Workgroup: Network Working Group
Published: 16 May 2023
Intended Status: Informational
Expires: 17 November 2023
Authors: R. Miao, Alibaba Group
S. Anubolu, Broadcom Inc
R. Pan, Intel Corporation
J. Lee, Intel Corporation
B. Gafni, NVIDIA
Y. Shpigelman, NVIDIA
J. Tantsura, Nvidia
G. Caspary, Cisco Systems

Inband Telemetry for HPCC++

Abstract

Congestion control (CC) is the key to achieving ultra-low latency, high bandwidth and network stability in high-speed networks. However, the existing high-speed CC schemes have inherent limitations for reaching these goals.

In this document, we describe HPCC++ (High Precision Congestion Control), a new high-speed CC mechanism which achieves the three goals simultaneously. HPCC++ leverages inband telemetry to obtain precise link load information and controls traffic precisely. By addressing challenges such as delayed signaling during congestion and overreaction to the congestion signaling using inband and granular telemetry, HPCC++ can quickly converge to utilize all the available bandwidth while avoiding congestion, and can maintain near-zero in-network queues for ultra-low latency. HPCC++ is also fair and easy to deploy in hardware, implementable with commodity NICs and switches.

Status of This Memo

Table of Contents

1. Introduction

The link speed in data center networks has grown from 1Gbps to 100Gbps in the past decade, and this growth is continuing. Ultralow latency and high bandwidth, which are demanded by more and more applications, are two critical requirements in today's and future high-speed networks.

Given that traditional software-based network stacks in hosts can no longer sustain the critical latency and bandwidth requirements as described in [Zhu-SIGCOMM2015], offloading network stacks into hardware is an inevitable direction in high-speed networks. As an example, large-scale networks with RDMA (remote direct memory access) often uses hardware-offloading solutions. In some cases, the RDMA networks still face fundamental challenges to reconcile low latency, high bandwidth utilization, and high stability.

This document describes a new congestion control mechanism, HPCC++ (Enhanced High Precision Congestion Control), for large-scale, high-speed networks. The key idea behind HPCC++ is to leverage the precise link load information from signaled through inband telemetry to compute accurate flow rate updates. Unlike existing approaches that often require a large number of iterations to find the proper flow rates, HPCC++ requires only one rate update step in most cases. Using precise information from inband telemetry enables HPCC++ to address the limitations in current congestion control schemes. First, HPCC++ senders can quickly ramp up flow rates for high utilization and ramp down flow rates for congestion avoidance. Second, HPCC++ senders can quickly adjust the flow rates to keep each link's output rate slightly lower than the link's capacity, preventing queues from being built-up as well as preserving high link utilization. Finally, since sending rates are computed precisely based on direct measurements at switches, HPCC++ requires merely three independent parameters that are used to tune fairness and efficiency.

HPCC++ is an enhanced version of [SIGCOMM-HPCC]. HPCC++ takes into account system constraints and aims to reduce the design overhead and further improves the performance. Detailed specification about HPCC++ can be found at [draft-miao-tsv-hpcc].

This document describes the architecture changes in switches and end-hosts to support the needed tranmission of inband telemetry and its consumption, that imporves the efficiency in handling network congestion.

2. Inband telemetry padding at the network switches

HPCC++ only relies on packets to share information across senders, receivers, and switches. The switch should capture inband telemetry information that includes link load (txBytes, qlen, ts) and link spec (switch_ID, port_ID, B) at the egress port. Note, each switch should record all those information at the single snapshot to achieve a precise link load estimate. Inside a data center, the path length is often no more than 5 hops. The overhead of the inband telemetry padding for HPCC++ is considered to be low.

As long the above algorithm is met, HPCC++ is open to a variety of inband telemetry format standards, which are orthogonal to the HPCC++ algorithm. Although this document does not mandate a particular inband telemetry header format or encapsulation, we provide concrete implementation specifications using strandard inband telemetry protocols, including IFA [I-D.ietf-kumar-ippm-ifa], IETF IOAM [RFC9179], and P4.org INT [P4-INT]. In fact, the emerging inband telemetry protocols inform the evolution for a broader range of protocols and network functions, where this document leverages the trend to propose the architecture change to support in-network functions like congestion control with high efficiency.

2.1. Inband telemetry on IFA2.0

For more details, please refer to IFA [I-D.ietf-kumar-ippm-ifa]

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  lns  |  deviceID                             |     rsvd      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Speed |     rsvd      |          rxTimestampSec               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       egressPort              |         ingressPort           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        rxTimeStampNs                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         residenceTime                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            txBytes                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       rsvd                    |      Queue Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             rsvd                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: Example IFA header

Figure 1 shows the packet format of the INT metadata after UDP and IFA metadata header. The field lns is the local name space and defines the format of the metadata. The field deviceID is a 20-bit field that uniquely identifies the device in the network. The Speed field is an encode field with the following encoding for port speed: 0 - 10G, 1 - 25G, 2 - 40G, 3- 50G, 4 - 100G, 5 - 200G, 6 - 400G. The field cn is the congestion field and denotes if the packet experienced congestion.

