Understanding Performance Implications of Third-Party Network Function

Shuwen Jethro Sun, David Choffnes

Khoury College of Computer Sciences, Northeastern University

{jethrosun,choffnes}@ccs.neu.edu

Background and Motivation

Network function virtualization (NFV) has become a popular approach to bring new functionality into existing networks in a cost-effective way. Most of the research in this area focuses on relatively simple applications intended to run on behalf of a network provider or enterprise, such as firewalls and load balancers. In this paper, we ask: *what would happen if network functions (NFs) deployed in a network were supplied by users or devices, instead of by the network owners*? Specifically, we focus on the question of the resource consumption, bottlenecks, and interactions among NFs provided by third parties (i.e., not the network operators) that go beyond simple load balancers and firewalls to include functionality specified by devices or users such as TLS validation, transcoding, and webpage processing.

Our analysis begins with a survey of recent NFV research to identify gaps in our understanding of how third-party NFs would perform and how they should be evaluated. Based on these gaps, we build four third-party NFs that are designed to test different potential resource bottlenecks and evaluate them under various workloads and scenarios to understand how they interact with the NFV platform and with each other. Main findings:

- Third-party NFs can be extremely resource intensive which can cause contention to harm the performance in coresident scenarios;
- Existing NFV metrics such as packet throughput and packet processing latency provide very little insight into the performance of the third-party NFs.
- The interaction between different types of third-party NFs triggers different behavior in action, where it can contribute to resource intensiveness, execution pattern, or robustness of the third-party NF.

Key Questions

- Would NFs react differently in practice with different kind of workload?
- Can we use NFV to build diverse, user-level NFs?
- What would be the new challenges if we want to apply NFV to these new directions?

Implementation

Prototype NFs are implemented in NetBricks [1] and in safe Rust.

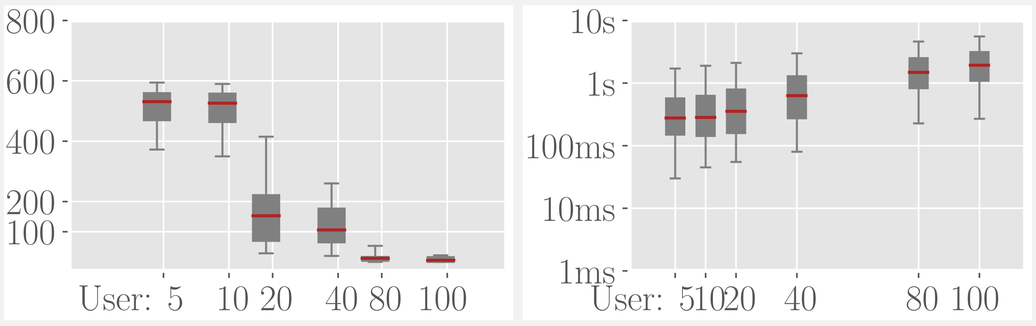
- Running NFs & measuring perf: 500
- 4 Third-party NFs: 1600
- 6 coresident NFs: 1900 LOC

Measurement

For the latency (packet processing time) measurement, we

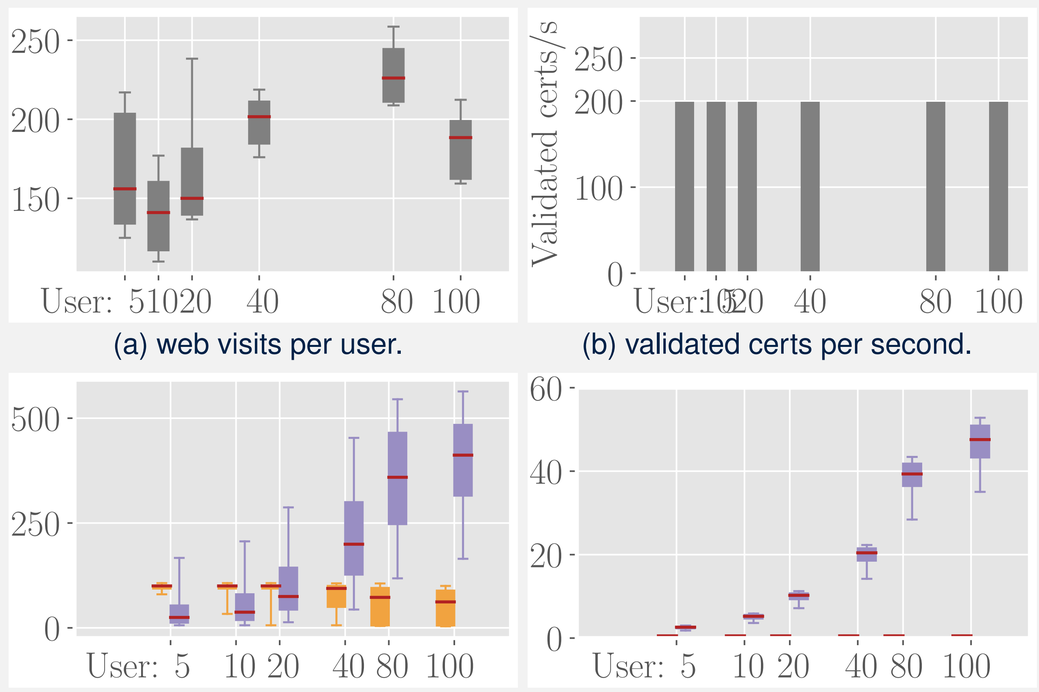
- Measure the processing time of each and every individual packets;
- Calculate the time delta from the time when the packet enters the NF to when the packet leaves the NF

Results



(a) packet tput. (b) web browsing latency.

