UNIVERSITY OF SAN FRANCISCO

Locating Link-Layer Compression Links on the Internet

Vahab Pournaghshband (vahab.p@usfca.edu)
Xiuhui Wang (xwang247@dons.usfca.edu)

Network and Security Research Laboratory
University of San Francisco

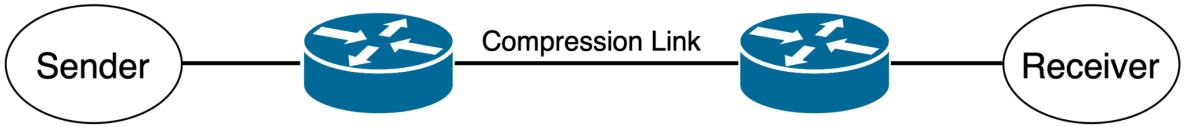
IARIA

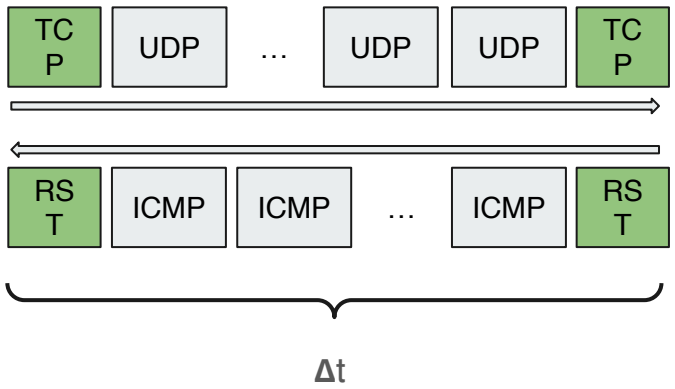
IARIA Congress 2024

Introduction

- Internet routers or nodes sometimes compress traffic flows at the link-layer or IP-level without knowledge of the end-hosts.
- When end-host applications are aware of intermediary compression, they can save time and resources by not applying compression themselves.
- We introduce a probing technique to **detect compression** of traffic flows and determine the **distance of the compression link** from the sender, if it exists.
- This technique is non-intrusive and does not require changes to or information from intermediate nodes.
- Detection relies solely on packet inter-arrival times and does not require receiver cooperation.
- The effectiveness of this technique is validated in a simulated environment using ns-3.

Detection Methodology



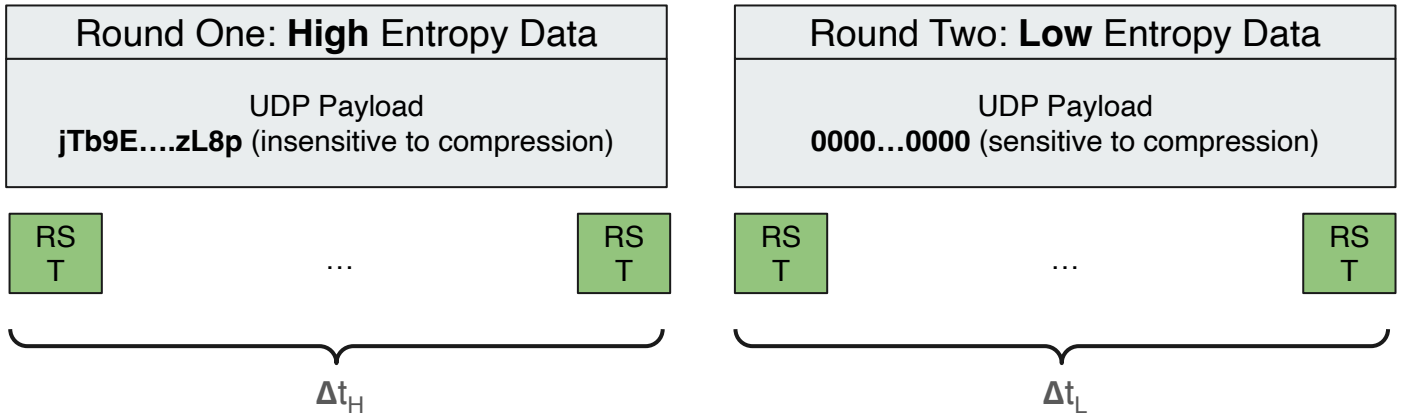


One Round

- In one round, sender transmits a single head SYN packet, followed by a train of n UDP packets, and then a tail SYN packet.
- Subsequently, the sender receives a head RST packet, a series of ICMP packets, and a tail RST packet.
- For each round, we measure the time difference between the two RST packets as **Δ**t.

