Implementing network slicing using P4 for the Tactile Internet

Belma Turkovic and Fernando Kuipers
Delft University of Technology, Mekelweg 4, 2628CD Delft, The Netherlands
{B-Turkovic-2, F.A.Kuipers}@tudelft.nl

***Abstract*—The main problem currently limiting the Tactile Internet from becoming a reality stems from its most ambitious requirement, the requirement of extremely low latency. In order to achieve the correct differentiation of this type of traffic in all the network nodes both SDN control plane and data plane programmability are needed. Delay in a typical network consists of four components. While transmission and propagation delay are constant (on a certain path) processing and queuing delay vary. This paper describes a new forwarding mechanism that can be used to minimize the delay introduced in the network by a single node.**

***Index Terms*—Tactile Internet, QoS, SDN, P4, programmable networks**

I. Introduction

The development of new services, like the Tactile Internet, is limited by the very strict network requirements that these flows have. Typical refresh rate of a haptic devices is 1 kHz which equals to 1000 packets/s (as we want to sent the data as soon as it is available) [1]–[4]. This flows are also bidirectional and have higher Quality of Service (QoS) constrains: maximum delay lees than 40ms; maximum jitter less than 10ms; and packet loss less than 10% [2]–[8]. Moreover, often haptic data are synchronized with visual and audio signals and need to be synced with them [9]. Taking all this into account and considering the current state of the Internet, available approaches for providing Quality of Service (QoS) to these new services are limited and inflexible. Service providers need a better solution that is both scalable and allows the fine-grained tuning of network traffic, especially low latency traffic.

Delay in a typical network consists of four components, i.e. transmission, propagation, processing and queuing delay. While transmission and propagation delay are constant (on a certain path) processing and queuing delay vary. The more intermediate routers we encounter along the way, the higher the processing and queuing delay. However, they can be minimized if correct traffic differentiation and appropriate resource management is implemented in every device on the path. Recently, different SDN enabled QoS frameworks have emerged, offering a lot of possibilities for advanced network reconfiguration and queue management [10] but new dataplane techniques [11] are needed in order to reduce both delay components simultaneously as well as to allow isolation of resources used for processing. This paper demonstrated a way this can be implemented on a simple network node.

While network buffers are present to absorb short-term rate fluctuations, they are often not dimensioned and managed properly. The main goal in reducing queuing delay for a given packet flow is to try to avoid standing queues in all the intermediate nodes. In our solution every node has a separate queue for tactile data that is prioritized. If packets are present in this queue they are going to be processed before any other traffic, thus guaranteeing the lowest queuing delay for this type of traffic. Because the expected throughput of a tactile flow is in the order of 100 kpbs this approach will never lead to the starvation of other flows present on the device. It is demonstrated that this approach significantly reduces the variation of the delay (i.e. jitter) present in this networks for this types of flows.

On the other hand, P4 [11] offers the possibility to implement new forwarding schemes for different types of traffic thus providing isolation between different applications and consequently reducing processing delay. General-purpose network stacks exhibit unnecessary overheads when used with a specialized application like a simple packet forwarding system. Traditional packet processors are designed to handle all possible cases for the network stack, use memory structures that are very complex so the allocation and deallocation of them takes too much CPU cycles to process thus increasing the latency and decreasing the throughput of the node. Packet processing applications also exhibit a high degree of data parallelism. In theory, stateless processing can be parallelized indefinitely, given enough processing and memory resources, since each processing thread works independently on a given packet [12]. This leads to the conclusion that given the right resource reservation (for the tactile data) best effort traffic will not impact the tactile data that is being processed. The implemented design choice for this tactile slice is to keep it as simple as possible in order to achieve high performance, high throughput and low latency. In order to do this the number of cycles per tactile packet needs to be minimized, because the CPU limits the number of packet headers that can be processed, while bandwidth and latency of the I/O buses limit the total throughput [13]–[15]. The number of match-action tables used to process these packets is thus minimized as every single apply action of the table takes around 900 additional CPU cycles. Because P4 offers the ability to add arbitrary data fields, we created tables that add a new label identifying the tactile flow on the egress routers and then these packets are processed with minimal delay at all the other routers according to that label. The label distribution is handled by a centralized controller that takes into account the different requirements of these flows and finds the paths that have the minimum delay (i.e. are not congested).

