Distributed active measurement of Internet queuing delays

P. Casoria[1,3], D. Rossi[1], J. Auge[2], M-O. Buob[2], T. Friedman[2] and A. Pescape[3]

[1] Telecom ParisTech, first.last@enst.fr
[2] UPMC Sorbonne Universites first.last@lip6.fr
[3] Universitá di Napoli Federico II first.last@unina.it

Abstract. Despite growing link capacities, over-dimensioned buffers are still causing, in the Internet of the second decade of the third millennium, hosts to suffer from severe queuing delays (or bufferbloat). While maximum bufferbloat possibly exceeds few seconds, it is far less clear how often this maximum is hit in practice. This paper reports on our ongoing work to build a spatial and temporal map of Internet bufferbloat, describing a system based on distributed agents running on PlanetLab that aims at providing a quantitative answer to the above question.

1 Introduction

Given the abundance of active measurement approaches, it may seem at first sight redundant to focus on bufferbloat measurement via active techniques. Yet, this work nicely fit in a gap of the design space explored by the research community.

Our system targets large-scale high-frequency scanning, customized to periodically report very detailed per-host statistics (e.g., percentiles). As individual probes are capable of scanning about 10K hosts in a second, it follows that using 100 PlanetLab nodes we could in principle follow about 1 million hosts every second or, trading space for time, cover the whole Internet in about one hour. Interest of our approach can be summarized as follows.

Due to architectural similarities with scanners [1, 10] and systems based on distributed agents [2, 3], our approach allows to achieve spatial scales larger than [8, 11] (already in this paper) and [4,5] (prospectively). Additionally, our efficient implementation allows much higher scan frequency than [2–5, 8, 11], where the frequency of background latency measurement is typically too sparse to offer an adequate bufferbloat characterization from the user perspective. Finally, in terms of the delay statistics, we avoid to measure *maximum* latency under controlled load as in [8, 9, 11], and rather gather the (typical) delay by continuous host measurement, hence sampling the user load during their *normal* activities.

2 Bufferbloat scanner architecture

Building over TopHat [6], we design a distributed architecture for Internet bufferbloat scanning. At the core of our scanner, lay an efficient tool to ping a large amount of hosts

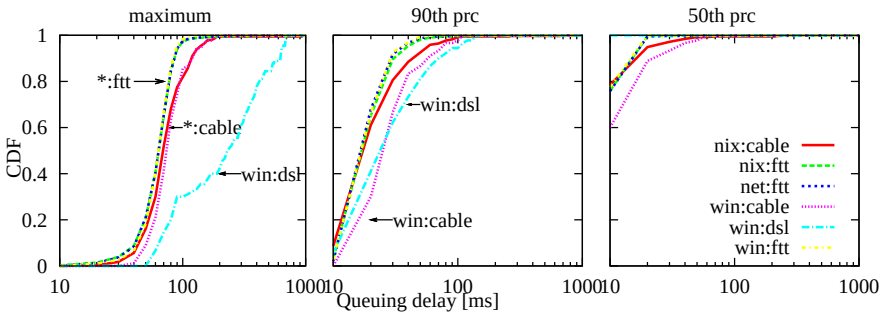

Fig. 1: Validation of the Internet measurement campaign.

with the least possible resources. While our tool is far less efficient than the recently released zMap, it is still an order of magnitude faster than the fastest settings of the Nmap Scripting Engine.

We divide measurement periods (of 5 minutes by default), at the beginning of which each measurement server ask for instructions (essentially, a list of destinations/subnets and the sampling frequency, 1 Hz by default). At the end of each measurement period, per-hosts statistics (delay percentiles, etc.) are collected for further post-processing.

For each target, we gauge the queuing delay via ICMP measurements as $q_i = RTT_i - \min_{j \leq i} RTT_j$. By ensuring that queuing does not happens at the measurement servers, we can however correctly infer the remote queuing delay. We validate this approach (i) to be very accurate with non-NATted hosts, (ii) to yield a coarse queuing indication (e.g., a binary bufferbloat flag) for NATted hosts.

