Computer Networks 186 (2021) 107763

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Survey paper

In-band Network Telemetry: A Survey

Lizhuang Tan [a,b,c], Wei Su [a,b,*], Wei Zhang [c], Jianhui Lv [d], Zhenyi Zhang [a,b], Jingying Miao [a,b], Xiaoxi Liu [a,b], Na Li [e]

[a] School of Electronics and Information Engineering, Beijing Jiaotong University, Beijing 100044, PR China
[b] National Engineering Laboratory of Next Generation Internet Interconnection Devices (NGIID), Beijing Jiaotong University, Beijing 100044, PR China
[c] Shandong Provincial Key Laboratory of Computer Networks, Shandong Computer Science Center (National Supercomputer Center in Ji'nan), Qilu University of Technology (Shandong Academy of Sciences), Ji'nan 250000, PR China
[d] International Graduate School at Shenzhen, Tsinghua University, Shenzhen 518057, PR China
[e] Shandong Branch of National Computer network Emergency Response technical Team/Coordination Center (CNCERT/SD), Ji'nan 250002, PR China

ARTICLE INFO

Keywords:
Network telemetry
In-band network telemetry
Network measurement
Network management
Programmable data plane
Software-defined network

ABSTRACT

With the development of software-defined network and programmable data-plane technology, in-band network telemetry has emerged. In-band network telemetry technology collects hop-by-hop network status information through business packets to achieve end-to-end visualization of network services. In-band network telemetry uses the data plane to directly drive the network measurement process, subverting the research idea of traditional network measurement that treats network switching device as an intermediate black box. In-band network telemetry technology has the advantages of flexible programming, strong real-time, less noise and path-level network status perception, etc. It has become an emerging representative of network telemetry technology and has received extensive attention from academia and industry.

In this survey, we conduct a comprehensive investigation on the situation of in-band network telemetry technology. Firstly, we review the different development stages of network measurement and give a chronology. And we sort out the development history and research results of in-band network telemetry. Secondly, we introduce in detail several existing representative solutions of in-band network telemetry, including INT, IOAM, AM-PM and ANT, in terms of systems, implementation idea, technical features, standardization, and research results. Thirdly, we analyze several key technologies of the in-band network telemetry system, which runs through data generation, export, storage, and analysis. We summarize the advantages and disadvantages of the existing solutions. Moreover, we investigate the latest applications of in-band network telemetry from two aspects: performance measurement and function measurement. Furthermore, we highlight the technical opportunities brought by in-band network telemetry technology to network measurement and management, as well as several technical challenges and future research directions.

1. Introduction

Network measurement is the foundation of network management and control [1]. Since the Natural Science Foundation (NSF) of the United States began to fund Internet measurement in 1995, academia and industry have proposed and improved many measurement methods. As shown in Fig. 1, according to the development of measurement methods, we can divide network measurement into three research ideas: (1) traditional network measurement developed since 1995; (2) software-defined measurement that was produced and developed with software-defined networks (SDN) since 2008; (3) network telemetry with the rise of programmable data plane (PDP) technology since 2015.

1.1. Traditional network measurement

According to RFC 7799[2], traditional network measurement is divided into active, passive and hybrid network measurement.

The research idea of active network measurement is to deduce the overall performance status of the network by constructing, observing and analyzing the operation status of the probe packet on the network. Representative schemes are Ping [3] and Traceroute [4]. The forwarding path of the out-of-band probe constructed by the active measurement scheme may not be exactly the same as the service traffic flow, so the measurement results cannot accurately reflect the

* Corresponding author at: School of Electronics and Information Engineering, Beijing Jiaotong University, Beijing 100044, PR China.
E-mail addresses: lzhtan@bjtu.edu.cn (L. Tan), wsu@bjtu.edu.cn (W. Su), wzhang@sdas.org (W. Zhang), lvjianhui2012@sz.tsinghua.edu.cn (J. Lv), 19120173@bjtu.edu.cn (Z. Zhang), 19120098@bjtu.edu.cn (J. Miao), 20120074@bjtu.edu.cn (X. Liu), lina@cert.org.cn (N. Li).

https://doi.org/10.1016/j.comnet.2020.107763
Received 7 August 2020; Received in revised form 13 December 2020; Accepted 22 December 2020
Available online 26 December 2020

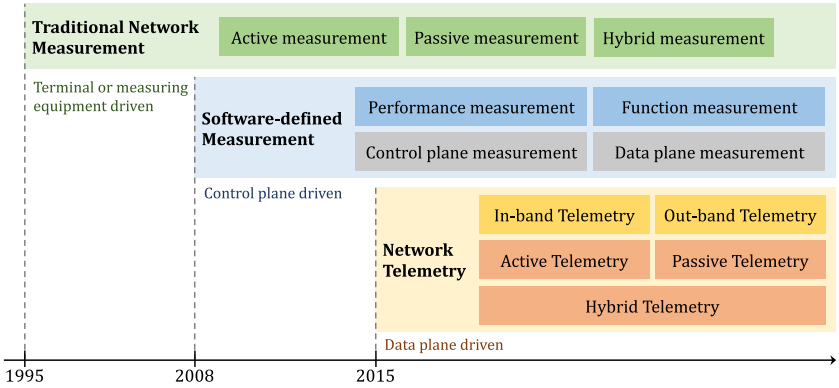


Fig. 1. Development stages and classifications of network measurement.

actual network performance. The passive measurement schemes derive and analyze the network traffic operation status through traffic mirroring and proxy reporting, represented by NetFlow [5], sFlow [6] and IPFIX [7]. The passive measurement schemes are limited by the switch performance, usually use sampling and compression mechanisms, which will sacrifice measurement accuracy [8]. Hybrid measurement can obtain more satisfactory measurement results by combining active and passive measurement methods or redesigning the measurement mechanism in combination with the advantages of active and passive measurement methods.

Because of its simplicity of deployment, traditional network measurement methods are widely used in the field of network management. Traditional network measurement schemes inevitably have "observer effect" and "inaccurate measurement effect", which means that the network measurement process itself affects the network state inevitably, and the network measurement mechanism itself is flawed, resulting in inaccurate measurement results [9].

1.2. Software-defined measurement

With the emergence of SDN [10] and PDP [11], more open control plane and data plane programming capabilities have reshaped the traditional network measurement. SDN reconstructs the network control plane and data plane without changing the foundation of the IP network. On one hand, it greatly improves the flexibility of the network measurement architecture. On the other hand, the flow collection and sampling methods [12] used in traditional network measurement techniques have been preserved. PDP technology enables network administrators to change the switch's message processing logic, implement new functions and new protocols. PDP offloads many network measurement applications from terminals or measurement servers to the programmable data plane, thereby realizing more direct and effective network measurement. A number of software-defined measurement schemes have been proposed in academia and industry, which have supported network performance measurements and function measurements such as available bandwidth [13], packet loss [14], throughput [15], latency [16], path tracing [17], data plane rule consistency verification [18], long flow detection [19], fault location [20], etc. Compared to traditional network measurement, software-defined measurement has great advantages in openness, transparency, and programmability.

1.3. Network telemetry and in-band network telemetry

Network telemetry [21] has emerged as a mainstream technical term to refer to the newer network data collection and consumption techniques. Telemetry is an automated process for remotely collecting and processing network information. Network telemetry has been widely considered as an ideal mean to gain sufficient network visibility with better scalability, accuracy, coverage, and performance than traditional network measurement technologies.

In-band network telemetry is an emerging representative of network telemetry, which has received extensive attention in both academia and industry in recent years. Different from the traditional network measurement and software-defined measurement, in-band network telemetry combines data packet forwarding with network measurement. In-band network telemetry collects the network status by inserting metadata into packet by switching nodes.

