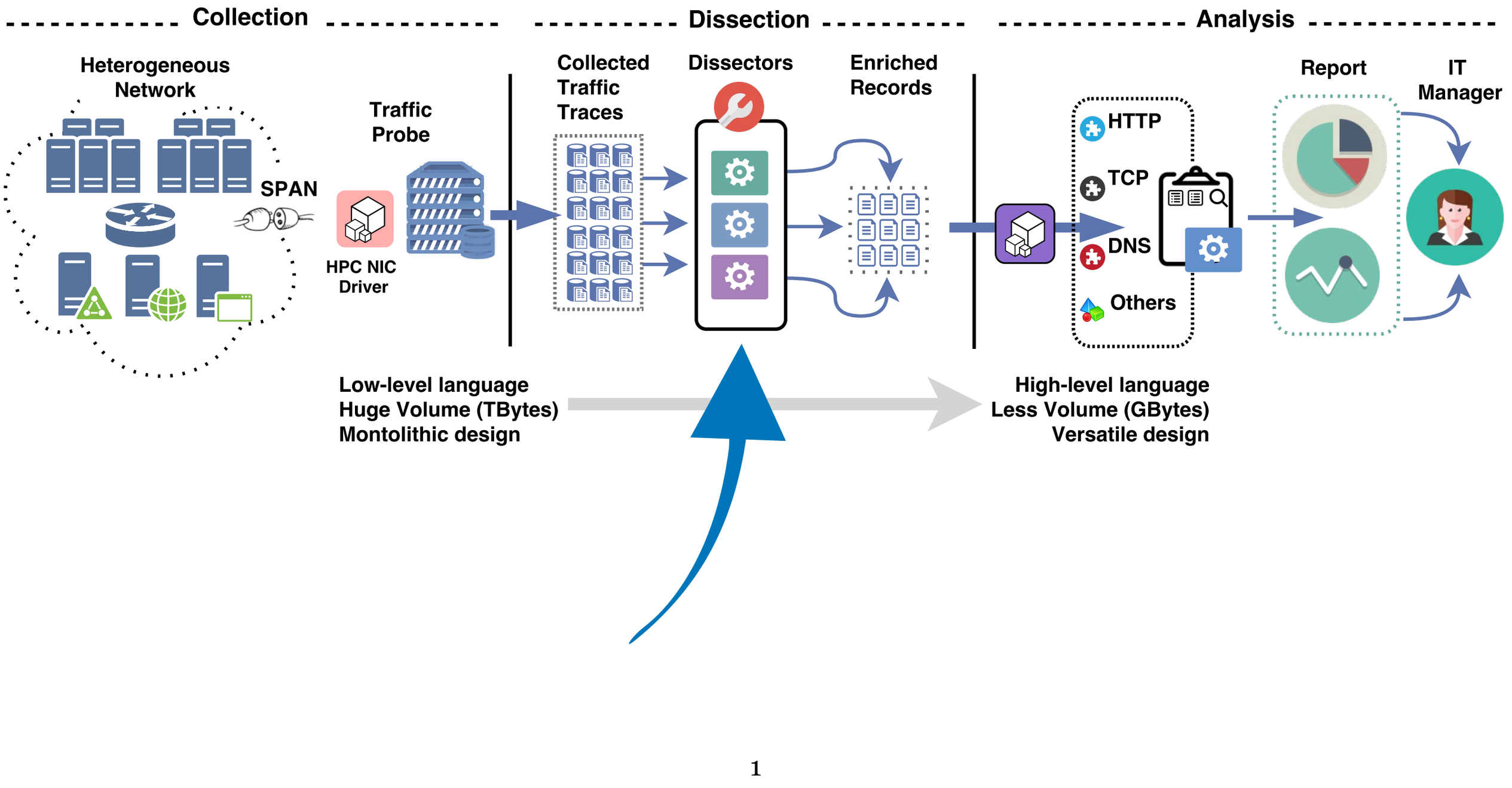
MULTI-GBPS HTTP TRAFFIC ANALYSIS IN COMMODITY HARDWARE BASED ON LOCAL KNOWLEDGE OF TCP STREAMS

