A Real-world, End-to-end Traffic Generator

HAMMER

I. Prevezanos, A. Angelou, C. Tselios, A. Stergiakis, V. Tsogkas and G. Tsolis

Citrix Systems Inc.
Patras, Greece

SUPERFLUIDITY

Problem

Since network agility, end-user quality of experience and overall performance are considered market disruptive factors, depending on obsolete software packages and outdated methods for network testing is a suboptimal approach for all players [1],[2].

Modern networks need a traffic generation platform capable of:

1. Generating a relative high load of realistic network traffic
2. Following a modular design for support and maintenance purposes
3. Keeping hardware requirements to the absolute minimum
4. Being deployed easily and relatively fast
5. Remaining testbed-agnostic and user-friendly

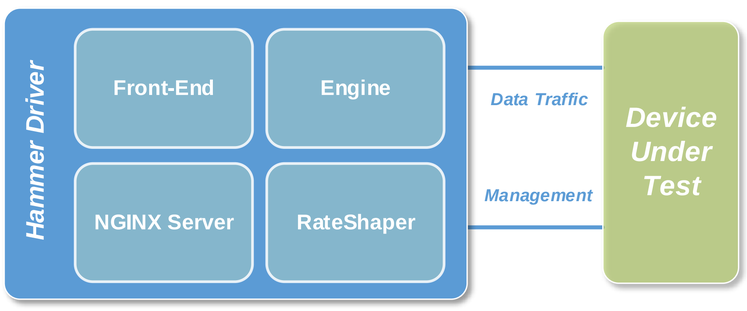
Solution

For providing a contemporary tool which generates network traffic that emulates real-world network conditions, we developed Hammer, an end-to-end network traffic generator capable of simulating complex and dynamic network, user and server behaviors.

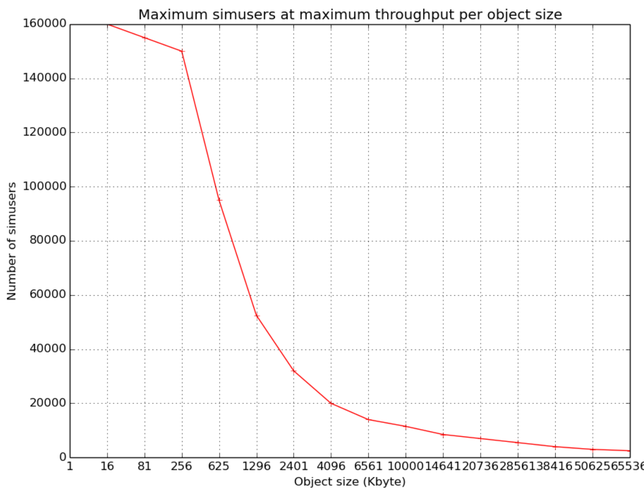
Design

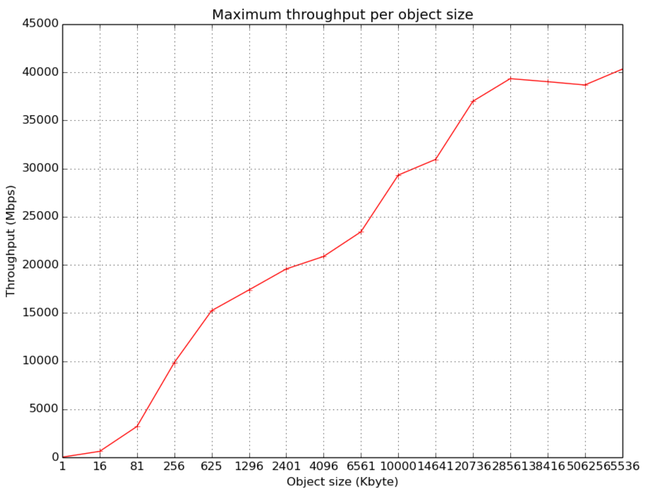
Hammer is designed largely on the premise that the tool can be installed easily in a lab with as low hardware requirements as possible. The host can be physical or virtual while Hammer is an application built on top of an Ubuntu Server base system. The interconnected testbed may vary in size and complexity.

