Received: 4 May 2022 | Revised: 18 August 2022 | Accepted: 27 August 2022

DOI: 10.1002/itl2.393

LETTER

WILEY

An INT-based packet loss monitoring system for data center networks implementing Fine-Grained Multi-Path routing

Pilar Manzanares-López[1] | Lizhuang Tan[2] | Juan Pedro Muñoz-Gea[1] | Josemaría Malgosa-Sanahuja[1]

[1]Department of Information Technologies and Communications, Universidad Politécnica de Cartagena, Cartagena, Spain

[2]School of Electronics and Information Engineering, Beijing Jiaotong University, Beijing, China

Correspondence
Pilar Manzanares-López, Department of Information Technologies, and Communications, Universidad, Politécnica de Cartagena, Cartagena, Spain.
Email: pilar.manzanares@upct.es

Funding information
Spanish State Research Agency (AEI), under project grant AriSe2: FINe, (Ref. PID2020-116329GB-C22 founded by MCIN/AEI/10.13039/501100011033); Natural Science Foundation of Shandong Province, Grant/Award Number: ZR2020LZH010

Correction added on 18 January 2023, after first online publication: The copyright line was changed.

In-band network telemetry (INT) is a newer network measurement technology that uses normal data packets to collect network information hop-by-hop with low overhead. Since incomplete telemetry data seriously degrades the performance of upper-layer network telemetry applications, it is necessary to consider the own INT packet loss. In response, LossSight, a powerful packet loss monitoring system for INT has been designed, implemented, and made available as open-source. This letter extends the previous work by proposing, implementing, and evaluating LB-LossSight, an improved version compatible with packet-level load-balancing techniques, which are currently used in modern Data Center Networks. Experimental results in a Clos network, one of the most commonly used topologies in today's data centers, confirm the high detection and localization accuracy of the implemented solution.

KEYWORDS

data center networks, in-band network telemetry, loss detection, loss localization, multi-path routing

1 | INTRODUCTION

INT is a recent network measurement method that can quickly collect network status data to monitor the quality of networks and services, using normal data packets to carry the status information of switching devices, instead of using specifically dedicated packets. Numerous INT-based solutions have been proposed, offering ways of obtaining useful parameters.[1] In addition, advanced INT-based network controls provide network efficiency improvements, including congestion control, routing decisions, abnormal behavior detection, path tracing, and artificial intelligence-based network management. However, the impact of INT packet loss has not been considered. To fill that gap, we designed a packet loss monitoring system for INT called LossSight.[2]

One of the advanced pending works was to evaluate the packet loss detection and localization components of LossSight in data center networks (DCNs). DCNs are typically based on Clos and Fat-Tree topologies[3] with many equal-cost multiple paths. A variety of load balancing (LB) mechanisms has been deployed to take advantage of the parallel paths. The most commonly used LB scheme in data centers is ECMP (Equal-Cost Multi-Path)[4] which assigns traffic flows to different paths using hash-based scheduling, that is, without considering network state. However, to overcome the limitations of ECMP, many other LB schemes have been proposed. Splitting the flows into subflows and taking into account the network status are the most important aspects that have been considered in defining more efficient schemes. In recent years, granularity has been reduced to per-packet LB methods, with solutions such as RPS (Random Packet Spraying).[5]

In this letter, (a) we study the suitability of the detection and localization modules of the novel LossSight packet loss monitoring system in DCNs, identifying the problem introduced by packet-level LB schemes, (b) we use the potential of programmable data-plane P4-enabled switches[6] to propose a solution valid for the full range of LB schemes, (c) we implement and evaluate LB-LossSight in a Clos network proving the validity of the proposal.

This letter is organized as follows. Section 2 briefly describes LossSight, with a particular focus on the loss detection and localization module, and discusses the impact of LB schemes in that functionality. LB-LossSight is described in section 3. Section 4 presents the performance evaluation. Finally, section 5 concludes the paper.