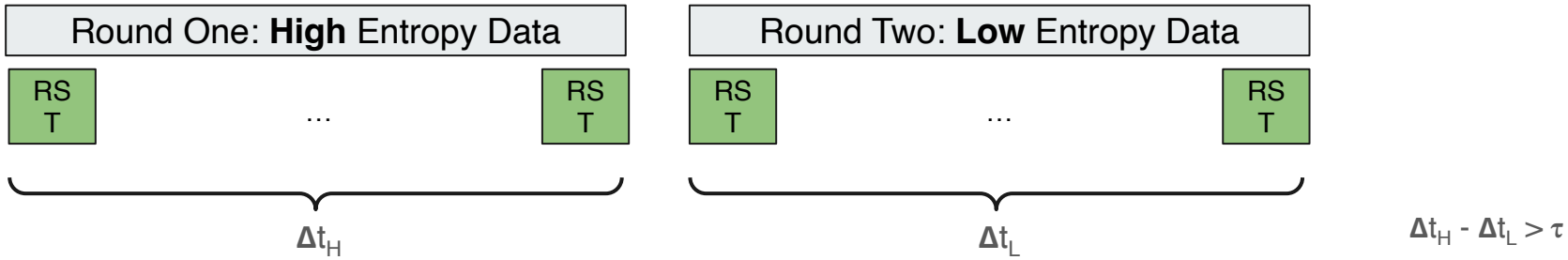
Detection Methodology

- Round 1: Send packets containing high-entropy data (e.g., random characters).
- Round 2: Repeat round 1, but with identical-sized packets containing low-entropy data (e.g., zeros).
- Measure the difference in arrival times between the first and last RST packet for both data types.
- Compressed low-entropy data packets are expected to be smaller than high-entropy packets, resulting in shorter transmission delays.

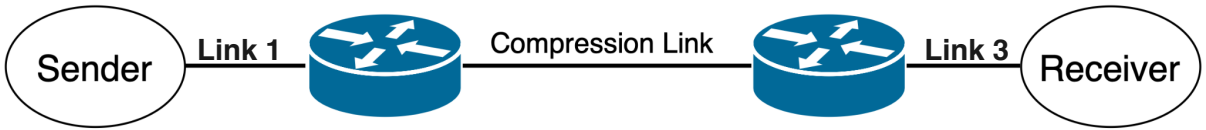


Detection Methodology

- If Δt_H - Δt_L is greater than a threshold τ, then compression is likely occurring.
- The threshold τ is used to eliminate normal Internet variability and is empirically set to 100 msec.
- The value of the threshold depends on the time precision and resolution of the machine performing the measurement.
- Our approach relies on RST responses instead of ICMP, as ICMP responses are not always generated, or they often are sent to the router's slow path or rate-limited.

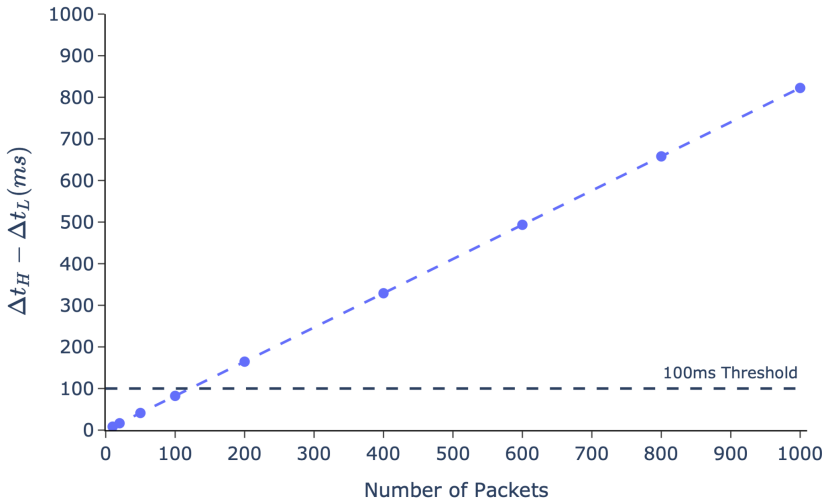


Parameters



- We tested the proposed approach using the topology above.
- Link 1 and link 3 are set to 100 Mbps. We assumed the compression is applied on the bottleneck link, which is a widely accepted network practice
- **Compression link capacity** is tested with discrete values from 5Mbps to 100Mbps.
- The **number of UDP packets** in the train is tested with discrete values from 10 to 1000.
- Each UDP **packet size** is set to 1100 Byte.
- The detection **threshold** is set to 100 ms.