Carlos Vega Moreno

Introduction



Introduction

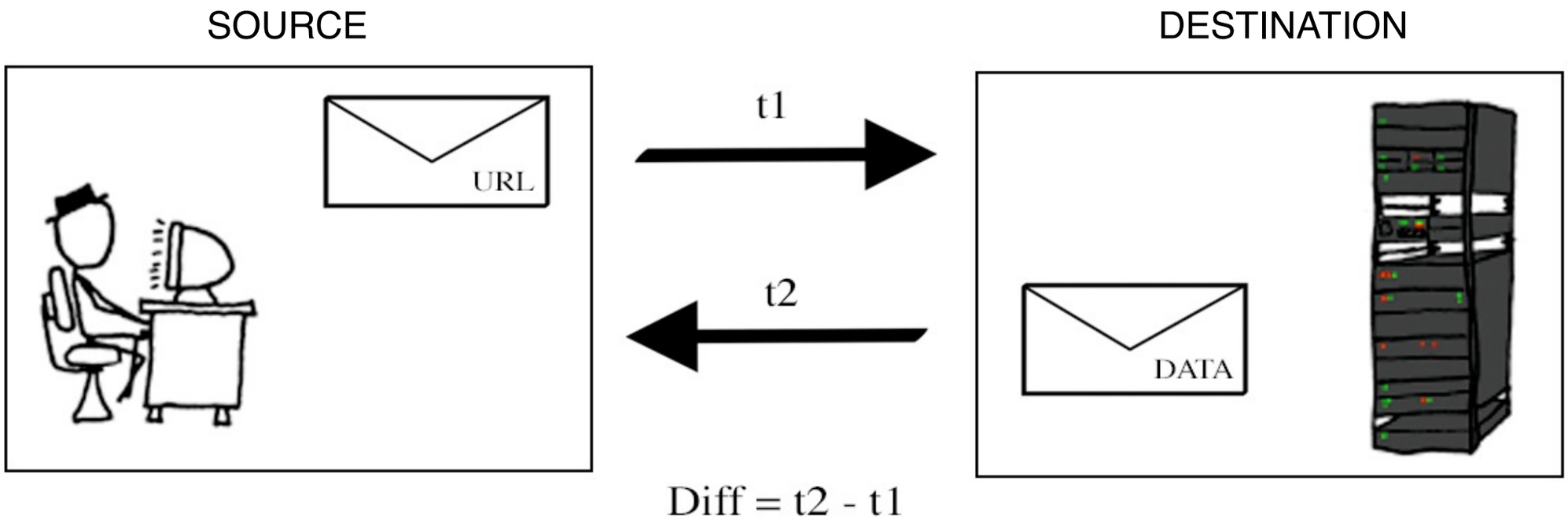
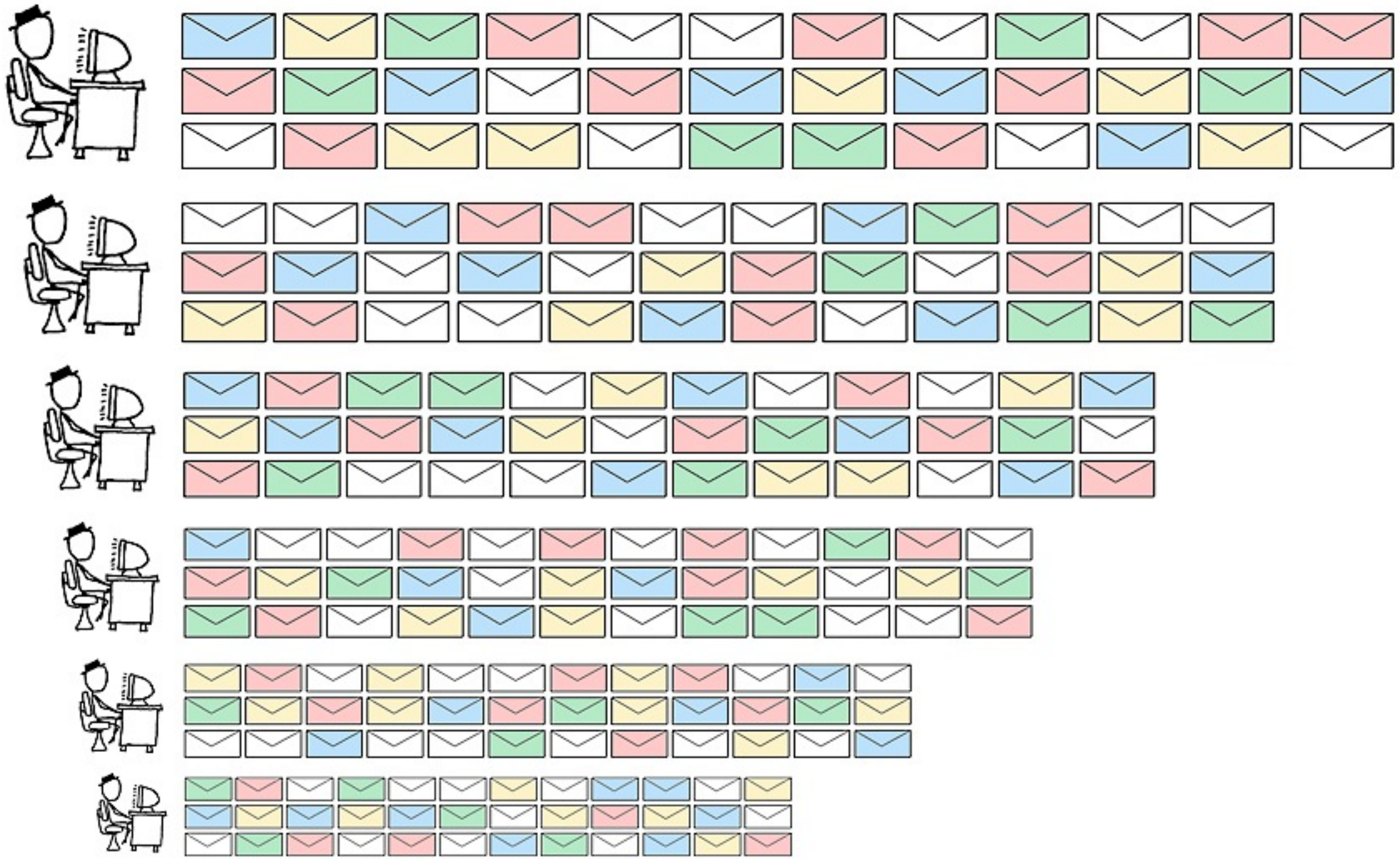