2 | LOSSSIGHT AND IMPACT OF LOAD-BALANCING

The basic operation of INT is as follows (see Figure 1): When a data packet reaches the INT Source node, it inserts an INT header including the INT telemetry instruction and also the collected data in the INT metadata field. Then, the packet is forwarded to the next switching device, which adds its own INT metadata. After the packet has passed through all the INT Transit nodes, it reaches the INT Sink node which extracts all INT metadata, and sends the telemetry report to the Telemetry Server. This allows the Telemetry Server to collect telemetry metadata from the data plane, including ingress/egress metadata values at device-level.

LossSight[2] uses a marking strategy (Multi-bit Cycle Marking) to offer low-overhead INT packet loss detection and localization. MCM employs a LossBit (Lb) field. Each node maintains a counter for each flow, monotonically and cyclically increased. This value is inserted in the Lb field in each INT packet. The telemetry server maintains expected Lb values for each flow and, based on the received INT Reports, builds a LossBit received matrix which is used to detect and locate the loss of INT packets.

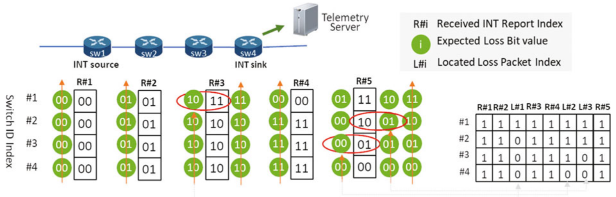
As can be seen in Figure 1, if the expected and received Lb values match (see R#1, R#2), this means there are no losses. However, if they do not match (see R#3), a loss has been detected. Comparing the received and expected values from the last (ending of the path) to the first (beginning of the path), it can be deduced that a loss happened between sw1 and sw2. After detecting a loss, it is crucial that the expected Lb values are updated adequately to represent the actual value of the counters at the nodes, in order to enable the proper detection and location of the following losses.

The increase of each flow counter in a node is independent of the other nodes' values and the value is only one, independently of the forwarding decision when a LB scheme is used. For these reasons, the above procedure becomes ineffective with packet-level LB solutions. As shown in Figure 2, where a packet-level LB scheme is executed, sw1 randomly routes the packets through sw2 and sw5. In this case, when R#4 is received, the Telemetry server detects that an INT Report loss has occurred, but is unable to determine the location of the loss. In fact, there are four possible loss locations, all of them compatible with the Lb values of R#4 (marked in the figure). This limitation affects the entire detection and localization process because, as explained, the process is based on the analysis and comparison of the received Lb values with the expected values. Therefore, the Telemetry Server needs to update the expected values adequately after detecting a packet loss to offer an accurate loss detection.

3 | LB-LOSSSIGHT: LOAD-BALANCING COMPATIBLE PROPOSAL

3.1 | Algorithm description

LB-LossSight adapts the making strategy to multi-path scenarios by defining of more than one marking counter per flow in the network devices, in particular, as many marking counters as possible output ports in the forwarding path choice. Algorithm 1



	R#1	R#2	L#1	R#3	R#4	L#2	L#3	R#5
#1	1	1	1	1	1	1	1	1
#2	1	1	0	1	1	1	1	1
#3	1	1	0	1	1	1	0	1
#4	1	1	0	1	1	0	0	1

FIGURE 1 LossSight: Loss detection and localization (Lb field = 2 bits). According to the R#3 received Lb values and the expected ones, an INT Report was received by sw#1 but not by sw#2, sw#3, and sw#4. L#1 is represented in the Loss Location Matrix and expected Lb Values are adequately updated. Similarly, L#2 and L#3 are located after the reception of R#5

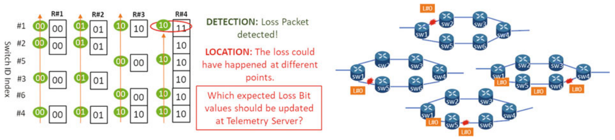

Switch ID Index
#1
#2
#5
#3
#6
#4
R#1
R#2
R#3
R#4
DETECTION: Loss Packet detected!
LOCATION: The loss could have happened at different points.
Which expected Loss Bit values should be updated at Telemetry Server?
L#0
sw1
sw2
sw3
sw4
sw5
sw6

FIGURE 2 Loss detection and localization problem of LossSight if a packet-level load-balancing (LB) solution is implemented

describes the detection and localization algorithm for a 4-ary Fat-Tree topology as shown in Figure 3. The algorithm can be adapted to other DCN topologies just by identifying the possible paths and, of course, it is compatible with single route topologies.

