Poster: Empirically Testing the PacketLab Model

Tzu-Bin Yan
University of Illinois at Urbana-Champaign
tbyan2@illinois.edu

Zesen Zhang
UC San Diego
zez003@eng.ucsd.edu

Bradley Huffaker
CAIDA/UC San Diego
bradley@caida.org

Ricky Mok
CAIDA/UC San Diego
cskpmok@caida.org

kc claffy
CAIDA/UC San Diego
kc@caida.org

Kirill Levchenko
University of Illinois at Urbana-Champaign
klevchen@illinois.edu

ABSTRACT

PacketLab [2] is a recently proposed model for accessing remote vantage points. The core design is for the vantage points to export low-level network operations that measurement researchers could rely on to construct more complex measurements. Motivating the model is the assumption that such an approach can overcome persistent challenges such as the operational cost and security concerns of vantage point sharing that researchers face in launching distributed active Internet measurement experiments. However, the limitations imposed by the core design merit a deeper analysis of the applicability of such model to real-world measurements of interest. We undertook this analysis based on a survey of recent Internet measurement studies, followed by an empirical comparison of PacketLab-based versus native implementations of common measurement methods. We showed that for several canonical measurement types common in past studies, PacketLab yielded similar results to native versions of the same measurements. Our results suggest that PacketLab could help reproduce or extend around 16.4% (28 out of 171) of all surveyed studies and accommodate a variety of measurements from latency, throughput, network path, to non-timing data.

ACM Reference Format:
Tzu-Bin Yan, Zesen Zhang, Bradley Huffaker, Ricky Mok, kc claffy, and Kirill Levchenko. 2023. Poster: Empirically Testing the PacketLab Model. In *Proceedings of the 2023 ACM Internet Measurement Conference (IMC '23), October 24–26, 2023, Montreal, QC, Canada.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3618257.3624999

1 INTRODUCTION

In a short IMC '17 paper, Levchenko *et al.* [2] proposed a new measurement endpoint model called PacketLab that aimed to address the high-cost problem for vantage point sharing, which is critical toward performing distributed active network measurement experiments. The core of the PacketLab design is to provide experimenters (researchers) with a VPN-like interface to measurement endpoints. To run an experiment, an experimenter runs a machine acting as the *experiment controller* that interacts with *measurement endpoints* to send and receive packets on behalf of the controller. Unlike a simple VPN endpoint, PacketLab timestamps all sent and received packets and allows a controller to schedule packet sending from the endpoint. These two features enable various timing-based network measurement that would otherwise be impossible to do accurately using a VPN endpoint.

Like a VPN, PacketLab introduces delay for controller packet issuance and reception. Such characteristic imposes limitations on the possible set of measurements that is feasible under PacketLab, and are critical toward the applicability of the model for distributed active network measurement experiments of interest to the research community. Though the PacketLab authors claimed that the PacketLab model could effectively support the deployment of a significant fraction of such experiments, the authors did not substantiate their claim with evidence. For this work, we addressed this gap by investigating the applicability of PacketLab toward distributed active network measurement experiments. We empirically analyzed the applicability of the PacketLab model based on commonly-used measurement methods used in distributed active network measurement experiments of past studies in IMC and SIGCOMM. We then implemented the representative methods (traceroute and TCP throughput) under the PacketLab model using existing tools released by the PacketLab team on their website (https://pktlab.github.io) and compared the performance of the PacketLab implementations against their native counterparts. Our findings showed that the representative common measurement methods are doable under PacketLab and could give similar results as their native counterpart. Such findings along with on-paper analysis of other methods suggest that the PacketLab is suitable for a large fraction of measurement methods of interest, which effectively substantiate the claim made by the original PacketLab paper.

2 SURVEY OF MEASUREMENT STUDIES

To understand if the PacketLab model would be applicable to studies of current interest, we surveyed measurement studies published in the last three years at IMC and SIGCOMM (SIGCOMM limited to measurement/telemetry sessions for relevance). Among 171 surveyed studies, we found 30 whose main contributions contained a distributed active measurement experiment, as well as 6 that contained experiments that could benefit from access to external vantage points either based on claims made by the authors or if the work involved an Internet-wide scan of HTTP/HTTPS services.

IMC '23, October 24–26, 2023, Montreal, QC, Canada

ACM ISBN 979-8-4007-0382-9/23/10.
https://doi.org/10.1145/3618257.3624999

Type	*Studies*	*Common Collected Data*
Latency	13 (36.1%)	DNS query latency, traceroute hop latency, ping latency, browser latency metrics.
Throughput	8 (22.2%)	TCP pipe filling throughput trace-inferred throughput.
Network Path	9 (25%)	Hop address.
Non-timing Data	25 (69.4%)	DNS resource records, web content.

Table 1: Summary of Internet Measurement Studies Survey Results

Categorizing measurement methods used in the identified experiments of the 36 studies, we identified four main categories of measurements: *latency*, *throughput*, *network path*, and *non-timing data*. Table 1 gives a summary of our results.