image credits: xkcd.com

Introduction



• • •

image credits: xkcd.com

Introduction

- State of the Art
  - Jin Xu et alii:
    - IP and TCP connection reassembly
    - Specific Hardware: Tilera many-core 64 cores.
    - Up to 2 Gbps
  - Zhang et alii: Solution for Intel achieving 20Gbps using 5 cores
    - 5 general purpose cores.
    - 20 Gbps in tests using samples with 2 million packets.

Requirements

- Developing a commodity hardware solution.
- Achieve 10 Gbps rates per core.
- Improving the matching speed of request and responses.
- Improving the load balance techniques for higher speeds.
- Evaluation in real scenarios and enterprise traffic.

Solution

- Achieve Higher Performance
  - Avoid the reassembly of the underlying TCP connection, mathing the first packet of the HTTP request and the first packet of the HTTP response, disregarding the rest of the connection.

Solution

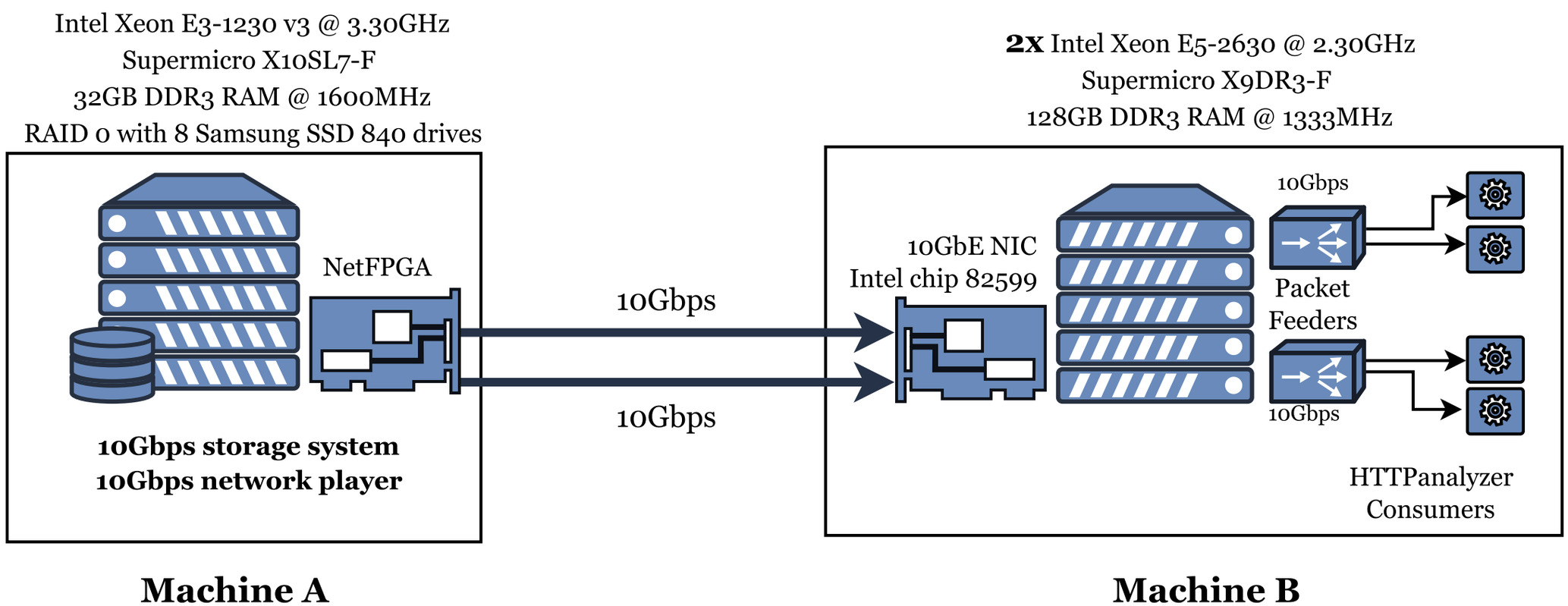
- Improve Load Balance
  - Instead of the traditional approach of distributing packets based on a connection basis, we propose a way to distribute them based on transactions.
  - This avoids heavy hitter issues since it distributes the packets at transaction level instead of connection level.

$$Hash\ Value = \text{Src}.\ IP \oplus Src.\ Port \oplus Dst.\ IP \oplus Dst.\ Port$$

$$Consumer = \begin{cases} Request: & Src.\ IP \oplus Src.\ Port \oplus Dst.\ IP \oplus Dst.\ Port \oplus ACK \oplus (Ack_1 \oplus Ack_2 \oplus Ack_3 \oplus Ack_4) \\ Response: & Src.\ IP \oplus Src.\ Port \oplus Dst.\ IP \oplus Dst.\ Port \oplus SEQ \oplus (Seq_1 \oplus Seq_2 \oplus Seq_3 \oplus Seq_4) \end{cases} mod.\ n$$

