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

Carlos Vega Moreno
Introduction
Introduction

\[
\text{Diff} = t2 - t1
\]

image credits: xkcd.com
Introduction
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, matching 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.

\[
\text{Hash Value} = \text{Src. IP} \oplus \text{Src. Port} \\
\quad \oplus \text{Dst. IP} \oplus \text{Dst. Port}
\]

\[
\text{Consumer} = \begin{cases}
\text{Request:} & \text{Src. IP} \oplus \text{Src. Port} \oplus \text{Dst. IP} \oplus \text{Dst. Port} \\
\quad \oplus \text{ACK} \oplus (\text{Ack}_1 \oplus \text{Ack}_2 \oplus \text{Ack}_3 \oplus \text{Ack}_4) \\
\text{Response:} & \text{Src. IP} \oplus \text{Src. Port} \oplus \text{Dst. IP} \oplus \text{Dst. Port} \\
\quad \oplus \text{SEQ} \oplus (\text{Seq}_1 \oplus \text{Seq}_2 \oplus \text{Seq}_3 \oplus \text{Seq}_4) \\
\end{cases}
\]
500 GB of test data, with more than 700 million packets and 16 million HTTP transactions.

Obtained performance: 10~13 Gbps with a single core Intel Xeon.

20 Gbps and more with an efficient method for load balancing.
500 GB of data, with ≈700 million packets and 16 million HTTP transactions
Results

Hash Value = Src. IP ⊕ Src. Port
⊕ Dst. IP ⊕ Dst. Port

Consumer = \{ Request: \{ Src. IP ⊕ Src. Port ⊕ Dst. IP ⊕ Dst. Port ⊕ ACK ⊕ (Ack_1 ⊕ Ack_2 ⊕ Ack_3 ⊕ Ack_4) \},
                  Response: \{ Src. IP ⊕ Src. Port ⊕ Dst. IP ⊕ Dst. Port ⊕ SEQ ⊕ (Seq_1 ⊕ Seq_2 ⊕ Seq_3 ⊕ Seq_4) \} \mod n \}
Results

Processing speed comparison

Tool used for the analysis

HTTP Analyzer: 11.21 Gbps
Tshark without reassembly: 0.26 Gbps
Tshark: 0.18 Gbps
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

- 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