2.2. Inband telemetry on IOAM

IOAM is the technology adopted by IETF to be used for in-situ telemetry. For the use of HPCC++ we would discuss the IOAM trace option as part of the IOAM architecture. IOAM trace supports both Pre-allocated and Incremental trace Options, meaning that a node in the network may either write data into an already-allocated space in the packet, or may it add the data as an extenation to the IOAM header, respectively. An IOAM data header has a modular design, where the data types written by a node are determined based on the IOAM trace header instruction list. For the full description of the IOAM header design please refer to IETF IOAM [RFC9179] specification. In order to fulfill the requirements set by the HPCC++ architecture we would suggest to use the below trace types:

*Hop_Lim and node_id Short

*Ingress_if_id and egress_if_id Short

*Queue Depth

*Timestamp Fraction: To be used as egress timestamp rather than an ingress timestamp

*Transmitted Bytes

Note that Transmitted Bytes trace type is defined in [I-D.draft-gafni-ippm-ioam-additional-data-fields] as a suggested extension to [RFC9179].

When using the above trace types, the IOAM data header would be constructed as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Hop_Lim     |              node_id                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     ingress_if_id             |          egress_if_id         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         queue depth                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     timestamp fraction                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            tx_bytes                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: Example of an IOAM data header

2.3. Inband telemetry on P4.org INT

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Node ID (Nth hop)                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Ingress Interface ID     |      Egress Interface ID      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Queue ID   |                Queue occupnacy                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Egress timestamp                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Egress timestamp (cont'd)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Egress interface Tx utilization              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Node ID (N-1th hop)                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               ...                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Node ID (1st hop)                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               ...                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Example P4.org INT v2.1 per-hop metadata header

Figure 3 shows the per-hop metadata format of the P4.org INT-MD mode (following INT v2.1 spec). Each hop switch along the path adds its Node ID for the sender to be able to track the path and detect a path change event. If so, it throws away the existing status records of the flow and builds up new records. Queue occupancy (24 bits) is the current buffer occupancy of the egress port and queue that the flow is going through. Egress timestamp (8 bytes) is used by HPCC++ algorithm to eventually compute interface utilization. Since P4.org INT reports Egress TX utilization in-band, the Egress timestsamp is not mandatory but optional. HPCC++ algorithm today doesn't require Ingress Interface ID. P4.org INT defines Ingress and Egress Interface IDs as one metadata instruction. We keep the Ingress ID for a future use.

3. IANA Considerations

This document makes no request of IANA.

4. Acknowledgments

The authors would like to thank RTGWG members for their valuable review comments and helpful input to this specification.

5. Contributors

The following individuals have contributed to the implementation and evaluation of the proposed scheme, and therefore have helped to validate and substantially improve this specification: Pedro Y. Segura, Roberto P. Cebrian, Robert Southworth and Malek Musleh.

6. Security Considerations

TBD

7. Normative References

8. Informative References

[Zhu-SIGCOMM2015]
Zhu, Y., Eran, H., Firestone, D., Guo, C., Lipshteyn, M., Liron, Y., Padhye, J., Raindel, S., Yahia, M. H., and M. Zhang, "Congestion Control for Large-Scale RDMA Deployments", ACM SIGCOMM London, United Kingdom, August 2015.

[P4-INT] "In-band Network Telemetry (INT) Dataplane Specification, v2.0", February 2020, <https://github.com/p4lang/p4-applications/blob/master/docs/INT_v2_0.pdf>.

[RFC9179] "Data Fields for In Situ Operations, Administration, and Maintenance (IOAM)", May 2022, <https://datatracker.ietf.org/doc/html/rfc9197>.

[I-D.draft-gafni-ippm-ioam-additional-data-fields]
"Additional data fields for IOAM Trace Option Types", May 2021, <https://datatracker.ietf.org/doc/html/draft-gafni-ippm-ioam-additional-data-fields-00>.

[I-D.ietf-kumar-ippm-ifa] "Inband Flow Analyzer", February 2019, <https://tools.ietf.org/html/draft-kumar-ippm-ifa-06>.

[draft-miao-tsv-hpcc] Miao, R., "HPCC++: Enhanced High Precision Congestion Control", June 2022.

[SIGCOMM-HPCC]
Li, Y., Miao, R., Liu, H., Zhuang, Y., Fei Feng, F., Tang, L., Cao, Z., Zhang, M., Kelly, F., Alizadeh, M.,

and M. Yu, "HPCC: High Precision Congestion Control", ACM SIGCOMM Beijing, China, August 2019.

Authors' Addresses

Rui Miao
Alibaba Group
525 Almanor Ave, 4th Floor
Sunnyvale, CA 94085
United States of America

Email: miao.rui@alibaba-inc.com

Surendra Anubolu
Broadcom, Inc.
1320 Ridder Park
San Jose, CA 95131
United States of America

Email: surendra.anubolu@broadcom.com

Rong Pan
Intel, Corp.
2200 Mission College Blvd.
Santa Clara, CA 95054
United States of America

Email: rong.pan@intel.com

Jeongkeun Lee
Intel, Corp.
101 Innovation Dr
San Jose, CA 95134
United States of America

Email: jk.lee@intel.com

Barak Gafni
NVIDIA
350 Oakmead Parkway, Suite 100
Sunnyvale, CA 94085
United States of America

Email: gbarak@nvidia.com

Yuval Shpigelman
NVIDIA
Haim Hazaz 3A
Netanya 4247417
Israel

Email: yuvals@nvidia.com

Jeff Tantsura
Nvidia

Email: jefftant.ietf@gmail.com

Guy Caspary
Cisco Systems
Ofek 10 Building, 8 Hatochen Street
Caesarea Industrial Park 3079534
Israel

Email: gcaspary@cisco.com