Results cont'd

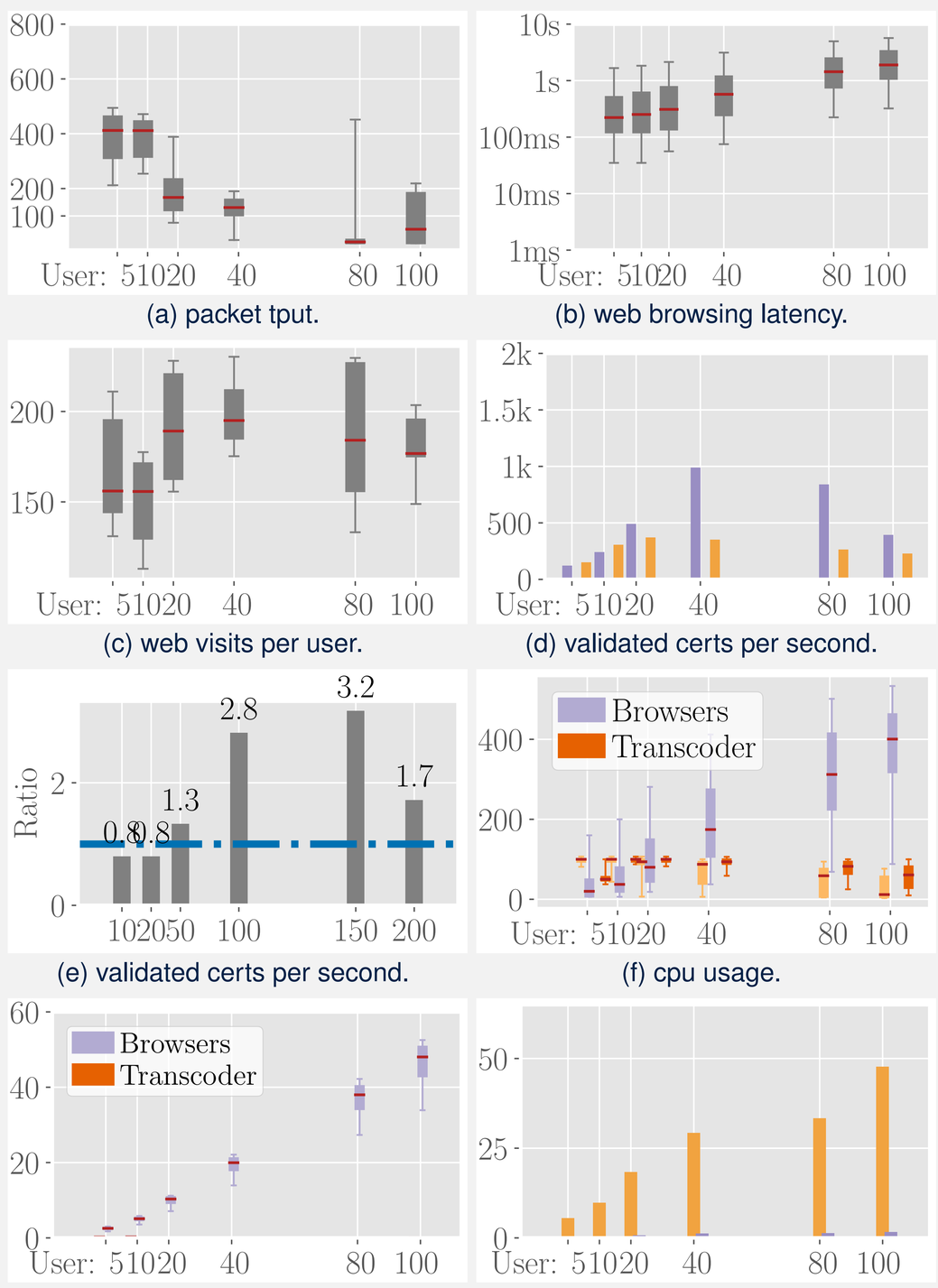


(a) web visits per user. (b) validated certs per second.

(c) cpu usage. (d) memory usage.

(e) tcp usage.

Figure: TLSV-RDR Coresident NF.



(a) packet tput. (b) web browsing latency.

(c) web visits per user. (d) validated certs per second.

(e) validated certs per second. (f) cpu usage.

(g) memory usage. (h) tcp usage.

Figure: RDR+XCDR Coresident NF.

References

[1] Aurojit Panda, Sangjin Han, Keon Jang, Melvin Walls, Sylvia Ratnasamy, and Scott Shenker. NetBricks: Taking the v out of NFV. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 203–216, 2016.

Northeastern University

Cybersecurity and Privacy Institute at Northeastern