Validation Results

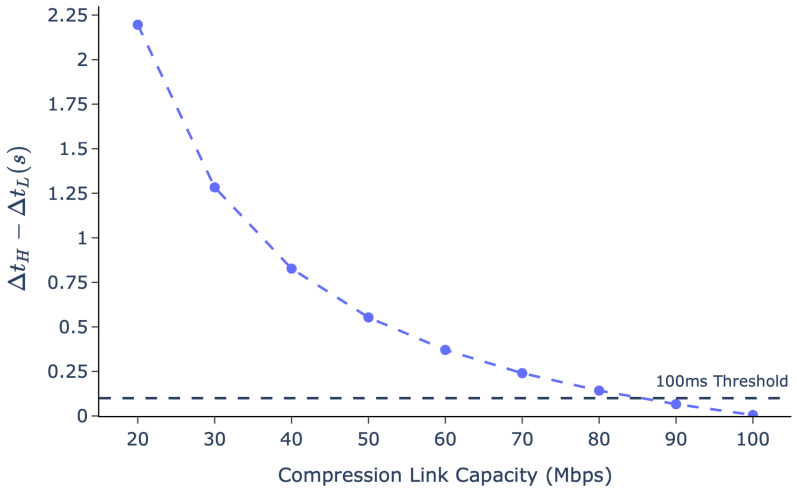


As the packet number of the UDP packet train increases, a linear increase in Δt_H - Δt_L is observed in the presence of a compression link.
For trains exceeding a length slightly over 110, a significant time difference exceeding the threshold is consistently observed, indicating the presence of compression.

The more packets, the easier to detect compression.

Compression link capacity = 10 Mbps
Packet Size = 1100 B
Number of Packets = 10, 20, 50, …, 1000

Validation Results



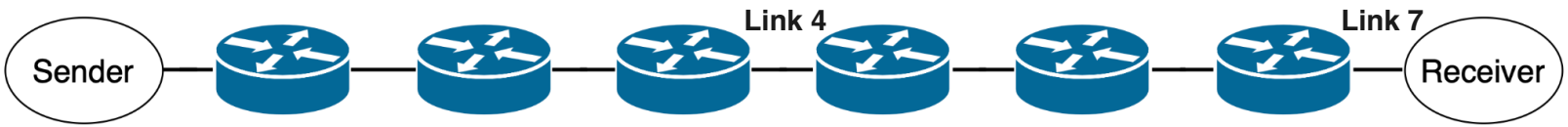
As the compression (bottleneck) link capacity increases, the Δt_H - Δt_L will rapidly decrease and gradually approach 0, meaning that we won't see significant differences in transmission time between high entropy data and low entropy data.

This implies that if the link where the compression intermediary resides is narrower, we will be more likely to detect it.

Compression link capacity = 20, 30, …, 100 Mbps
Outer link capacity = 100 Mbps
Packet Size = 1100 B
Number of Packets = 6000

* For compression link capacity smaller than 20, the difference is even larger

Locating Methodology



By increasing the TTL of UDP packets, we are able to locate a compression link in a network.

- Gradually increase the TTL value in the UDP header starting from 1.
- When the TTL is not enough to reach the compression link, neither high entropy nor low entropy packets will be compressed, resulting in a very small $\Delta t_H - \Delta t_L$.
- When the TTL is large enough, $\Delta t_H - \Delta t_L$ will change significantly, at which point the distance of the compression link from the sender can be determined.

Verification:

- Build an 8-node link and verify the following three cases: no compression link, compression link at link 4, and compression link at link 7.
- For each case, increase the TTL from 1 to 7 gradually and record the change of the $\Delta t_H - \Delta t_L$.

Validation Results - Locating the compression Link

Units: second

	TTL	Δt_H	Δt_L	$\Delta t_H - \Delta t_L$
Compression on 4th Link	3	0.543	0.543	0
	4	**5.477**	0.543	4.934
	5	**5.477**	0.543	4.934
Compression on 7th Link	5	0.543	0.543	0
	6	0.543	0.543	0
	7	**5.477**	0.543	4.934
No Compression	1	0.542	0.542	0
	4	0.543	0.543	0

When compression locates at link 4, the difference is 0 when TTL ≤ 3.

All TTLs > 4 allow packets to be compressed, and the difference shows that the transmission delay of low entropy data is significantly smaller than that of high entropy data.

Same effect was observed when compression was placed on link 7.

For no compression case, no difference is detected between Δt_H and Δt_L for all TTL values.

Conclusion

- This work explores the feasibility of detecting compression performed by intermediaries on the network path.
- We introduced an end-to-end packet-train-like method that functions in uncooperative but responsive environments.
- We also provided an approach utilizing TTL to locate the compression link.
- Future work includes developing a practical tool with minimal measurements (ideally one) for detection. This tool should be lightweight for mobile applications with limited resources.