In-band network telemetry data and user data usually share the same link or even the same packet. In fact, the measurement idea of in-band network telemetry already proposed in early 2000. Yaar et al. [22] proposed a datagram marking mechanism Pi for DDoS attack detection in 2002. In Pi, each router on the path adds a router ID (2 bits) to the packet header to detect packet forwarding path changes. In the same year, Katabi et al. [23] designed the XCP congestion control protocol. Each XCP packet header carried a congestion value field H_feedback that is initialized by the sender and can be changed by the router along the way. The router calculated and fills in the feedback congestion value hop by hop. The receiver extracted the congestion header and sends it back to the sender in the ACK message. The sender adjusted the congestion window based on the congestion feedback value. Luckie et al. [24] proposed the IP Measurement Protocol (IPMP) in 2004, which captured path information by adding a hop-by-hop path record through the IPMP-enabled routers in the network. For example, in order to calculated end-to-end delay, IPMP used the path records represented by three fields TTL, Timestamp and IPv4 address in IPMP echo packet. At the same time, Mahajan et al. [25] designed and proposed Tulip, which was used to diagnose packet reordering, packet loss, and queuing delay. In Tulip, the packet header was developed with five fields: Path signature, Sample TTL, Timestamp, Counter, and Interface id for storing necessary path information. Because the traditional network data plane had the characteristic of protocol correlation in forwarding processing, the measurement protocol above did not actually deployed. Inspired by the above research, Pezaros et al. [26] proposed an on-path measurement scheme, namely In-line, based on IPv6 extension header. In In-line, business packets carried richer measurement commands and measurement data. In 2010, Pezaros et al. [27] proposed In-band Performance Measurement (IPM), and implemented end-to-end delay, packet loss, and traffic burst measurements at the rate of 10 Gb/s based on FPGA hardware, verifying that the use of filtering, aggregation and sampling can reduce the impact of IPM on network forwarding performance.

The most typical research results of in-band network telemetry are In-band Network Telemetry (INT) [28], In situ Operation Administration and Maintenance (IOAM) [29], Alternate Marking-Performance Measurement (AM-PM) [30] and Active Network Telemetry (ANT) [31].

To sum up, in-band network telemetry uses business packets to carry telemetry instructions or telemetry data, and the source node embeds telemetry tags or instructions in the business packets to indicate the network information that needs to be measured. The data plane fills the telemetry information into the business packet while matching and forwarding. Last hop reports all telemetry data to the telemetry server. In-band network telemetry changes the network measurement process from control plane/management plane driven to data-plane driven. Using in-band network telemetry, network administrators can directly capture transient problems from the data plane caused by performance bottlenecks, network failures or misconfigurations.

1.4. Related research institutions

Because network telemetry has very broad theoretical research value and engineering application prospects in network management, many international organizations, research institutions and universities are engaged in network telemetry. At present, active research institutions include Internet Engineering Task Force (IETF), Open Network Foundation (ONF) and Barefoot Networks.

The IETF has established working groups such as IPPM [32], IPFIX [33], OPSA [34], RMONMIB [35], PSAMP [36] and IOAM [37] to study related issues of network measurement. Among them, IETF IPPM WG [32] and OPSA WG [34] are working on the standardization of IOAM [29], PBT [38], iFIT [39] and AM-PM [30].

The ONF focuses on software-defined network measurement and in-band network telemetry. The current projects related to network telemetry includes NG-SDN, P4[40], Information Model and INT [41].

Barefoot Networks is a high-performance programmable network solution provider. It is committed to the development of high-speed programmable Ethernet switch chips. It has launched Deep Insight [42], a packet-by-packet network monitoring application based on the Tofino forwarding chip and P4 language [43].

In addition, scientific research institutions such as Harvard [21], Stanford [44], POSTECH [45], Tsinghua [46], CAS [47], USTC [48], BUPT [17], Cisco [49], Broadcom [50], Marvell [51], Arista [52], Mellanox [53], Juniper [54], VMware [55], Alibaba [56], and Huawei [57] have carried out research work on network telemetry technology.

1.5. Organization of this survey

Although there are a number of research reviews on network measurement [1,58–60], a survey on in-band network telemetry technology is not yet available. This survey has combed the background, research status, applications and technical opportunities of in-band telemetry technology, paving the way for further research on related issues. The organization of this survey is as follows: Section 2 outlines some representative schemes of in-band network telemetry in the data plane. Section 3 collates and analyzes some key technologies of in-band network telemetry systems. Section 4 summarizes typical applications based on in-band network telemetry. Section 5 discusses the problems and challenges in the development of in-band network telemetry technology, and analyzes future research directions. For a better understanding of the structure and organization of this survey, we refer the reader to Fig. 2. Table 1 provides a list of commonly used acronyms in this survey.

Table 1
A list of commonly used acronyms in this survey.

Abb.	Definition
AM-PM	Alternate Marking-Performance Measurement
ANT	Active Network Telemetry
BMv2	Behavioral Model
gRPC	Google Remote Procedure Call
INT	In-band Network Telemetry
INTO	In-band Network Telemetry Orchestration
IOAM/In situ OAM	In situ Operation Administration and Maintenance
IPFIX	IP Flow Information Export
KDN	Knowledge-defined Network
MTU	Maximum Transmission Unit
NETCONF	Network Configuration Protocol
OVS	Open vSwitch
P4	Programmable Data Plane
PBT	Postcards-Based Telemetry
PDP	Programmable Data Plane
PISA	Protocol Independent Switch Architecture
POF	Protocol Oblivious Forwarding
RCP	Rate Control Protocol
SDM	Software-defined Measurement
SDN	Software-defined Network
SFC	Service Function Chain
SR	Segment Routing
VNF	Virtual Network Function
VPP	Vector Packet Processing

2. Representative schemes for in-band network telemetry

Currently, data plane in-band network telemetry research includes INT [28] led by P4.org and IOAM led by the IETF IPPM working group. INT focuses on how to use programmable data planes to implement path-level network measurement, and proposes basic implementation ideas [61]. IOAM [62] is promoted by the IETF, which conducts research and standardization on the architecture and protocol of in-band network telemetry. Many RFC documents are being formed and have received extensive attention from equipment manufacturers. Compared with INT and IOAM, AM-PM and ANT are more flexible and easy to deploy.

It should be pointed out that the generalized in-band network telemetry is a general term that includes the four representative schemes described in this Section. The narrow in-band network telemetry specifically represents the P4 language-based INT described in Section 2.1. In order to distinguish the two, this survey uses the full name (In-band network telemetry) to represent the former, and the abbreviation (INT) to represent the latter.

2.1. INT

INT [28] was jointly proposed by Barefoot, Arista, Dell, Intel and VMware in 2015. It is a framework that does not require the intervention of network control plane, and collects and reports network status by network data plane. In the INT architecture, the switching device forwards and processes packets which carry telemetry telemetry instructions. When telemetry packets pass through the device, the telemetry instructions instructs the INT device to collect and insert network information.

In 2014, inspired by the active network [63], Jeyakumar et al. [64] designed the programming interface, namely TPP, for the exchange of information between switches and data packets. The terminal can send a customized TPP packet to achieve congestion control, load balancing, network diagnostics and network performance monitoring. In terms of protocol design, TPP adds Control, Instructions and Memory fields. The Control field describes the length of the TPP instruction to be inserted, so that the switch can process it. Instructions field indicates what switch should do after receiving the TPP packet.

Instructions include four operations, they are Insert (LOAD), Read (STORE), Conditional Store and Execute (CSTORE), and Conditional Perform (CEXEC). Among them, the Insert instruction instructs the switch to fill the state value into the packet, the Read instruction instructs the switch to read the state value from the packet.

The Memory field is responsible for storing switch status information. This is an earlier paper to implement network measurement directly through the programmable data plane.

In 2015, Kim et al. [28] proposed an INT technology based on programmable data plane, and provided a demonstration of HTTP instantaneous delay measurement. Combining path marking and queue

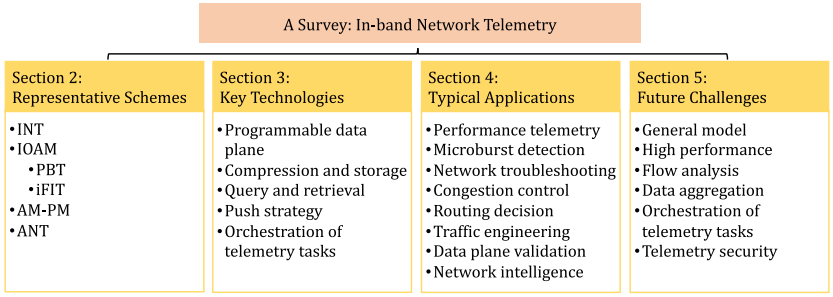


Fig. 2. Structure of this survey.

Table 2
INT term and description.