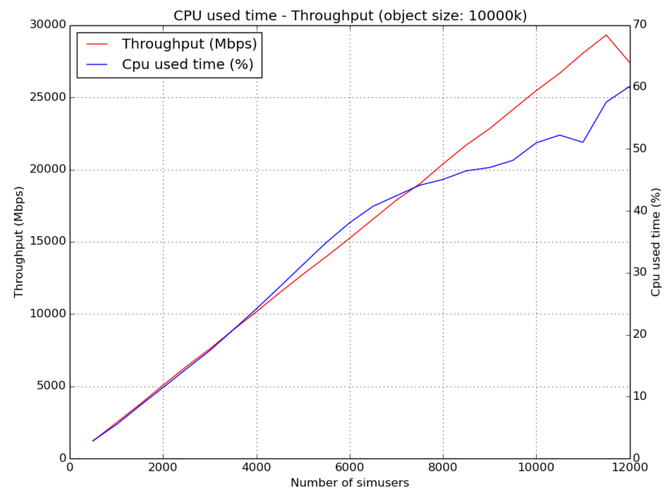
Any assortment of devices that expose an ingress and an egress network interface can be placed under test using Hammer. Ingress and Egress networks emulate clients and servers respectively. All traffic is routed back and forth through the testbed, while all clients and terminating servers are located inside the Hammer host.

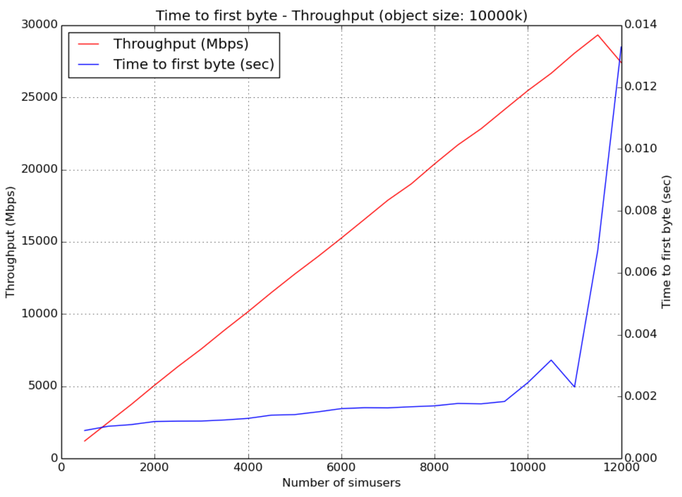


Benchmarking





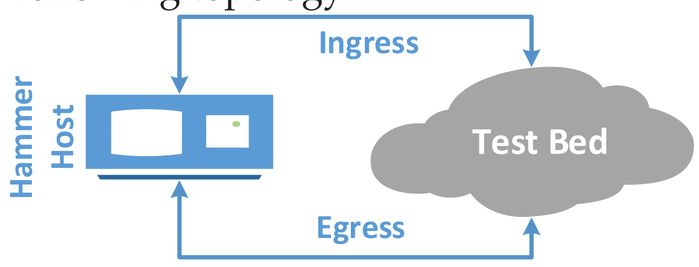




Our tests show that Hammer's operation is linearly linked to the underlying hardware resources, however, even when the simulator is installed in a resource-bound environment, it can still deliver traffic loads that correspond to thousands of interconnected users each with real-world behavior per session.

Demonstration

In order to demonstrate Hammer, we have selected three distinct scenarios and created different profiles for each one. In **Scenario One**, our intention is to show the rise on CPU utilization percentage when a certain number of simusers is introduced. Provided that simusers are introduced by Lua script execution in Hammer Engine module, CPU utilization increase is expected when adequate number of simusers is generated. **Scenario Two** demonstrates Hammer's ability to saturate Gigabit Ethernet networking connections, even when the deployed Hammer Engine node has limited available resources. **Scenario Three**, is a combination of the previous scenarios, by introducing enough simusers to increase CPU utilization, and have each of those simusers generate enough traffic for driving the available networking interfaces to their functional limits. All three scenarios will be executed in the following topology:



Conclusions

We have presented Hammer, a real-world, end-to-end network traffic simulator, capable of simulating complex and dynamic network, user and server behaviors. The focus of this tool is to primarily facilitate investigations related to product stability, for instance different aspects of capacity, longevity, memory leaks, cores and also handle customer content testing that will reveal the behavior of the device under test in realistic network conditions. Hammer has a modular design which offers excellent scalability, rendering the platform capable of being installed on commodity hardware.

In addition, Hammer has a resource-savvy nature, thus requires limited computational resources to generate significant traffic loads. Instead of operating on a packet level, Hammer offers application-layer workload interaction along with inherent data plane acceleration, delivering brisk performance with unparalleled flexibility and ease of use. Hammer can replace numerous commercial solutions since it is able to collect metrics from various layers and functions of a platform.

References

[1] G. Bianci, et al. Superfluidity: a flexible functional architecture for 5G networks. In *TETT Vol. 27(9), pg. 1178-1186, 2016.*

[2] C. Tselios and G. Tsolis. On QoE-awareness through Virtualized Probes in 5G Networks. In *IEEE CAMAD, pg. 159-164, 2016.*

Acknowledgements

This work is partly supported by the European Commission under the auspices of Superfluidity Project, Horizon 2020 Research and Innovation Action (Grant Agreement No. 671566).