Algorithm 1. LB-Loss Detection and Localization

Input: lb: 1 (multi-rooted paths), 0 (one flow path); R: received INT report; network: network topology.
Output: Detection and localization result.
Executed after receiving an INT Report following the T1-A1-C1-A3-T4 path. Code similar for the other three paths.

```
1: path=obtain_path(R)
2: LbV=obtain_Lb(path)
3: expLbV=obtain_expLb(path)
4: if LbV==expLbV then
5:     update_expLbV(path)
6:     update_pending_values(path)
7: else
8:     if LbV[A3]!=expLbV[A3] then
9:         nr_losses=(LbV[A3]-expLbV[A3]+$2^{Lb}$)mod($2^{Lb}$)
10:        DetectedLocatedLoss(nr_losses, path, A3)
11:        if LbV[C1]!=expLbV[C1] then
12:            update_expLbValues(path)
13:        else
14:            update_expLbValues(path2)
15:    if LbV[C1]!=expLbV[C1] & pending_value(path[C1])!=0
       then
16:        update_UnLocatedLoss(path,C1)
17:        update_expLbValues(C1)
18:        update_pending_values(path,C1,A1)
19:    if LbV[C1]!=expLbV[C1] then
20:        nr_losses=(LbV[C1]-expLbV[C1]+$2^{Lb}$)mod($2^{Lb}$)
21:        DetectedLocatedLoss(nr_losses, C1)
22:        update_expLbValues(path)
23:    if LbV[A1]!=expLbV[A1] & pending_value(path[A1])!=0
       then
24:        update_UnLocatedLoss(path,A1)
25:        update_expLbValues(path)
26:        update_pending_values(path, A1)
27:    if LbV[A1]!=expLbV[A1] then
28:        nr_losses=(LbV[A1]-expLbV[A1]+$2^{Lb}$)mod($2^{Lb}$)
29:        DetectedLocatedLoss(nr_losses, A1)
30:        update_expLbValues(path)
31:    if LbV[T1]!=expLbV[T1] then
32:        nr_losses=(LbV[T1]-expLbV[T1]+$2^{Lb}$)mod($2^{Lb}$)
33:        if lb==1 then
34:            DetectedUnLocatedLoss(nr_losses, path2)
35:        else
36:            DetectedLocatedLoss(nr_losses, T1)
```

After receiving an INT report, the Telemetry server applies the proposed algorithm to monitor the transmission. As an example, consider that the received INT report followed the path T1-A1-C1-A3-T4. If the received Loss Bit values (LbV) correspond to the expected values (expLbV), the transmission takes place as expected (lines 4-6 in the algorithm). Otherwise, a loss has been detected, which can correspond to four possible situations:

1. The fourth. LbV (LbV[A3]) differs from the expected value (expLbV[A3]) (lines 8-14). That is, the loss packet reached A3 but was lost in the last forwarding. The loss is detected and localized appropriately and, in order to detect the following losses, the expected LossBit Values must be also updated adequately. In this case, the INT loss packet could have followed T1-A1-C1-A3 or T1-A1-C2-A3. Comparing LbV[C1] and expLbV[C1] it is possible to identify the path followed by the loss packet.
2. LbV[A3] and expLbV[A3] are equal, but the third. value (LbV[C1]) deviates from the expected value (expLbV[C1]) (lines 15-22). The difference between both values does not mean a loss directly. This difference could be motivated by a pending update of the expected LbV as a consequence of a previously detected but unlocated loss (explained in case d)). The loss was

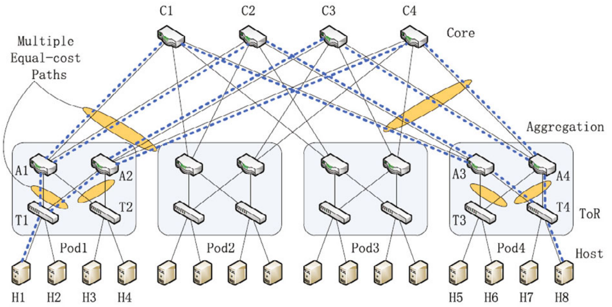


FIGURE 3 4-ary Fat-Tree Network Topology, from[7]