Terms	Description
INT Header	The header information in the telemetry packet that indicates the content of the telemetry
INT Packet	Data packets carrying telemetry information
INT Metadata	Telemetry data package to collect telemetry data
INT Instruction	Instructions to declare the INT metadata items to be collected by telemetry data packets
INT Source Node	The startpoint entity to insert INT Headers
INT Sink Node	The endpoint entity to extract INT information
INT Transit Hop	The intermediate entity to insert INT Metadata
Telemetry Server	A telemetry server that receives and processes telemetry data

delay information in INT messages, the authors observed an occasional steep increase in HTTP request latency due to bottleneck switch queue occupancy contention.

Based on previous research, the P4.org defined the INT data plane specification [61], giving the terms of the INT system, the telemetry metadata specification and INT encapsulation and implementation examples based on VXLAN, Geneve, NSH, TCP, UDP and GRE protocol.

As shown in Table 2, the INT forwarding plane consists of three trusted entities: INT Source Node, INT Sink Node and INT Transit Hop. INT Source Node is responsible for embedding telemetry instruction into the normal business packets or telemetry business packets. INT Sink Node extracts and reports the telemetry results. INT Source Node and INT Sink Node can be network applications, terminal network protocol stacks, network management programs, NICs, and sender/receiver of ToR switches. INT Transit Hop only needs to fill in telemetry metadata according to the instruction of INT packet.

As shown in Fig. 3, the INT system consists of a telemetry server and some switches that support INT. According to the needs of the actual telemetry tasks, the system may also need other devices such as a time synchronization server to complete auxiliary work.

The P4-based INT [65] is the earliest INT implementation scheme. On this basis, the academics have successively proposed INT scheme for application scenarios such as wireless networks, packet-optical networks, and new data plane based on protocol-independent forwarding.

Due to hierarchical architecture and lack of switch programmability, INT cannot be directly deployed in packet-optical network. Anand et al. [66] designed an intent-driven INT framework POINT. The POINT node first converts the intent of the telemetry command and parses it into a specific telemetry requirement related to the current device. Then, POINT inserts telemetry metadata into the business packet to realize the end-to-end status collection. POINT uses Intent Type and CiroupiD to represent telemetry intent, in which 13-bit Intent Type can represent 8000 different telemetry intent combinations, and 8-bit CiroupiD can encode 256 switching devices.

Gulenko et al. [67] presented a prototypical implementation of INT in Open vSwitch and evaluated the prototype regarding the expressiveness of the collected data and performance overhead. The experiments show that when the int_transit is not triggered, there is no measurable CPU overhead, which means that the performance of the virtual switch is not affected when the INT function is not used. When enabling the int_transit action, the CPU overhead of the kernel module was measured around 0.3% for 1 MByte/s and 1% for 100 MByte/s for the commodity hardware CPU. The overhead scaled sub-linearly for the intermediate throughputs.

Karaagac et al. [68] extended INT from wired network to wireless network, and proposed “INT in 6TiSCH” for industrial wireless sensor networks. INT in 6TiSCH is designed to minimize resource consumption and communication overhead. It is suitable for scenarios such as abnormal monitoring, congestion control, centralized routing and scheduling management.

As shown in Fig. 4, INT in 6TiSCH inserts telemetry data in the Information Elements (IEs) field of the IEEE 802.15.4 MAC frame. The telemetry message consists of three parts: IEs Subtype ID, INT Header and INT Content.

INT Header is composed of INT Control header (8 bits), Sequence Number (8 bits) and Bitmap (8 bits). INT Control header is used to define the telemetry mode and telemetry function. Sequence Number is used to distinguish different INT telemetry data from the same node. Each bit in Bitmap represents a predefined telemetry data.

INT Control header defines three telemetry modes through HBH Mode (2 bits). When Mode = 00, all intermediate nodes participate in telemetry. When Mode = 10, intermediate nodes will randomly add telemetry data. When Mode = 01, intermediate nodes calculate the probability value based on the time when the telemetry data was added last time, the length of the forwarded frame and the number of remaining hops, and optionally add the telemetry data based on the probability value. The setting of HBH Mode can reduce the telemetry bandwidth overhead.

Bitmap Mode defines two bitmap modes: Content Bitmap and Node Bitmap. In Content Bitmap mode, intermediate node will add telemetry data according to the telemetry combination specified by the Bitmap field. In Node Bitmap mode, intermediate nodes are allowed to add custom bitmaps and INT data. The base telemetry data model defined by INT in 6TiSCH includes: Node ID (2 Bytes), Receive Channel & Timestamp (2 Bytes), Utilization indicator (1 Byte), RSSI (1 Byte) and other types of retained information.

Tang et al. [69] implemented a real-time programmable and selectable INT solution Sel-INT on the software switch Open vSwitch [70] based on the Protocol Oblivious Forwarding (POF). Compared with software switches such as PISCES [71] and BMv2[72] that do not support dynamic compilation of Pipeline process, Sel-INT allowed real-time compilation, and supports dynamic adjustment of telemetry instructions, telemetry sampling rate and other telemetry behaviors.

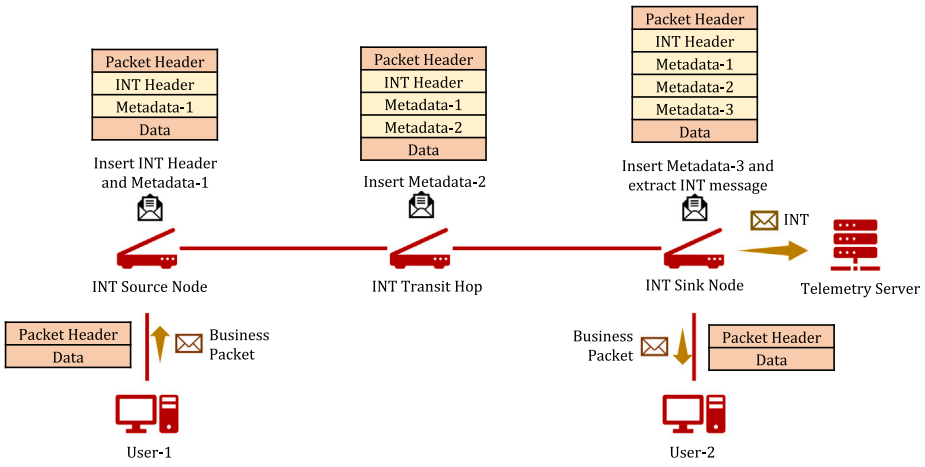


Fig. 3. INT process.

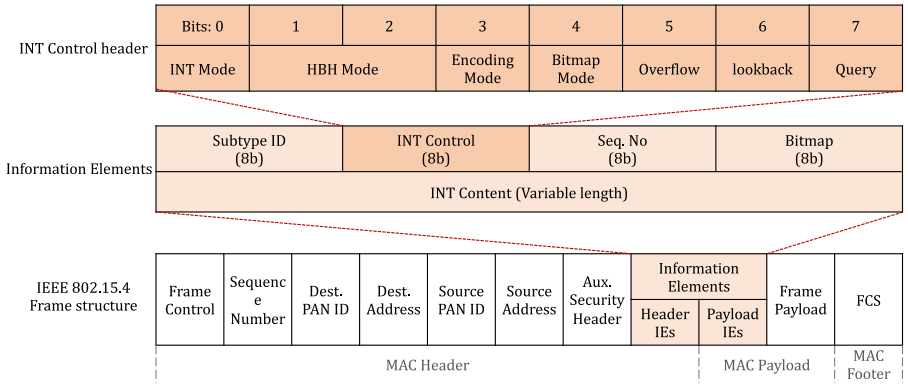


Fig. 4. Frame structure of INT in 6TiSCH.

Niu et al. [73] designed a P4-based flexible multilayer INT (ML-INT) system to visualize an IP-over-optical network in real time. ML-INT selected a small portion of packets in an IP flow to encode INT header, while each INT header only contains a part of the statistics of all the electrical/optical network elements on the flow's routing path.

In summary, INT has some inherent disadvantages. First, the payload ratio of the normal packet is reduced due to the encapsulation of telemetry instruction and metadata. Secondly, the construction, encapsulation, filling and extraction of telemetry instruction and metadata increase the processing burden of switch. In addition, INT only exports telemetry data at INT Sink, which may lead to loss of telemetry data due to packet loss in network. Lastly, INT cannot measure and diagnose packet loss.

2.2. IOAM

OAM [74] stands for operation, management, and maintenance, and refers to the set of tools used for performance measurement, fault detection, and troubleshooting. In situ operation administration and maintenance (IOAM) is a technology to record operational information in the packet while the packet traverses a path between two points in the network [62]. IOAM is to complement current out-of-band OAM mechanisms based on ICMP or other types of probe packets. IOAM entity includes encapsulating node, decapsulating node and transit node. IOAM can implement complex OAM functions such as path tracing, path verification and SLA verification.