Moreover, most of the processing actions applied to packets are fairly simple, often a sequence of arithmetic operations. The complexity, typically, lies in matching an incoming packet header against a set of header patterns stored in a table. The match type can be exact, longest-prefix, range, exact or wildcard. This aper demonstrates that exact matching, as opposed to longest prefix match used in standard routers, is the fastest lookup technique and should be used for the tactile flows. Other important part is cache management. Cache thrashing causes the cycles per packet to increase too as the actual processing of the packet is idling for data. By separating the forwarding tables for the tactile slice and best effort slice, the number of entries in the tactile table is reduced and thus the processing delay. Also, as cache thrashing occurs due to the size of the tables (i.e. number of flow table entries), dedicating specific parts of the memory only to these small tactile forwarding tables will reduce the artifacts caused by this (delays and jitter).

The remainder of this paper is structured as follows: We present related work in section II. Section III provides an overview of the implementation used to reduce the processing delay and Section IV the overview of the priority queuing used to minimize the queuing delay in all the nodes. Results and measurements are presented in the Section V.

References

[1] E. Steinbach, S. Hirche, J. Kammerl, I. Vittorias, and R. Chaudhari, "Haptic data compression and communication," *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 87–96, Jan 2011.

[2] V. Gokhale, O. Dabeer, and S. Chaudhuri, "Hoip: Haptics over internet protocol," in *2013 IEEE International Symposium on Haptic Audio Visual Environments and Games (HAVE)*, Oct 2013, pp. 45–50.

[3] V. Gokhale, S. Chaudhuri, and O. Dabeer, "Hoip: A point-to-point haptic data communication protocol and its evaluation," in *2015 Twenty First National Conference on Communications (NCC)*, Feb 2015, pp. 1–6.

[4] F. Brandi and E. Steinbach, "Prediction techniques for haptic communication and their vulnerability to packet losses," in *2013 IEEE International Symposium on Haptic Audio Visual Environments and Games (HAVE)*, Oct 2013, pp. 63–68.

[5] T. H. Tee, K. M. Yap, A. Marshall, and K. S. Eu, "A quantitative evaluation of haptic data prediction techniques over best-effort network," in *16th International Conference on Advanced Communication Technology*, Feb 2014, pp. 762–768.

[6] T. H. Tee, K. S. Eu, K. M. Yap, A. Marshall, and T. H. Lee, "A network-adaptive compensation technique for tele-haptics using position prediction algorithm," in *2013 IEEE International Symposium on Haptic Audio Visual Environments and Games (HAVE)*, Oct 2013, pp. 39–44.

[7] X. Xu, B. Cizmeci, A. Al-Nuaimi, and E. Steinbach, "Point cloud-based model-mediated teleoperation with dynamic and perception-based model updating," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 11, pp. 2558–2569, Nov 2014.

[8] Z. Shi, H. Zou, M. Rank, L. Chen, S. Hirche, and H. J. Muller, "Effects of packet loss and latency on the temporal discrimination of visual-haptic events," *IEEE Transactions on Haptics*, vol. 3, no. 1, pp. 28–36, Jan 2010.

[9] C. Bachhuber and E. Steinbach, "Are today's video communication solutions ready for the tactile internet?" in *2017 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, March 2017, pp. 1–6.

[10] A. Mendiola, J. Astorga, E. Jacob, and M. Higuero, "A survey on the contributions of software-defined networking to traffic engineering," *IEEE Communications Surveys Tutorials*, vol. 19, no. 2, pp. 918–953, Secondquarter 2017.

[11] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, and D. Walker, "P4: Programming protocol-independent packet processors," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, pp. 87–95, Jul. 2014.

[12] S. Abdi, U. Aftab, G. Bailey, B. Boughzala, F. Dewal, S. Parsazad, and E. Tremblay, "Pfpsim: A programmable forwarding plane simulator," in *2016 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, March 2016, pp. 55–60.

[13] D. Raumer, F. Wohlfart, D. Scholz, and G. Carle, "Performance Exploration of Software-based Packet Processing Systems," in *Proceedings of Leistungs-, Zuverlässigkeits- und Verlässlichkeitsbewertung von Kommunikationsnetzen und Verteilten Systemen, 6. GI/ITG-Workshop MMBnet 2015*, Hamburg, Germany, Sep. 2015.

[14] S. Gallenmller, P. Emmerich, F. Wohlfart, D. Raumer, and G. Carle, "Comparison of frameworks for high-performance packet io," in *2015 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, May 2015, pp. 29–38.

[15] P. Emmerich, D. Raumer, F. Wohlfart, and G. Carle, "Assessing soft- and hardware bottlenecks in pc-based packet forwarding systems," 2015.