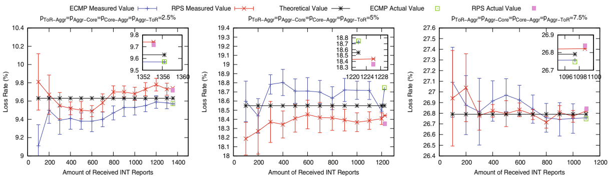


FIGURE 4 Detection accuracy: Measured Loss Rate obtained by LB-LossSight, Actual Loss Rate value (obtained directly from the switches' log files), and Theoretical value, considering loss rate between switches equal to 2.5%, 5%, and 7.5%

detected before, but the expected LbVs could not be updated. In this case, the location of that unlocated loss is identified now and the expected values can be updated adequately (lines 15-18). Otherwise, lines 19-22, a loss is detected and localized. The loss packet was forwarded by C1, but it did not reach A3.

3. The last two LbVs correspond to the expected ones, but the second. value (LbV[A1]) deviates from the expected value (expLbV[A1]) (lines 23-30). This case is similar to case b).
4. The last three LbVs correspond to the expected ones, but the first (LbV[T1]) deviates from the expected value (expLbV[T1]) (lines 31-36). Unlike the previous situations, the exact location of the loss cannot be inferred. The lost INT packet was sent by the T1, and the loss could occur at that point. However, the loss packet could also have followed the alternative route (T1-A1-C2-A3) and been lost in one of the following hops. Using the information available at this time, the loss is detected but the location of the loss cannot be directly inferred. For example, it is not possible to know whether the loss packet reached switches A1 and C2. If the loss happened in the last hop, C2 incremented its counter. Therefore, if the expected Lb value (expLbV[C2]) is not incremented, the next received INT Report of an INT packet passing through C2 will trigger a false INT packet loss detection. The same goes for A1. To address this problem, the proposed algorithm defines a status called "pending update" to each output port of a switch. This status is set if a packet might have been sent and lost, but it is uncertain. Thus, the pending values are taken into account when comparing expected and received LossBit values (line 15 and line 23 in Algorithm 1) in order to locate previously detected but unlocated losses.

The algorithm has a linear time complexity because it takes $O(k)$ basic operations to complete for an k-ary Fat-Tree topology. Regarding the implementation complexity, the proposed algorithm can be developed just using if/else commands, as well as subtractions, additions, rotations ($mod2^{Lb}$), and logical operators over the Lb bits.

3.2 | Implementation details in P4

LB-LossSight implements the INT loss monitoring solution with extremely low overhead. Each switch performs two main actions: the forwarding decision and the marking process. The forwarding decision is made in the ingress control flow of the switch, where the LB solution is implemented: the hash function and the random function offered by the P4 language are used to implement ECMP and RPS LB solutions. As a result, the output port is selected. The second action is performed in the egress control flow, where P4 registers are used to keep track of the marking counters used by the switch. As shown in Listing 1, the egress port selected by the LB scheme acts as a pointer to read/write the register used to manage the Lb values in the P4 switch.

```
Register<bit<2≫(size=NUM_PORTS) loss_counter;
Control MyEgress(...) {.
action add_swtrace(switchID_t swid) {
loss_counter.read(temp_Lb, [bit<32>] standard_metadata.egress_port);
hdr.mri.count=hdr.mri.count+1;
hdr.swtraces[0].swid=swid;
hdr.swtraces[0].bitvalue=temp_Lb;
if (temp_Lb<Lb) {temp_Lb=temp_Lb+1;}
else {temp_Lb=0;}
loss_counter.write((bit<32>standard_metadata.egress_port, temp_Lb);
}
table swtrace {.
actions={add_swtrace; NoAction;}
default_action=NoAction();
}
apply {if (hdr.mri.isValid()) {swtrace.apply();}}
}
```