IOAM has the following capabilities:

1. A flexible data format to allow different types of information to be captured as part of an IOAM operation, including not only path tracing information, but additional operation and telemetry information such as timestamps, sequence numbers, or even generic data such as queue size, geo-location of the node that forwarded the packet, etc.
2. A data format to express node as well as link identifiers to record the path that a packet takes with a fixed amount of added data.
3. The ability to actively process information in the packet, for example to prove in a cryptographically secure way that a packet really took a pre-defined path using some traffic steering method such as service chaining or traffic engineering.
4. The ability to include OAM data beyond simple path information, such as timestamps or even generic data of a particular use case.

As shown in Table 3, a detailed description of IOAM concepts [62], data-formats [75–78], encapsulations [79–83], exporting [84,85] and securely proof [86]can be found in IETF internet drafts.

The IOAM is implemented as a plugin in Cisco Vector Packet Processing (VPP) program [49], Linux Kernel [87] and OpenDaylight [88].

However, IOAM has similar deficiencies as INT:

1. IOAM's hop-by-hop telemetry mechanism generates a large amount of telemetry data. IOAM lacks filtering, deduplication, aggregation and compression operations on telemetry data.
2. IOAM encapsulation nodes and decapsulation nodes need to insert and extract telemetry data packets. IOAM transit nodes need to constantly check the position of the data packet flag, insert telemetry data into the OAM field and recalculate the checksum. The processing performance of switching devices in the IOAM domain has become a potential network performance bottleneck.

Table 3
IETF internet drafts related to IOAM.

IETF Internet Draft	Descriptions
Data Fields for In situ OAM [62]	Discusses the data fields and associated data types for in situ OAM.
In situ OAM IPv6 Options [75]	Outlines how IOAM data fields are encapsulated in IPv6.
In situ OAM Flags [76]	Presents new flags in the IOAM Trace Option headers, such as Loopback and Active flags.
In Situ OAM Profiles [77]	Introduces the concept of use case-driven IOAM profiles.
A YANG Data Model for In-Situ OAM [78]	Defines a YANG module for the IOAM function.
Network Service Header (NSH) Encapsulation for In situ OAM (IOAM) Data [79]	Outlines how IOAM data fields are encapsulated in the Network Service Header (NSH).
MPLS Data Plane Encapsulation for In situ OAM Data [80]	Defines how IOAM data fields are transported using the MPLS data plane encapsulation.
VXLAN-GPE Encapsulation for In situ OAM Data [81]	Outlines how IOAM data fields are encapsulated in VXLAN-GPE.
Geneve encapsulation for In situ OAM Data [82]	Outlines how IOAM data fields are encapsulated in Geneve.
EtherType Protocol Identification of In situ OAM Data [83]	Defines an EtherType that identifies IOAM data fields as being the next protocol in a packet, and a header that encapsulates the IOAM data fields.
In situ OAM raw data export with IPFIX [84]	Discusses how IOAM information can be exported in raw format from network devices to systems using IPFIX.
In situ OAM Direct Exporting [85]	Introduces the Direct Export (DEX) option for IOAM data to be directly exported without being pushed into in-flight data packets.
Proof of Transit [86]	Defines mechanisms to securely prove that traffic transited said defined path.

3. IOAM also faces the limitation of packet maximum transmission unit (MTU) caused by long telemetry path. The actual number of telemetry hops may have a upper limit.
4. IOAM is sensitive to packet loss and cannot solve the problem of missing telemetry data due to packet loss.

In order to reduce IOAM data plane processing overhead and business packet length limitation, Song et al. [89] proposed Postcards-Based Telemetry (PBT) based on IOAM, which realized telemetry by marking business packets or inserting telemetry instruction into business packets without carrying telemetry data. The mark scheme was called PBT-M, and the instruction insertion scheme was called PBT-I.

As shown in Fig. 5, in PBT-M, a business packet enters OAM domain, and Head Node marks the packet, generates a telemetry data packet and exports it to the telemetry server. When the marked business packet passes through Path Node, the telemetry information will be uploaded immediately. Finally, End Node removes the mark and restores it to a normal business packet.

The mark bit of PBT-M is only 1 bit usually, which cannot represent telemetry instruction. The data plane device must rely on controller or telemetry server to configure the telemetry instruction template. In response to this problem, the PBT-I scheme continues to use the telemetry instruction field of IOAM. The Head Node inserts a telemetry instruction into the business packet to instruct Path Nodes what telemetry information to collect. However, both PBT-M and PBT-I do not use business packets to carry telemetry data. PBT-I is a compromise scheme between IOAM and PBT-M, combining the advantages of both. The advantages of PBT over IOAM include:

1. PBT exports telemetry information hop by hop, so that telemetry is not restricted by packet MTU.
2. PBT reduces the processing complexity of data plane for telemetry data packets.

At the same time, the mechanism of PBT hop-by-hop export of telemetry information may lead to problems such as telemetry bandwidth overhead and performance degradation of telemetry server. To address these shortcomings, Lu et al. [39] proposed the telemetry framework iFIT, which integrated the advantages of IOAM and PBT, added mechanisms such as adaptive telemetry packet sampling, non-time-sensitive telemetry data export, and event-triggered anomaly detection. It improved the availability and scalability of the PBT telemetry system.

Ballamajalu et al. [90] proposed an in situ network telemetry mechanism, named Co-iOAM, for monitoring and troubleshooting 6LoWPAN/LPWAN network. The advantages of Co-iOAM included flexibility, lightweight operation, cross-layer telemetry and optimized representation.

2.3. AM-PM

The alternate marking-performance measurement (AM-PM) [30] is a hybrid telemetry for end-to-end network packet loss and delay measurement, which has the advantages of flexible deployment and high statistical accuracy. In AM-PM, each packet header contains a binary marking field, Marking Bit, whose value is "0" or "1". The marking bit decomposes the flow into consecutive blocks of packets, which are used to synchronize and coordinate measuring events between the two measurement checkpoints. AM-PM are being considered in the context of various encapsulations, including Geneve, SFC NSH, BIER, MPLS, and QUIC. Notably, AM-PM can also be applied over existing network protocols such as IP. AM-PM is under discussion in six working groups in the IETF, and is being pursued in several Internet drafts, written by over 20 different authors, indicating the wide interest in AM-PM from various vendors and operators.

As shown in Fig. 6(a) and (b), AM-PM proposes two detection synchronization methods for marking [91]: Pulse Detection and Step Detection. The detection point uses the marking bit field of adjacent data packets to achieve single-point pulse marking or alternate step marking to complete the binding and notification of telemetry events.

AM-PM includes two alternate marking methods: double marking and multiplexed marking. Double marking is the most primitive AM-PM marking scheme, which uses the color bit and delay bit of each telemetry packet between measurement points to achieve packet loss and delay measurement. As shown in Fig. 6(c), the color bit adopts the step detection method, and the delay bit adopts the pulse detection method. The color bit is switched periodically to define different time intervals. The measurement points maintain their own color bit counters. By comparing the color bit counter values among different measurement points, the telemetry server calculates the packet loss rate between the two measurement points. Delay bit is used to mark a specific data packet used for delay measurement. The measurement point records and reports the current local timestamp after receiving the data packet by the delay bit. The telemetry server calculates the delay information by comparing the timestamps of the same telemetry packet at different measurement points. In addition, since color bit adopts the step detection, packet disorder will not affect the telemetry performance. As shown in Fig. 6(d), the multiplexed marking method only uses 1-bit to measure the packet loss and delay by performing an exclusive OR operation on the color bit and the delay bit. AM-PM

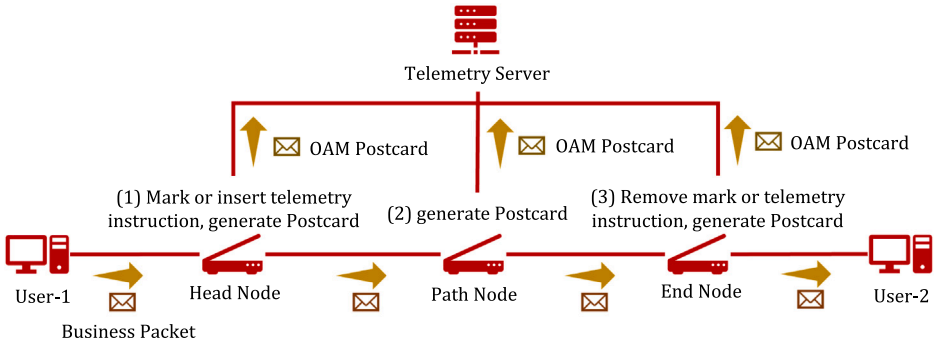


Fig. 5. PBT forwarding process.