We notice that the generally measurement are *initiated* by the end-host [4, 5, 8, 9] (SamKnows/BISmark [11] slightly differ in that measurement starts from the HGW). In our case, measurement are instead *targeting* the end-hosts: this is common in large-scale census studies [1, 10] (of which we inherit the scalability property) but has not been explored so far, to the best of our knowledge, for bufferbloat measurements.

3 Measurement campaign

We report results on a preliminary measurement campaign. We focus on moderate number of hosts O(10^4) on the same ISPs, that we continuously probe at 0.5 Hz frequency from 2 separate PlanetLab nodes for a period of about 8 continuous hours. Overall, we receive replies to 47% of our sent packets, for a total of O(10^8) valid samples – using only two PlanetLab servers, we already achieve a quite significant scale in terms of spatial reach and temporal frequency.

For validation purposes, we infer (i) the access type (AT) of our target hosts by issuing reverse DNS queries, as well as (ii) the remote operating system (OS) through `nmap` fingerprinting. As for the access type, we expect the breakdown of queuing delay along DSL, FTTH and cable access to yield an intuitive validation of the observed statistics.

Additionally, we argue that in case the remote OS is reliably found to be a Windows OS, then queuing delay are representative of non-NATted host statistics (where our methodology is more reliable). Overall, we manage to infer both AT and OS information for 2546 hosts: while this subset is not statistically significant, it nevertheless allows to validate our methodology as it covers the full AT$\times$ OS cross-product.

We collect per-host percentiles during 5 minutes windows: Fig. 1 reports the Cumulative Distribution Function (CDF) of a few per-host queuing delay statistics, gathered over all hosts and measurement rounds. Left plot reports the maximum queuing delay CDF: as expected, from the picture clearly emerges that (a) fiber access suffers the lowest delays irrespectively from the OSs, (b) cable delays are only slightly higher, whereas (c) DSL end-hosts may seldom suffer from delays close to 1 sec. We further report the 50th (right) and 90th (middle) percentiles CDF. Notice further that (d) from practical purposes, the 90th percentile is lower than 100 ms under any combination of OS and AT – including end-hosts behind DSL. Moreover, since the *win:cable* and *win:dsl* lines now clearly separate from the others, we infer that the methodology needs to be refined as it likely underestimates bufferbloat delay for NATted hosts – although observation (d) suggests bufferbloat to be a not necessarily frequent problem.

Acknowledgements

This work has been carried out during Pellegrino Casoria internship at LINCS `http://www.lincs.fr`. The research leading to these results has received funding from the European Union under the FP7 Grant Agreement n. 318627 (Integrated Project "mPlane").

References

1. http://internetcensus2012.bitbucket.org/.
2. `http://www.caida.org/projects/ark/`.
3. `http://www.netdimes.org/new/`.
4. Z. Bischof, J. Otto, M. Sánchez, J. Rula, D. Choffnes, and F. Bustamante. Crowdsourcing ISP characterization to the network edge. In *ACM SIGCOMM W-MUST*, 2011.
5. Z. S. Bischof, J. S. Otto, and F. E. Bustamante. Up, down and around the stack: ISP characterization from network intensive applications. In *ACM SIGCOMM W-MUST*, 2012.
6. T. Bourgeau, J. Augé, and T. Friedman. Tophat: supporting experiments through measurement infrastructure federation. In *TridentCom*, 2010.
7. C. Chirichella and D. Rossi. To the moon and back: are internet bufferbloat delays really that large. In *IEEE INFOCOM Workshop on Traffic Measurement and Analysis (TMA)*, 2013.
8. H. Jiang, Y. Wang, K. Lee, and I. Rhee. Tackling bufferbloat in 3G/4G networks. In *ACM IMC*, 2012.
9. C. Kreibich, N. Weaver, B. Nechaev, and V. Paxson. Netalyzr: Illuminating the edge network. In *ACM IMC*, 2010.
10. D. Leonard and D. Loguinov. Demystifying service discovery: implementing an internet-wide scanner. In *ACM IMC*, 2010.
11. S. Sundaresan, W. de Donato, N. Feamster, R. Teixeira, S. Crawford, and A. Pescapè. Broadband internet performance: a view from the gateway. In *ACM SIGCOMM*, 2011.