Listing 1: Egress control flow of LB-LossSight

4 | PERFORMANCE EVALUATION

This section evaluates the performance of the detection and localization modules of LB-LossSight. ECMP and RPS load-balancing schemes are considered. The environment was built using the following software tools: Mininet, P4-Bmv2 switches, and client-server traffic application. The detection and localization software is implemented in Python. We built a 4-ary Fat-Tree topology as shown in Figure 3 to evaluate the loss detection and localization accuracy of LB-LossSight. The packet loss probability of links is defined in accordance with the values considered in[2]. The test results show the average value of 100

TABLE 1 Loss detection accuracy

	Loss Probability		
	2.5%	**5.0%**	**7.5%**
ECMP			
Theoretical value	9.631%	18.549%	26.791%
Actual value	9.572%	18.747%	26.840%
Measured value	9.566%	18.727%	26.757%
RPS			
Theoretical value	9.631%	18.549%	26.791%
Actual value	9.719%	18.353%	26.747%
Measured value	9.736%	18.442%	26.822%

TABLE 2 Loss localization accuracy (RPS scheme, $P = 5\%$)

	Loss INT Packets		
	Theoretical	**Actual**	**Measured**
ToR_1-Aggr_1	37.5	36.57	39.37
ToR_1-Aggr_2		36.39	39.32
Aggr_1-Core_1	17.8125	16.93	16.19
Aggr_1-Core_2		18.26	17.35
Aggr_2-Core_3		17.13	16.44
Aggr_2-Core_4		17.26	16.31
Core_1-Aggr_3	16.9219	16.39	16.23
Core_2-Aggr_3		16.82	16.76
Core_3-Aggr_4		16.36	16.60
Core_4-Aggr_4		17.41	17.06
Aggr_3-ToR_4	32.1516	32.43	32.39
Aggr_4-ToR_4		32.68	32.62

runs, with each run transmitting a flow of 1500 INT packets between hosts. MCM marking is implemented with a Loss Bit field of Lb = 2 bits.

First, we measure the accuracy of the loss detection by comparing the theoretical value of the loss rate (equation (1) for ECMP and (2) for RPS), the measured loss rate obtained by the LB-LossSight, and the actual loss rate in the experiments. As explained in section 2, LossSight solution is totally unsuitable for Load-Balancing topologies due to its inability of detecting INT loss packets in the presence of multi-paths. That is why LossSight is not considered in these experiments.

$$\mathrm{LR}_{\mathrm{ECMP}} = 1 - \left(1 - p_{T_1A_i}\right)\left(1 - p_{A_iC_j}\right)\left(1 - p_{C_jA_k}\right)\left(1 - p_{A_kT_4}\right) \tag{1}$$

for any of the equal-cost paths chosen by the ECMP path selection where $p_{T_1A_i}$ is the loss probability of the ToR-Aggr. link, $p_{A_iC_j}$ is the loss probability of the Aggr.-Core link, $p_{C_jA_k}$ is the loss probability of the Core-Aggr. link, and $p_{A_kT_4}$ is the loss probability of the Aggr.-ToR link.

$$\begin{aligned}\mathrm{LR}_{\mathrm{RPS}} = {} & \frac{1}{4}\left[1 - \left(1 - p_{T_1A_1}\right)\left(1 - p_{A_1C_1}\right)\left(1 - p_{C_1A_3}\right)\left(1 - p_{A_3T_4}\right)\right] + \frac{1}{4}\left[1 - \left(1 - p_{T_1A_2}\right)\left(1 - p_{A_2C_2}\right)\left(1 - p_{C_2A_4}\right)\left(1 - p_{A_4T_4}\right)\right] \\ & + \frac{1}{4}\left[1 - \left(1 - p_{T_1A_2}\right)\left(1 - p_{A_2C_3}\right)\left(1 - p_{C_3A_4}\right)\left(1 - p_{A_4T_4}\right)\right] + \frac{1}{4}\left[1 - \left(1 - p_{T_1A_2}\right)\left(1 - p_{A_2C_4}\right)\left(1 - p_{C_4A_4}\right)\left(1 - p_{A_4T_4}\right)\right]\end{aligned} \tag{2}$$