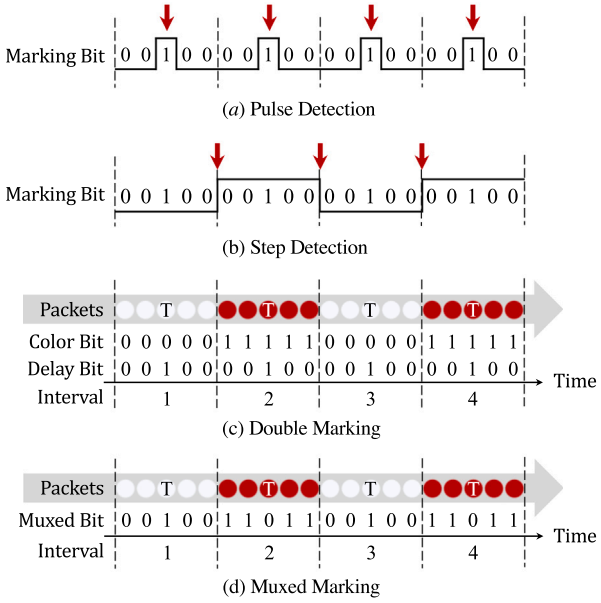


Fig. 6. AM-PM detection and marking.

needs to maintain two counters for delay and packet loss measurement. An AM-PM telemetry system requires 2 mn packet counters with m measurement points and n monitored flows.

Riesenberg et al. [92] built an AM-PM prototype system based on Marvell Prestera chip and P4 programming language. The verification results showed that the double-mark measurement delay error was less than 100 ns, and the statistical error of the packet loss rate was less than 10^{-6}.

Karaagac et al. [93] proposed a monitoring solution using the AM-PM for end-to-end and hop-by-hop reliability and delay performance measurement of Industrial Wireless Sensor Networks (IWSNs).

Fang et al. [56] built an automated diagnosis system VTrace, which solved the problem of continuous packet loss in cloud networks. The core idea is "task-matching-coloring-collection-analysis". Combined with big data analysis technology, VTrace automatically analyzed end-to-end traffic fluctuations in the cloud network and determine the exact cause and solution.

AM-PM uses the information convention of neighboring packets to reduce the amount of telemetry data that the data packet itself needs to carry, and trades higher processing complexity for lower storage overhead. Its advantage is to avoid data packet length overload and reduce the calculation amount of data plane; the disadvantage is that the telemetry efficiency is low, and the telemetry time accuracy is the smallest step, which is often greater than the interval between adjacent data packets.

2.4. ANT

Active network telemetry (ANT) is a telemetry mechanism developed on the basis of active measurement. The basic idea is to actively construct a telemetry probe to traverse the required telemetry path.

EverFlow [94], a Microsoft data center telemetry system, proactively constructs telemetry packets that are identical to the packets with network failure. After the virtual network device detects the telemetry data packet, it uploads status information to the SDN controller. The control plane locates faults by analyzing status information. EverFlow is a real-time telemetry scheme that can only detect the network failure status that telemetry packets are experiencing, and it is difficult to find occasional network failures.

Liu et al. [46] proposed the concept of network telemetry as a service, and designed an active network telemetry platform NetVision. NetVision proactively sends probe packets in an appropriate number and format to match the network status and telemetry tasks, thereby reducing telemetry overhead and increasing telemetry coverage and scalability. NetVision uses segmental routing for route control, customizing detection path by changing SR tags. The NetVision probe has a "SR+INT" dual-stack structure. The SR stack includes the length of the SR list and the output port label, and the INT stack includes the length of the label list and the telemetry label list. The router completes a telemetry forwarding by popping the SR label and placing the INT label.

Pan et al. [31] combined in-band network telemetry and active telemetry, and proposed the in-band network telemetry framework INT-path. Similar to NetVision, INT-path couples the INT probe with the source routing label stack to accommodate the user-specified monitoring path. The controller pre-selects the telemetry source node and end node and calculates the INT path through a centralized routing algorithm. The telemetry source node rewrites the empty probe periodically sent by the external host and presses the INT path into the empty probe header in the form of an SR tag, and add INT information. At the end node, the destination address of the INT probe is set to the IP address of the controller and is sent to the controller. Since P4 does not support parsing dual-length variable stacks, INT-path statically allocates fixed-length SR label stacks, and uses right shift operation "≫" to perform stack pop.

Lin et al. [95] provided a new framework of network telemetry for data center networks, called NetView. NetView can support various telemetry applications and telemetry frequencies on demand, monitoring each device via proactively sending dedicated probes while only one vantage server is required. Technically, NetView divides the probe into a forwarding stack and a telemetry stack, which are respectively responsible for flexible forwarding and network status monitoring,

Table 4
INT term and description.

Schemes	Business packet carry telemetry instruction	Business packet carry telemetry data	MTU Limitation	Telemetry data export location	Network overhead	Telemetry metadata
INT	Yes	Yes	Limited	Decapsulate node	Large	Many
IOAM	Yes	Yes	Limited	Decapsulate node	Large	Many
PBT-M	Yes	Yes	Unlimited	Every hop	Large	Many
PBT-I	Yes	No	Unlimited	Every hop	Little	Many
iFIT	Yes	No	Unlimited	Every hop	Little	Many
AM-PM	Yes	No	Unlimited	Every hop	Little	Few
ANT	Yes	Yes	Limited	Decapsulate node	Large	Many

Table 5
Classification and summary of representative works of in-band network telemetry.

Schemes	Representative works	Technical characteristics
INT	Kim [28]	An INT demo based on programmable data plane of HTTP instantaneous delay measurement.
	POINT [66]	An Intent-driven INT framework.
	Gulenko [67]	A prototypical implementation of INT in Open vSwitch.
	INT in 6TiSCH [68]	Extends INT from wired network to wireless network.
	Sel-INT [69]	A real-time programmable and selectable INT solution based on the Protocol Oblivious Forwarding (POF).
	ML-INT [73]	A P4-based flexible multilayer INT system for IP-over-optical network.
IOAM	PBT [89]	Realizes telemetry by marking business packets or inserting telemetry instruction into business packets without carrying telemetry data. The mark scheme was called PBT-M, and the instruction insertion scheme was called PBT-I.
	iFIT [39]	Integrates the advantages of IOAM and PBT.
	Co-iOAM [90]	An in situ network telemetry mechanism for monitoring and troubleshooting resource constrained 6LoWPAN/LPWAN network.
AM-PM	AM-PM [30]	The early AM-PM solution.
	Mizrahi [91]	Introduces the AM-PM workflow and various mechanisms.
	Riesenberg [92]	An AM-PM prototype system based on Marvell Prestera chip and P4 programming language. The double-mark measurement delay error was less than 100 ns, and the statistical error of the packet loss rate was less than 10^{-6}.
	Karaagac [93]	An AM-PM-based monitoring solution that can measure end-to-end and hop-by-hop reliability and delay performance of Industrial Wireless Sensor Networks.
	VTrace [56]	An automated diagnosis system, which solved the problem of continuous packet loss in cloud networks.
ANT	EverFlow [94]	A Microsoft data center real-time telemetry system, proactively constructs telemetry packets that are identical to the packets with network failure.
	NetVision [46]	Network telemetry as a service.
	INT-path [31]	Couples the INT probe with the source routing stack to accommodate the user-specified monitoring path.
	NetView [95]	Supports various telemetry applications and telemetry frequencies on demand, monitoring each device via proactively sending dedicated probes.

achieving full coverage and visibility. Besides, a series of probe generation algorithms and update algorithms largely reduce probe number, providing high scalability.

As shown in Table 4, due to the use of business packets to carry telemetry data, the number of bytes occupied by INT, IOAM, and ANT is limited by MTU. PBT, iFIT and AM-PM only use business packets to carry telemetry instruction and export telemetry information hop by hop, leading to high network overhead. AM-PM only supports telemetry delay and packet loss rate, other telemetry solutions support richer telemetry information. As shown in Table 5, we have sorted out the aforementioned representative works for the convenience of readers.