Results

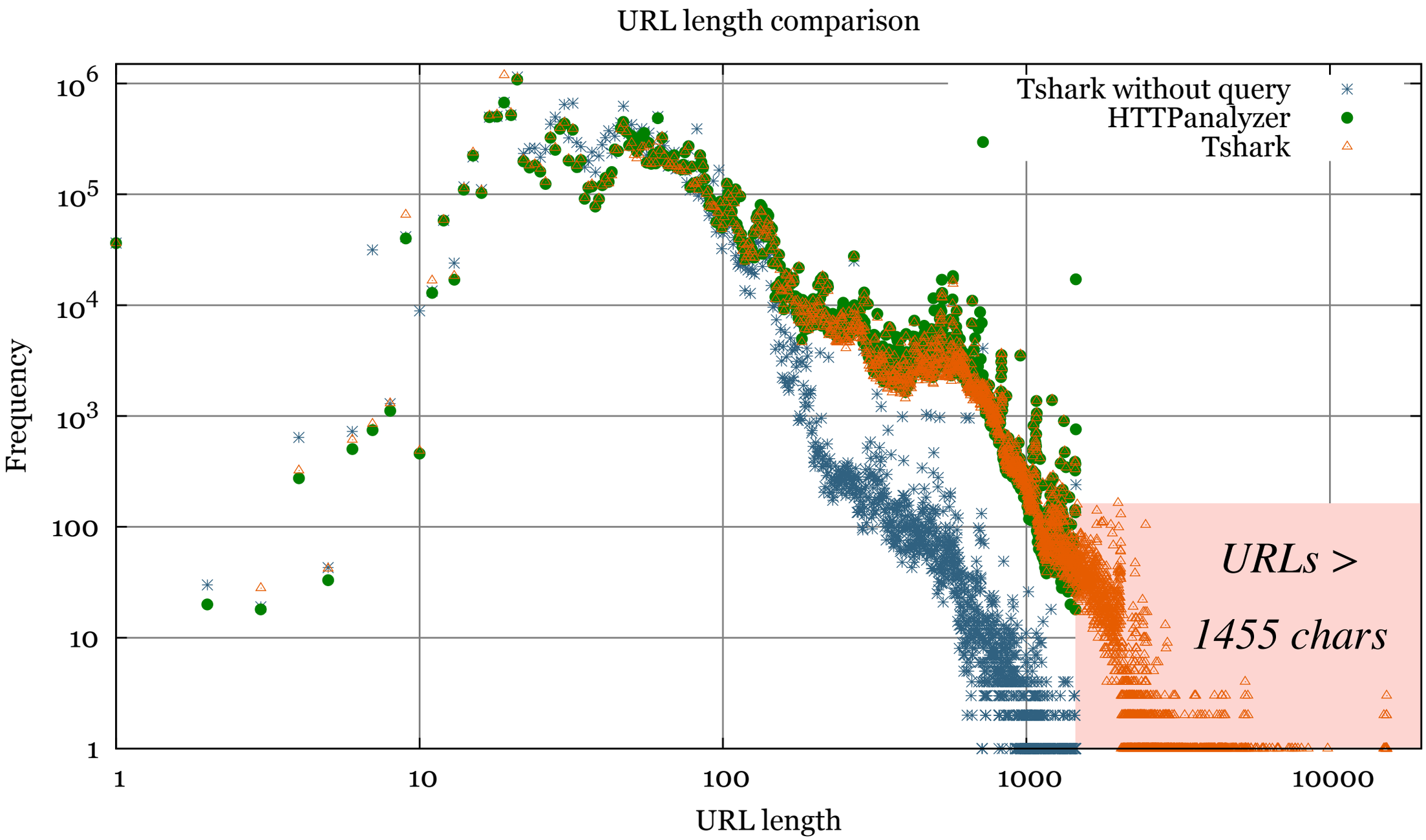
Scenarios



- 500 GB of test data, with more than 700 millon packets and 16 millon HTTP transactions.
- Obtained performance: 10~13 Gbps with a single core Intel Xeon.
- 20 Gbps and more with an efficient method for load balancing.