Figure 4 shows the results considering a loss probability of 2.5%, 5% and 7.5% at each point in the topology. In either case, the theoretical value of the network loss rate corresponds to the value obtained in a steady state. When the number of INT reports received is low, the test duration is far from the steady state and consequently, the measured values differ from the theoretical value (and have a larger confidence interval). As the transmission progresses, the actual value tends to the theoretical value and, thanks to the correct functioning of the LB-LossSight detection mechanism, so does the measured value. The actual value of the loss rate is obtained at the end of the run by analyzing the log information stored locally by each p4 switch. Table 1 allows us to compare the obtained values in more detail. As can be seen, although the measured value differs from the ideal theoretical value, it adequately reflects the actual value. As expected, due to Mininet behavior to emulate loss probability, the theoretical value and the measured and actual values differ more for lower values of loss probability in the topology.

Next, we evaluate the accuracy of loss localization by comparing the theoretical number of loss INT Reports at the different network points with the actual amount of losses and the losses localized by LB-LossSight. Again, the actual number of loss Reports at each point is obtained after analyzing the local log information of each switch in Mininet. Table 2 shows the results obtained when implementing the RPS with a loss probability of 5%. This is the more complex case because when using ECMP, all packets follow the same path, which simplifies the loss localization.

The detection and localization algorithm of LB-LossSight was developed specifically for this k-ary Fat-Tree topology. In this case, the difference between the Lb value of the first hop would result in immediate loss detection but inaccurate loss localization, but the subsequent use of the enabled state of the pending update allows the algorithm to determine the location of the detected loss in most cases. Although not shown due to space limitations, similar results are obtained for the other experiments.

5 | CONCLUSIONS

In this letter, we propose an improved version of the INT Report loss detection and localization algorithm in order to adapt LossSigth to modern Data Center Networks. The potential of programmable data plane devices and P4 language enable the implementation at network devices. Experimental results show high detection and localization accuracy even with packet-level load-balancing schemes such as RPS. LB-LossSight, which is totally compatible with non multi-path solutions, extends its use to a wider range of network scenarios.

6 | FUNDING INFORMATION

Spanish State Research Agency (AEI), under project grant AriSe2: FINe, (Ref.PID2020-116329GB-C22 founded by MCIN/AEI/10.13039/501100011033) and the project (Natural Science Foundation of Shandong Province under Grant No. ZR2020LZH010).

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Pilar Manzanares-López https://orcid.org/0000-0003-1296-7158

REFERENCES

1. Tan L, Su W, Zhang W, et al. In-band network telemetry: a survey. *Comput Netw*. 2021;186:107763. https://doi.org/10.1016/j.comnet.2020.107763
2. Tan L, Su W, Zhang W, Shi H, Miao J, Manzanares-Lopez P. A packet loss monitoring system for in-band network telemetry: detection, localization, diagnosis and recovery. *IEEE T Netw Serv Man*. 2021;18(4):4151-4168.
3. Al-Fares M, Loukissas A, Vahdat A. A scalable, commodity data center network architecture. *Comput Commun Rev*. 2008;38(4):63-74.
4. Hopps C. Analysis of an equal-cost multi-path algorithm. *RFC 2992, IETF*. 2000.
5. Dixit A, Prakash P, Hu YC, Kompella RR. On the impact of packet spraying in data center networks. *Proceedings IEEE INFOCOM*. 2013;2130-2138.
6. P4-16 Portable Switch Architecture (PSA), The P4.org Architecture Working Group, https://p4.org/p4-spec/docs/PSA.html
7. Hu J, Ruan C, Wang L, Alfarraj O, Tolba A. Coding-based distributed congestion-aware packet spraying to avoid reordering in data center networks. *IEEE Access*. 2021;9:35539-35548.

How to cite this article: Manzanares-López P, Tan L, Muñoz-Gea JP, Malgosa-Sanahuja J. An INT-based packet loss monitoring system for data center networks implementing Fine-Grained Multi-Path routing. *Internet Technology Letters*. 2023;e393. doi: 10.1002/itl2.393