We believe that PacketLab has the necessary mechanisms to support all of the above measurements in some form. To confirm this, we implemented two representative experiments: a TCP throughput test, meant to broadly represent all throughput measurements, and traceroute, meant to represent both latency and network path measurements. Our implementation results are given in section 3.

3 EVALUATION

For the TCP throughput and traceroute experiments, we deployed 6 PacketLab endpoints: one on an Intel NUC (7i7BNH) under a local residential fiber connection running Ubuntu 22.04 (kernel v5.19.0) and five endpoints on `t2-medium` AWS instances running Ubuntu 22.04 (kernel v5.15.0), located in the following regions: Oregon, Tokyo, Montréal, Frankfurt, and San Paulo.

For the two experiments, our native implementation counterparts for comparison were Butskoy's traceroute [1] for the traceroute experiment and custom TCP throughput client and server programs written by us. Butskoy's traceroute is a well-known traceroute implementation on Linux, where the default probes used are UDP probes. The custom TCP throughput client and server programs allow transmission of 25 MB of random data in either direction. To allow for more accurate peak throughput estimation, we skipped the TCP slow start phase during throughput computation.

For the PacketLab implementations, we designed them to follow the native implementations as closely as possible. For traceroute, the PacketLab implementation (`pl_traceroute`) sends out UDP probes that contain the exact same values as Butskoy's traceroute, except that IPv4 IPID and UDP source port fields are selected uniformly at random from the range used by the kernel. For TCP throughput, the PacketLab implementation acts similarly to the native client program. However, to accommodate the unique characteristics of PacketLab, we also designed the PacketLab implementation to (1) schedule future endpoint sending to achieve the desired pipe-filling behavior for the server-upload direction and (2) disable and re-enable endpoint to controller events after a sufficiently long period (10 seconds) to prevent bandwidth competition.

We ran both experiments 100 times and collected three metrics: throughput, hop latency, and total unique address count. Our results showed that both implementations gave similar means for throughput and hop latency, with the difference always being less than 7% of those reported by the native implementations (fluctuating between either implementation giving higher mean). For total unique address count, we found that in our experiment `pl_traceroute` always saw more distinct addresses (< 7% of those reported by the native implementation for all endpoints), which we suspected was caused by the value selection method of IPv4 IPID and UDP source port in `pl_traceroute`. We confirmed this by capturing the probes for 100 runs of both implementations and looking at their IPID and UDP source port fields, where we found a given number of probes, `pl_traceroute` gave more unique IPID (+1.8%) and source port (+3.6%) values than the native implementation. With similar statistical results shown for collected metrics, we consider as a whole that the PacketLab model is applicable in producing accurate results for the selected experiments.

PacketLab Feasibility. The fact that PacketLab was successful at performing the representative experiments we chose gives some hope that perhaps PacketLab could support *all* measurements of the tested category. To further understand the limitations of PacketLab, we examined each experiment and considered how we might implement each experiment in PacketLab. While our armchair exercise does not carry the weight of an actual implementation, we believe most studies (28 out of 36, 77.8%) can be implemented in PacketLab, among which 18 are strictly of the non-timing-data-measurement type. Of the remaining 8 experiments, we found the main reason an experiment is not possible under PacketLab is that it also measured some client-side computation such as browser page load time, game client response delay, and video conferencing performance.

Acknowledgments. AWS results presented in this paper were obtained using CloudBank[3], which is supported by the National Science Foundation (NSF) under award #1925001. The PacketLab project is also supported by NSF award #1764055 / 1903612 and a gift from Comcast. The views herein are those of the authors and do not necessarily represent endorsements, either expressed or implied, of NSF.

REFERENCES

[1] Dmitry Butskoy. 2023. TRACEROUTE for Linux. (2023). `https://traceroute.sourceforge.net/`

[2] Kirill Levchenko, Amogh Dhamdhere, Bradley Huffaker, kc claffy, Mark Allman, and Vern Paxson. 2017. Packetlab: A Universal Measurement Endpoint Interface. In *Proceedings of the 2017 Internet Measurement Conference (IMC '17)*. Association for Computing Machinery, New York, NY, USA, 254–260. `https://doi.org/10.1145/3131365.3131396`

[3] Michael Norman, Vince Kellen, Shava Smallen, Brian DeMeulle, Shawn Strande, Ed Lazowska, Naomi Alterman, Rob Fatland, Sarah Stone, Amanda Tan, Katherine Yelick, Eric Van Dusen, and James Mitchell. 2021. CloudBank: Managed Services to Simplify Cloud Access for Computer Science Research and Education. In *Practice and Experience in Advanced Research Computing (PEARC '21)*. Association for Computing Machinery, New York, NY, USA, Article 45, 4 pages. `https://doi.org/10.1145/3437359.3465586`