TCP Proxy Bypass: all the gain with no pain!

G. Siracusano*, R. Bifulco†, S. Salsano*
University of Rome Tor Vergata*, NEC Laboratories Europe†

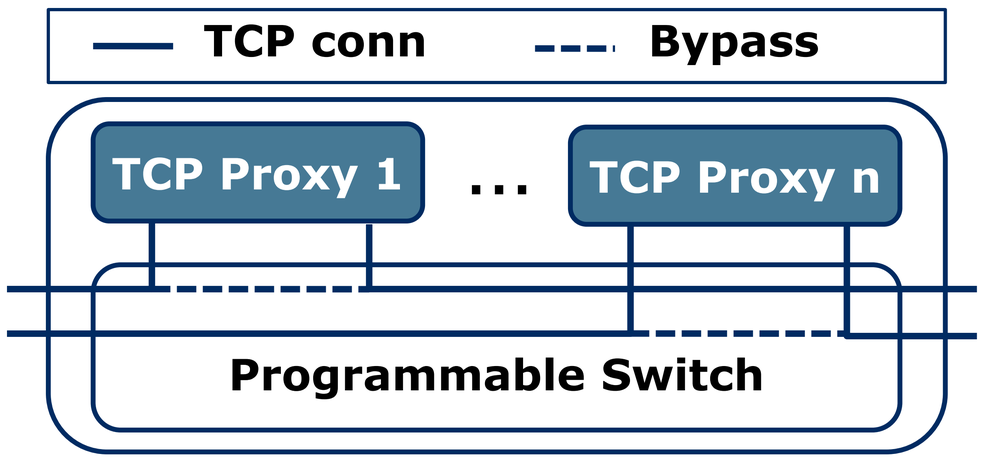
Motivation

- TCP proxies are widely deployed in modern networks [1].
  - Level 7 load balancers.
  - Redirection proxies.
  - Protocol optimizer proxies in celluar networks.
- In many cases, the proxy is only required during the initial phases of the connection.
  - Often TCP connections carry just a single HTTP request [2].
  - With the increase of TLS encrypted traffic only the first few frames of the connection can be (meaningfully) processed by the proxy [3].

Solution

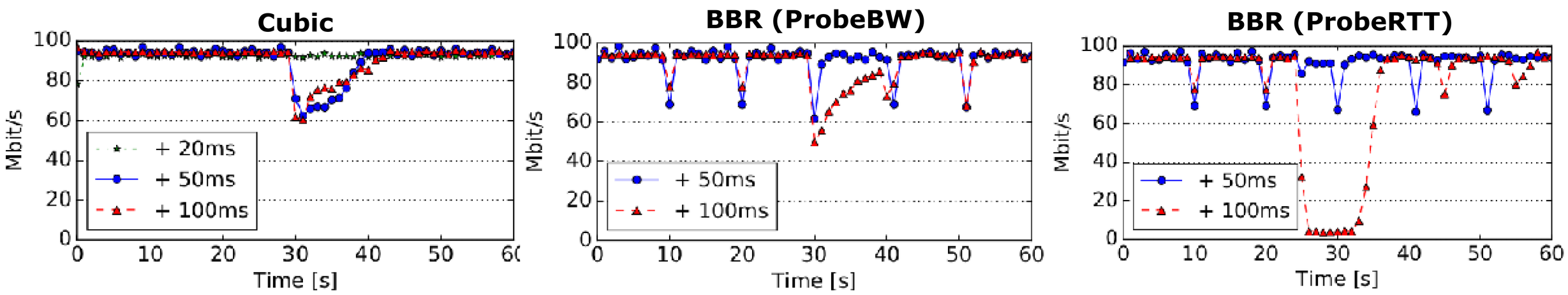
- **Idea:** Can established connections be offloaded from the TCP proxy?
- **Solution:** Use programmable switches to offload an established TCP connection
  - When proxy's operations are limited to relaying packets.
  - Bypass operations:
    - NAPT (Network Address & Port Translation) operation.
    - ACK and SEQ number translation
  - Handle TCP Options:
    - The same set of options has to be negotiated for both client and server.
    - Server side options can be cached and then negotiated with the client.
- **Expected benefit**: Save precious resources by transparently remove a TCP proxy from the data path.

PoC Implementation



- Unikernel TCP proxy (Miniproxy [4]) plus a Programmable Switch (PISCES [5]).
  - The operating system needs to expose additional information at socket level.
  - The switch has to parse and modify TCP header fields, such as TCP sequence and ACK numbers.
- Workflow:
  - The Proxy selects the TCP connections to be offloaded.
  - It waits until there are no in-flights packets on both the client- and server-side connections.
  - The Proxy configures the translation operation in the switch.
  - The switch performs the bypass operations and the packet forwarding for the offloaded connections.
- Considerations on system load:
  - Configuration: proxy as a **virtual machine** and PISCES as **software switch**.
  - Typical configuration **one core assigned for each proxy**, at least one core for the software switch.
  - With all the connections offloaded, the **proxy's CPU cores stay idle**, while the switch resource consumption does not increase.

Impact on congestion control



- TCP flow for 60s over a 100Mbps link. After 30s (25s in rightmost pic.), additional delay is added on the link, summing up to a total RTT value of 100ms for all the cases.
- Removal of a proxy corresponds to a sudden variation of the perceived end-to-end **RTT**.
- To avoid side effects on the connection's throughput, we introduce a gradual synthetic delay, e.g., in steps of 20ms, before performing the connection offload.

REFERECENCES
[1] Xing Xu et al.2015. Investigating transparent web proxies in cellular networks. In PAM. Springer.
[2] Mohammad Al-Fares et al.2011. Overclocking the Yahoo!: CDN for Faster Web Page Loads(IMC '11). ACM, New York, NY, USA.
[3] David Naylor et al.2014. The cost of the S in HTTPS. In CoNEXT. ACM, New York, NY, USA.
[4] Giuseppe Siracusano et al.2016. On the Fly TCP Acceleration with Miniproxy. In ACM SIGCOMM HotMiddlebox.
[5] Muhammad Shahbaz et al.2016. Pisces: A programmable, protocol- independent software switch. In ACM SIGCOMM.

ACKNOWLEDGMENTS
This paper has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 671566 ("Superfluidity"). This paper reflects only the authors' views and the European Commission is not responsible for any use that may be made of the information it contains.