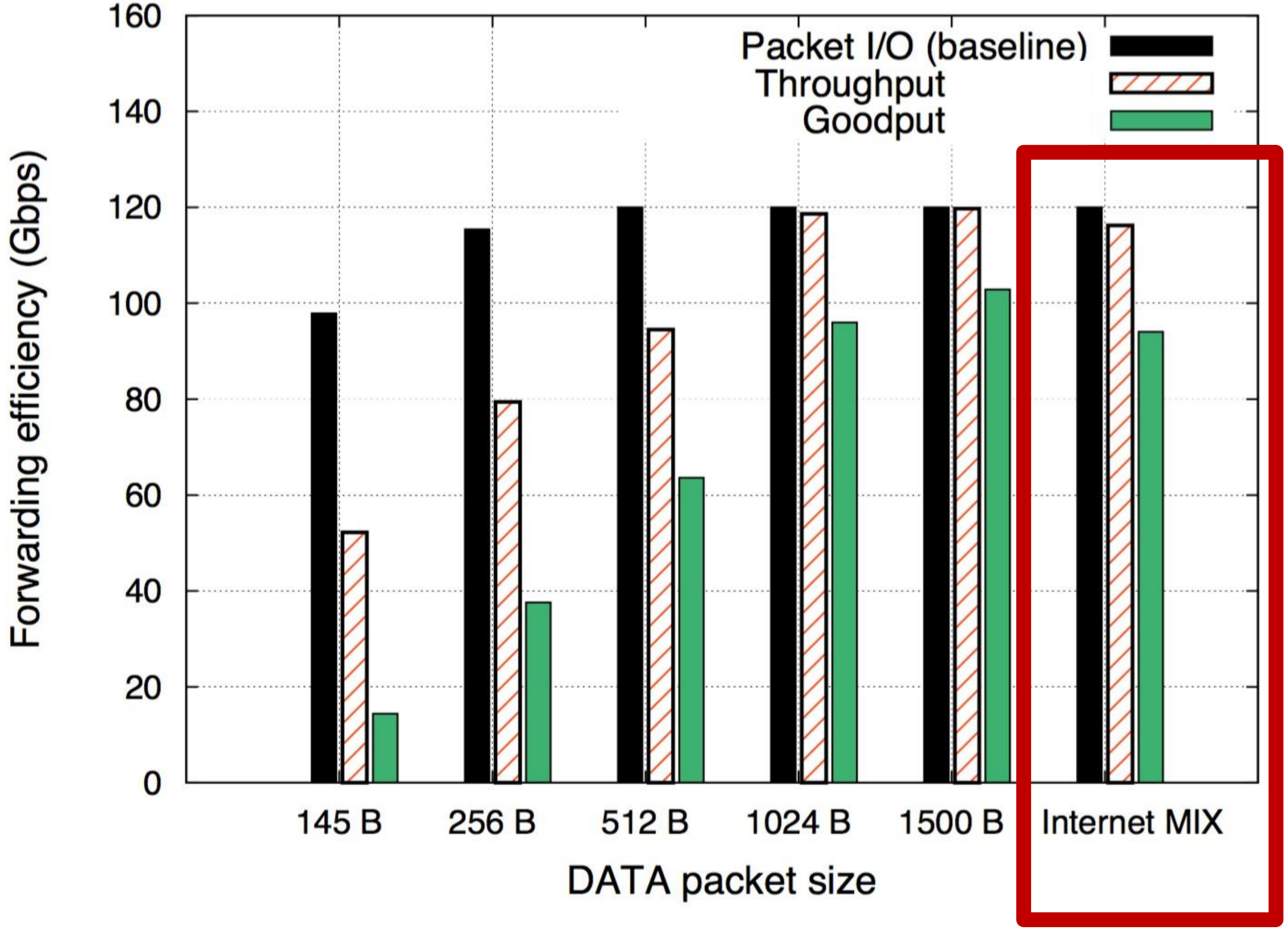


Fig. 14: Communication overhead along various path lengths: theoretical upper bound in plain colors, and average case in pattern colors.

Throughput and Goodput

- **Commodity server** as Faultprints router receiving traffic at 120 Gbps



- Sampling rate 10%
- Bloom filter false positive rate 0.02
- Path length 5 ASes

Conclusion

- **Faultprints localizes Internet-wide packet drop, delay, and modification**
- **Low storage requirements: ~46 MB for 10 Gbps traffic rate**
- **Secure against storage exhaustion attacks and framing attacks**
- **Real-world traffic forwarded on commodity server at ~117 / 120 Gbps**