3. Key technologies of in-band network telemetry

The in-band network telemetry system can be divided into telemetry service layer, telemetry platform layer and telemetry sensing layer from top to bottom. The telemetry service layer is deployed with various types of telemetry applications. The telemetry platform layer deploys telemetry tasks and strategies, analyzes, stores and processes the telemetry data reported by the data plane, and provides a real-time or non-real-time telemetry service query interface layer for the telemetry service. The key component technologies of telemetry platform layer include telemetry data analysis , telemetry primitive design, telemetry data fusion, telemetry task orchestration, etc. The telemetry sensing layer is responsible for the collection of telemetry data, and the implementation of telemetry data export through gRPC [96], IPFIX [7] or NETCONF [97]. Key technologies of telemetry sensing layer include telemetry data generation, event-triggered push, data export strategy and network clock synchronization, etc. Table 6 summarizes these key technologies and representative work in detail. This section details some key technologies of telemetry platform layer and telemetry awareness layer.

3.1. Programmable data plane

In-band network telemetry relies on data plane programmability. Because the traditional network data plane has the characteristics of protocol correlation, in-band measurement has not been widely deployed in practice. In 2008, the OpenFlow switch abstracted the various lookup tables in the traditional network data plane into a common flow table structure, and abstracted the data forwarding process into a general match–action process, which to a certain extent improved

Table 6
Classification and summary of key technologies of in-band network telemetry.

Key Technologies	Representative works	Technical characteristics
Compression and Storage	Pingmesh [3]	Uses Hadoop-like self-developed big data system COSMOS for additional storage of telemetry data.
	PIQ [98]	Simplifies the complexity of in-band network telemetry traceability query and analysis by dividing the analysis plane into two parts: long-term query and interactive query.
	Ballamajalu [90]	proposes context aware compression and simple distributed probabilistic update techniques to reduce the overheads of Co-iOAM.
	INT-label [99]	Periodically labels device states onto sampled packets.
	PRoML-INT [48]	Each telemetry packet only counts a small amount of electrical/optical network information on the path to reduce server telemetry pressure.
	FS-INT [100]	A sampling mechanism.
	PINT [101]	A probabilistic variation of INT, that provides similar visibility as INT while bounding the per-packet overhead according to limits set by the user.
Query and Retrieval	PathDump [102]	A query system for data center network failure debugging, including five get APIs and four action APIs.
	Marple [103]	A declarative query language for P4 switches, including four basic functions of map, filter, group by and zip.
	EQuery [104]	An event-driven declarative query language.
	NetVision [46]	Encapsulates telemetry primitives as telemetry service APIs to provide telemetry functions.
	Newton [105]	An intent-driven traffic monitor.
Push Strategy	INTCollector [106]	Designs two channels for INT information processing: fast channel and slow channel.
	Hyun [107]	Presents the design of the overall INT management architecture.
	Selective-INT [108]	Adjusts the insertion rate of telemetry metadata according to the severity of network status information changes.
	Kučera [109]	A push-based network monitoring method.
	Vestin [110]	A programmable INT event detection mechanism.
	Hyper-Tester [111]	Realizes various network measurement tasks on switches with limited programmability and resources.
	HyperSight [112]	A network traffic monitor with both high coverage and low overhead.
Orchestration	INTO [113]	Proposes the problem of in-band network telemetry orchestration (INTO) for the first time.
	INTOPP [114]	Solves the in-band network telemetry orchestration plan problem (INTOPP) using a machine learning-assisted optimization model.
	IntOpt [115]	Designs a telemetry framework for monitoring the performance of Service Function Chain.
	Cociglio [116]	Uses clustering algorithm considering the location of the measurement points to extract the smallest subnetworks where performance can be measured.
	INT-path [31]	An in-band network telemetry optimal path planning framework.
	NetVision [46]	Uses the Hierholzer algorithm to calculate the telemetry path at time complexity O(E).

the data plane's support for custom matching forwarding processing processes ability. However, the OpenFlow forwarding model does not implement protocol irrelevance, does not support programmable logic analysis, and cannot achieve the ideal universal programmable forwarding effect. Academia and industry continue to work hard in the field of general programmable data plane to promote the development of data plane [117]. One of the most typical general programmable data plane solutions is the McKeown team's programmable protocol independent switch architecture (PISA) [118]. PISA adds a programmable parser on the basis of OpenFlow, designed the Match–Action process at the entrance and exit respectively. The data packets arriving at the PISA system are first parsed by the programmable parser, then enter the queue system through the Match–Action operation on the ingress side, and finally are reorganized by the Match–Action operation on the egress side and sent to the output port. In order to further improve the programmability of the general programmable data plane, Mckeown et al. [43] proposed the programmable protocol-independent processing language P4. Currently, the new PISA-based programmable forwarding chips all support telemetry. As shown in Table 7, the industry has proposed some PDP products based on general CPUs, dedicated GPUs and network processors, etc., which can all deploy in-band network telemetry functions.

However, we have not seen a comprehensive and reliable evaluation of network telemetry for data plane forwarding performance. While improving the forwarding performance, it is also a meaningful work to accurately assess the impact of in-band network telemetry on the forwarding performance.

3.2. Compression and storage of telemetry data

The in-band network telemetry system needs to design a storage mechanism to reduce the load on network and server, and reduce the amount of calculation required to obtain data from the telemetry platform layer.

Pingmesh [3], a network latency measurement and analysis system for Microsoft data centers, constructs more than 200 billion probes and generates 24 TB data every day. The transmission and storage cost of these data is extremely high. Pingmesh uses Hadoop-like Microsoft self-developed big data system COSMOS for additional storage of telemetry data. COSMOS theoretically has unlimited storage space.

Telemetry data are mostly time series data. Michel et al. [98] compared performance differences of relational database PostgreSQL [122] and time series database TimescaleDB [123] in data insertion, reading and storage, and analyzed the feasibility of the new relation-based time series database in the in-band network telemetry system. The persistent telemetry data storage and query framework PIQ is given. PIQ simplifies the complexity of in-band network telemetry traceability query and analysis by dividing the analysis plane into two parts: long-term query and interactive query.

Table 7
Network switching chip technology.

Schemes	Representative products	Performance	Cost	Power consumption	Programming ability
CPU	Intel IXP465 RouteBricks [119]	45Gbps	Low	High	High
GPU	PacketShader [120]	150Gbps	Low	High	Low
NP	Huawei ENP	480Gbps-1Tbps	High	Mid	High
FPGA	NetFPGA NetMagic Arista 7124 FX	100Gbps	Low	Mid	High
ASIC	Broadcom Tomahawk [50], Barefoot Tofino [121]	12.8Tbps	High	Low	High

For some telemetry tasks that do not require high real-time performance and accuracy, the telemetry sensing layer can use sampling mechanism to reduce the overhead of telemetry system.

Ballamajalu et al. [90] proposed context aware compression and simple distributed probabilistic update techniques to further reduce the byte overheads of Co-iOAM data while maintaining the quality of network monitoring.

Pan et al. [99] proposed INT-label, a lightweight network-wide telemetry architecture without introducing probe packets. INT-label periodically labels device states onto sampled packets, which is cost-effective with minuscule bandwidth overhead and seamlessly adapts to topology changes.

Tang et al. [48] designed a programmable in-band network telemetry system PRoML-INT based on P4 for IP-over-optical networks. PRoML-INT inserts the telemetry header by selecting only a part of the packets belongs to same IP flow. Each telemetry packet only counts a small amount of electrical/optical network information on the path to reduce server telemetry pressure.

Suh et al. [100] proposed a sampling mechanism FS-INT. FS-INT consists of two mechanisms: based on both packet arrival rate (FS-INT-R) and event-based (FS-INT-E). the FS-INT-R is the most efficient way to sample packets. FS-INT-R inserts telemetry data after equal interval of r packets. FS-INT-E determines whether to insert the INT metadata value according to whether the current measurement value exceeds the threshold, whether the difference between the previous hop telemetry measurement value Δ exceeds the predefined deviation threshold or other threshold events. Under the experimental results of hop-by-hop delay with mean and standard deviation of 10 ms and 50 ms respectively, compared to original INT mechanism, the measurement overhead of FS-INT-R decreases linearly with the increase of r, but the accuracy of the results is poor; FS-INT-E reduces the overhead by 50% when $\Delta = 5$ ms, and the result is accurate.