Results

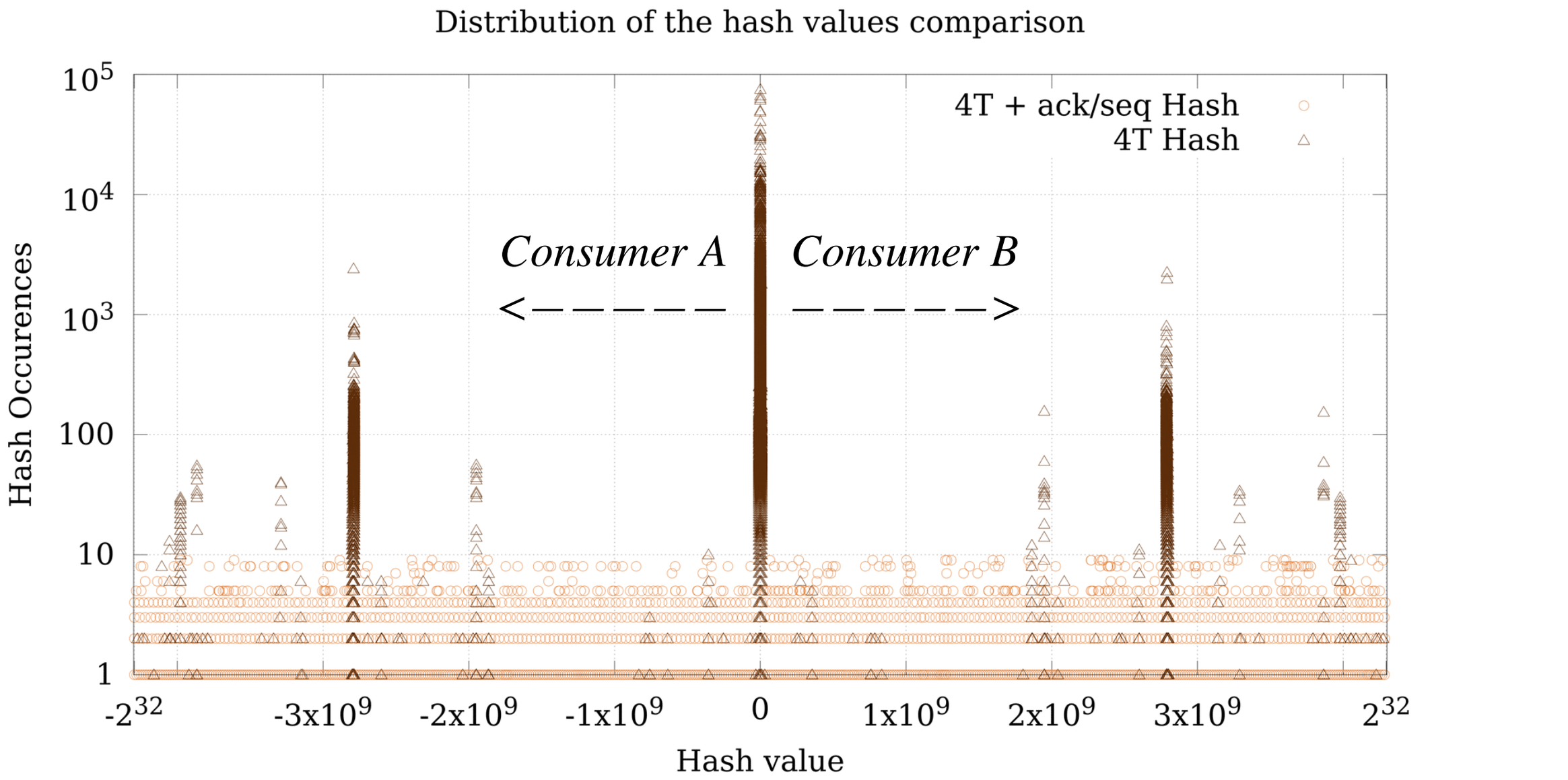
Limitations



500 GB of data, with ≈700 millon packets and 16 millon HTTP transactions

Results

Load balance

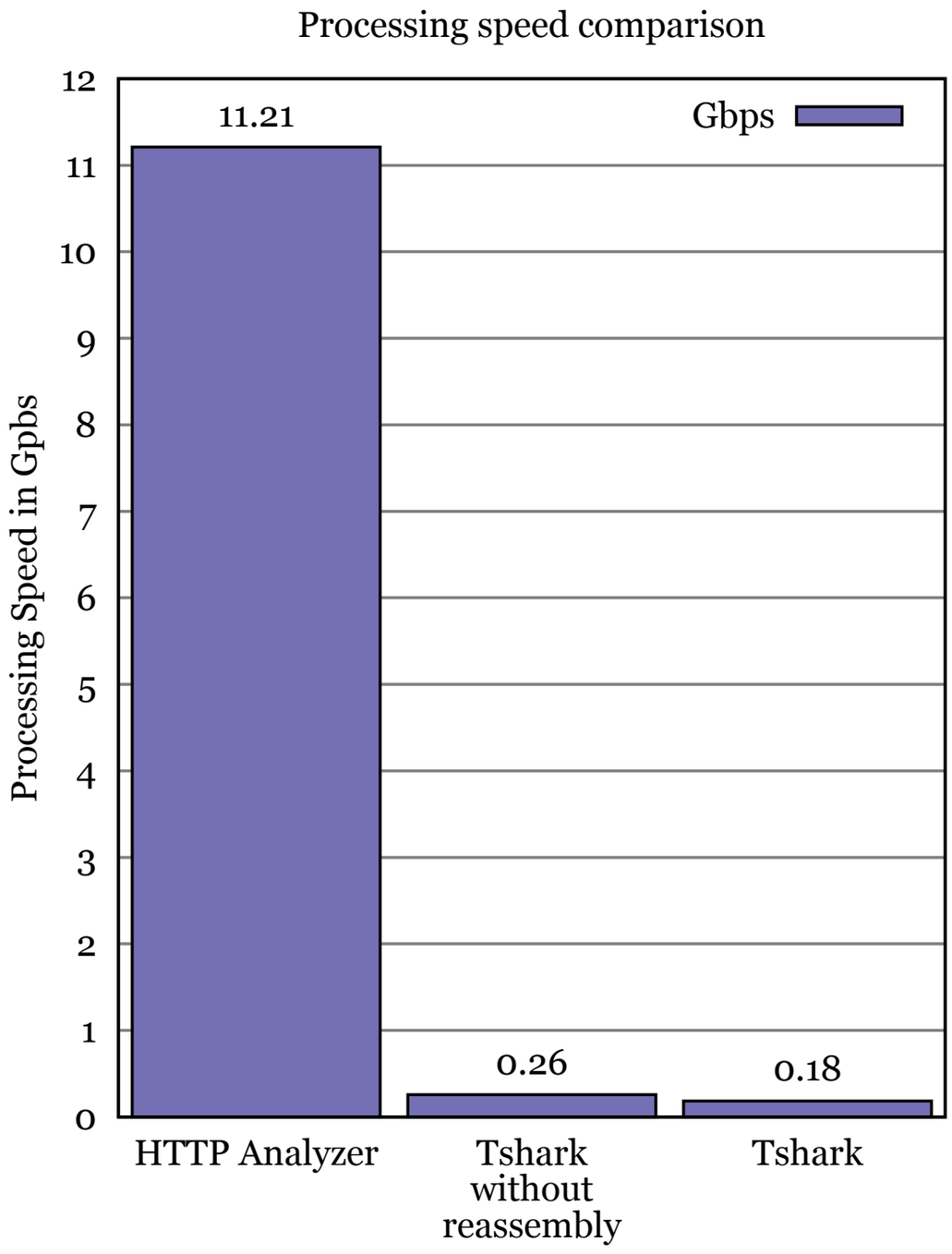


$$Hash\ Value = \text{Src.}\ IP \oplus Src.\ Port \oplus Dst.\ IP \oplus Dst.\ Port$$

$$Consumer = \begin{cases} Request: & Src.\ IP \oplus Src.\ Port \oplus Dst.\ IP \oplus Dst.\ Port \oplus ACK \oplus (Ack_1 \oplus Ack_2 \oplus Ack_3 \oplus Ack_4) \\ Response: & Src.\ IP \oplus Src.\ Port \oplus Dst.\ IP \oplus Dst.\ Port \oplus SEQ \oplus (Seq_1 \oplus Seq_2 \oplus Seq_3 \oplus Seq_4) \end{cases} mod.\ n$$

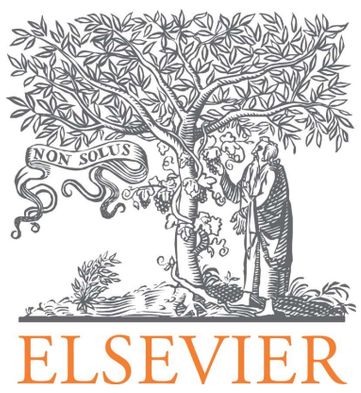
Results

Performance



Multi-Gbps HTTP Traffic Analysis in Commodity Hardware Based on Local Knowledge of TCP Streams

- This work is been published in **Computer Networks** in 2017
  - `https://doi.org/10.1016/j.comnet.2017.01.001`
- Is also available at **Arxiv**
  - `https://arxiv.org/abs/1701.04617`
- The code is also available for free and further research at **Github**
  - `https://github.com/carlosvega/httpDissector`



What's next?

- Find new HPC methods for HTTPS
- Face new protocols such as QUIC and HTTP 2
- i.e. Use the logs for correlation of traffic events (e.g. 0 Win) and Application Errors.

Questions


UAM
High Performance
Computing and
Networking


naudit
High Performance Computing and Networking