Basat et al. [101] designed probabilistic in-band network telemetry (PINT), a probabilistic variation of INT, that provides similar visibility as INT while bounding the per-packet overhead according to limits set by the user. PINT allows the overhead budget to be as low as one bit, and leverages approximation techniques to meet it.

While in-band network telemetry brings abundant data to network measurement, it also brings huge transmission and storage overhead. Therefore, the compression and aggregation of telemetry data will always be the key technology of in-band network telemetry.

3.3. Query and retrieval of telemetry data

The in-band network telemetry system needs to design a high-level query language with rich functions to shield the complexity of the underlying telemetry process and achieve user-friendliness. It allows network administrators to filter and aggregate telemetry data in an abstract manner of state declaration, which is convenient for network administrators to quickly query network performance.

Tammana et al. [102] designed a server and controller-oriented query system PathDump for data center network failure debugging, including five get APIs and four action APIs. PathDump's Operations API for alarm, execute, install and uninstall, and they support two processing processes: direct query and hierarchical query.

Narayana et al. [103] designed a declarative query language Marple for P4 switches, including four basic functions of map, filter, groupby and zip. Marple requires programmable switches to have key–value storage and divide the key–value storage into high-speed On-chip cache in SRAM and slow-speed off-chip storage in DRAM. The Marple compiler supports compilation and distribution from Marple query language to switch programming language. Marple maintains a pktstream for every flow, containing switch, qid, hdrs, uid, tin, tout and qsize. Switch and qid identify the switch and queue, hdrs records the header information of packet, uid uniquely determines a packet, tin and tout indicate the timestamps when packets enter and exit queue, and qsize represents the queue depth the packet enter.

Similar to Marple, EQuery proposed by Ran et al. [104] is an event-driven declarative query language for programmable network measurements. The user does not need to know the hardware details of data plane, but rather uses EQuery, a high-level SQL-like language, to indicate measurement task. The EQuery compiler converts user queries into telemetry rules that are deployed to network devices to complete telemetry infomation acquisitions. Currently, EQuery already supports traffic statistics, long-flow detection, link throughput, packet loss rate, packet loss location, DDoS attack detection and other telemetry tasks.

Liu et al. [46] designed a set of telemetry primitives NetVision including telemetry metadata and query primitives, and encapsulated them as telemetry service APIs to provide telemetry functions such as end-to-end delay, real-time packet transmission rate and link/node black hole discovery.

On the basis of the above research, Zhou et al. [105] proposed an intent-driven traffic monitor Newton. Firstly, Newton enabled operators to dynamically create, delete and update data plane queries without interrupting normal packet forwarding. Then, Newton optimized the telemetry system to achieve accurate network traffic monitoring. Finally, Newton executed a highly elastic query of the status of entire network. Newton has flexibility, scalability and resource efficiency, proving its great potential for deployment in large-scale programmable networks.

It is difficult to design a telemetry query language that can satisfy all network scenarios. How to balance the simplicity and scalability of the telemetry language deserves continued attention.

3.4. Push strategy of telemetry data

There are two strategies for pushing telemetry data. The first is cadence-based telemetry, which periodically uploads information to establish historical benchmark. The second type is event-based telemetry, which exports telemetry data when specific events such as link interruption or counter overflow occur, reducing the network and telemetry server load.

Tu et al. [45] designed a mechanism for extracting and filtering important network information from the raw data of INT, namely INTCollector [106]. INTCollector addressed the problem of high computational and data storage overhead. INTCollector filtered important network events such as the arrival of new flows or drastic changes of hop-by-hop delay, and stores important network events in a time series database, reducing CPU usage and storage cost. INTCollector designed two channels for INT information processing: fast channel and slow channel. The fast channel is responsible for receiving INT report packets, and the slow channel is responsible for processing and storing important INT report. The experimental results show that the INTCollector dual-channel mechanism reduces the amount of INT storage by

2–3 orders of magnitude, and significantly reduces the computing load of the telemetry server.

On the basis of INTCollector, Hyun et al. [107] presented the design of the overall INT management architecture and its two main components: the INT management system and INTCollector. The INT management system controls heterogeneous INT-enabled devices through a common interface. INTCollector uses eXpress Data Path and an event detection mechanism.

Kim et al. [108] designed Selective-INT, which adjusted the insertion rate of telemetry metadata according to the severity of network status information changes. Selective-INT reduces network overhead by 37% compared to the original INT.

Kučera et al. [109] proposed a push-based network monitoring method that allowed various types of telemetry information waiting to be reported to be directly aggregated on data plane. Notifications from switch to controller are triggered by network events, such as severe traffic fluctuations, buffer accumulation, etc., to avoid transmit or process unnecessary telemetry data.

Vestin et al. [110] developed a programmable INT event detection mechanism in P4 that allows customization of which events to report to the monitoring system. At the stream processor, fast INT report collector used the kernel bypass technique AF_XDP, which parses telemetry reports and streams them to a distributed Kafka cluster, which can apply machine learning, visualization and further monitoring tasks. While the INT report collector can process around 3 Mpps telemetry reports per core, using event pre-filtering increases the capacity by 10-15x.

HyperTester [111] included template-based packet generation, false-positive-free counter-based queries, and stateless connections. These functions can realize various network measurement tasks on switches with limited programmability and limited resources.

Zhou et al. [112] presented HyperSight, a network traffic monitor with both high coverage and low overhead. The key idea of HyperSight is to monitor networks at the behavior level via tracking packet behavior changes. HyperSight proposed three designs for behavior-level monitoring. First, HyperSight presented a declarative query language to facilitate expressing various network monitoring tasks. Second, HyperSight proposed Bloom Filter Queue (BFQ) to empower in-network capability for monitoring packet behavior changes. Third, HyperSight proposed virtual BFQ to support dynamic query compilation. Evaluation results showed that HyperSight supports a wide range of network event queries and can monitor over 99% packet behavior changes while keeping remarkably low overheads.

3.5. Orchestration of telemetry tasks

The telemetry platform layer integrates and orchestrates the telemetry tasks from different telemetry applications of telemetry service layer, and delivers them to the telemetry sensing layer in order to achieve the efficient deployment of telemetry tasks.

Orchestration of in-band network telemetry task studies how to achieve greater coverage and higher efficiency of network telemetry through lower cost. Evaluation indicators include telemetry accuracy, telemetry granularity, telemetry freshness, telemetry intrusion and telemetry scope.

In-band network telemetry orchestration selects appropriate network flows to carry telemetry instruction and data based on telemetry items, consistency and information freshness, which reduces in-band network telemetry intrusion, and provides higher-efficiency network monitoring services. Telemetry system needs to consider the constraints of potential impact factors such as network traffic and node processing capabilities. Constraints include:

1. Data link layer MTU limitation.
2. The cumulative limitation of telemetry data packet length.
3. Restrictions on service capabilities of switches/routers.
4. The remaining bandwidth limit of the network link.
5. The freshness limitation of telemetry information.
6. The spatiotemporal correlation between telemetry tasks and network traffic.

Marques et al. [113] modeled the problem of in-band network telemetry orchestration (INTO) for the first time, and proposed INTO problems where the optimization objectives were to minimize the number of telemetry activity flows and the saturation probability of any telemetry link. Both derivative forms are NP-Complete problems. At the same time, Marques et al. also proposed two corresponding heuristic solution algorithms, which can compute the optimal solution for telemetry flow distribution in 1 s in real WAN verification topology.

Hohemberger et al. [114] solved the in-band network telemetry orchestration plan problem (INTOPP) using a machine learning-assisted optimization model. The INTOPP problem is described as finding a match that maximizes temporal and spatial correlation for multiple telemetry tasks and multiple network flows. The temporal correlation refers to the correlation between the freshness of the telemetry task information and the flow survival time, and the spatial correlation refers to the correlation between telemetry task nodes and path nodes. INTOPP formalized the telemetry task into a mixed integer linear programming problem (MILP) and proved to be an NP-Hard problem. Hohemberger et al. also deduced the existence of an approximate optimal solution for MILP, and designed an orchestration model based on machine learning. Compared with INTO, INTOPP's network anomaly detection rate increased by 8 times.

Bhamare et al. [115] designed a telemetry framework IntOpt for monitoring the performance of Service Function Chain (SFC), and proposed a heuristic random greedy metaheuristic (SARG) based on simulated annealing algorithm. Similar to INT-path, SARG chose to minimize telemetry overhead as the optimization goal, i.e., to minimize the number of telemetry paths needed to cover all telemetry tasks. First, SARG initializes a telemetry data flow and assigns any random chain to it. Then, a new random link is added. If the new link has similar or less telemetry requirements than the link already assigned to the given telemetry data flow, the flow accepts the new link. Finally, SARG adds repeatedly new link until the telemetry data flow exceeds the threshold. In addition, SARG also uses a simulated annealing algorithm to avoid the random greedy allocation falling into a local optimum. Numerical calculations show that , SARG can reduce monitoring overhead by 39% and total telemetry delay by 57%.

AM-PM can be applied only to unicast flows because it assumes that all the packets of the flow measured on one node are measured again by a single second node. Cociglio et al. [116] presented a novel model based on AM-PM to passively monitor backbone networks. This model used clustering algorithm considering the location of the measurement points to extract the smallest subnetworks where performance can be measured.

Active network telemetry path planning studies how to use source routing to achieve maximum network coverage under minimum cost. Pan et al. [31] proposed an in-band network telemetry optimal path planning framework INT-path. INT-path decouples path planning into Routing mechanism and path generation strategy. The routing mechanism specifies the routing path of the telemetry packet by embedding the source route into the INT telemetry packet. The INT-path path planning goal is to minimize the telemetry overhead, that is, the telemetry path is as few as possible and balanced. The path generation strategy is responsible for generating the minimum coverage of the entire network. There are two generation algorithms for non-overlapping INT paths, they are depth search priority and Euler tracking. Among them, the depth search priority algorithm has low time complexity and faster speed, but the number of paths solved is more. The Euler trace algorithm can generate INT paths with the minimum number of non-overlapping INT paths.

Based on Euler's theorem in graph theory, Liu et al. [46] used the Hierholzer algorithm to calculate the telemetry path at time complexity O(E).

Table 8
Summary of network performance telemetry applications.

Schemes	Research objectives	Telemetry type	Technical characteristics
Kim [28]	One-way delay	INT	Combines INT path positioning information and switch queuing delay information to achieve instantaneous delay measurement of HTTP applications.
Mizrahi [91]	One-way delay	AM-PM	Uses a small number of bits to mark business packets, and telemetry server calculates the network hop-by-hop delay.
Riesenberg [92]	One-way delay	AM-PM	Based on Marvell Prestera chip and P4 programming language, the delay error is less than 100 ns using double mark measurement, and the packet loss rate error is almost zero.
IntMon [124]	One-way delay	INT	The first implementation of INT encapsulated in UDP based on "ONOS controller + BMv2 switch".
Pingmesh [3]	Two-way delay	ANT	Detects data center network delay changes through active Ping operations.
EverFlow [125]	Two-way delay	ANT	A packet-level network telemetry system for data center networks, which sends probes to test and confirm potential network failures.
Fioccola [30]	Packet loss	AM-PM	Uses a small number of bits to mark business packets, and the telemetry server calculates the network packet loss rate based on the number of missing bits.
INT-path [31]	Queue depth	ANT	Designs INT active probes based on source routing encapsulation, and draws traffic graphs by specifying monitoring paths.
SIMON [44]	Queue depth	INT	An accurate and scalable measurement system for data centers, accelerating measurement through multi-layered neural networks.
CAPEST [126]	Available bandwidth	INT	The P4 program is designed to make up for the lack of network capacity and available bandwidth measurement in the existing INT solution. The measurement intrusion is low, the accuracy is high, and the freshness is high.
DNM [114]	QoS	INT	The performance of the distributed network function virtualization service quality monitoring system based on integer linear programming is 3 times higher than that of EverFlow.
IntOpt [115]	QoS	ANT	The controller creates a minimum number of telemetry flows based on simulated annealing algorithm to monitor the Qos of the service function chain.
NFT [127]	QoS	INT	The earliest network function virtualization performance telemetry architecture.
Transverse [128]	QoS	INT	The control plane and data plane collaborate to detect and verify end-to-end SLA across SDN domains.
Isolani [129]	QoS	INT	An SDN-based framework for slice orchestration using INT monitoring techniques, which is capable of fine-grained statistics gathering via INT-enabled nodes and periodic statistics analysis against current application QoS requirements.

4. In-band network telemetry applications

In-band network telemetry technology is originally applied to network performance measurement, and has now been extended to fault location, congestion control, routing decision-making, traffic engineering, and network data plane verification. In general, network telemetry applications can be divided into performance telemetry applications (see Table 8) and functional telemetry applications (see Table 9).

4.1. Network performance telemetry

In order to meet the needs of network operation and security, the control plane and management plane need to obtain real-time network performance status. Common network performance telemetry includes: delay, packet loss, available bandwidth and QoS measurement, etc.

For delay measurement, Kim et al. [28] implemented the HTTP request one-way delay measurement based on INT of the programmable data plane. EverFlow [125] and Pingmesh [3] actively construct network probes to count network end-to-end delay and jitter changes, and detect network faults with drastic delay changes for data centers. Fioccola et al. [30] and Riesenberg et al. [92] use AM-PM to calculate the network hop-by-hop delay, which often leads to low data plane overhead and high measurement accuracy .

For available bandwidth measurement, Kagami et al. [126] proposed CAPEST, an INT-based data plane offloading scheme for network capacity and available bandwidth estimation. This scheme compensates for the lack of link bandwidth and effective bandwidth metadata in existing INT schemes. Compared to the traditional scheme, CAPEST constructs telemetry packets containing Port ID, Capacity and Available Bandwidth when link capacity and available bandwidth need to be estimated. First, the P4 switch does an autocorrelation operation on the estimated histogram of the statistical information and the inverse of the estimated histogram to calculate the corrected capacity estimate. Then, CAPEST obtains the estimate of available bandwidth from the estimate of capacity and utilization. Finally, the value of link bandwidth and available bandwidth are inserted into the telemetry data in the business packet. CAPEST significantly reduces telemetry intrusions, and improves accuracy and freshness.

SIMON is an accurate and scalable measurement system for data center that reconstructs key network state variables like packet queuing times at switches, link utilizations, and queue and link compositions at the flow-level. Geng et al. [44] demonstrated that the function approximation capability of multi-layered neural networks can speed up SIMON by a factor of 5000–10000, enabling it to run in near real-time.

4.2. Microburst detection

Microbursts means that the port receives a lot of burst data in a very short time (millisecond level). Due to the technical limitation of second-by-second traffic monitoring cycles, it is difficult for SNMP or NetFlow-based tools to catch microbursts.

The TPP proposed by Jeyakumar et al. [64] detected the degree of network microburstiness by recording the length of the switch queue at the moment of packet forwarding. TPP allocated storage space in the Ethernet header for hop-by-hop switch IDs (16 bits), port numbers (16 bits), and queue lengths (16 bits). Assuming that the maximum number of hops in the data center network is 5 hops, only an additional 54 bytes of load is required to forward the packet, where the TPP telemetry header is 12 bytes, the telemetry command is 12 bytes, and the hop-by-hop statistics are 30 bytes.

Joshi et al. [144] used P4 language to detect microbursts on Barefoot Tofino switch by reporting only microburst packet telemetry information, resulting in a 10 times reduction in telemetry overhead.

Table 9
Summary of network function telemetry applications.

Schemes	Research objectives	Telemetry type	Technical characteristics
TPP [64]	Microburst detection	INT	Records switch, port number and queue length hop by hop.
PRoML-INT [48]	Network troubleshooting	INT	After obtaining the fine-grained telemetry data of the optical packet network within 1 ms, the deep neural network offline training can be used to achieve abnormal data detection and cause classification.
Hohemberger [130]	Network troubleshooting	INT	Uses machine learning to organize telemetry deployment plans, the detection rate of anomalies reaches 97%
NetSight [131]	Network troubleshooting	IOAM	The packet history filter diagnoses network faults through the postcard reported hop by hop.
KeySight [132]	Network troubleshooting	IOAM	The fault detection platform based on Bloom filter uploads postcard when packet is abnormal, which can take into account the detection coverage and scalability.
Jia [133]	Network troubleshooting	INT	A rapid gray failure detection and localization mechanism based on INT.
NetVision [46]	Routing loop detection	ANT	Proactively sends probe data packets of the appropriate number and format that match the network status and telemetry tasks, and finds routing black holes by comparing the sending rate and the receiving rate.
Sel-INT [69]	Path verification	INT	Collects device ID at a sampling rate of 10% to detect inconsistent forwarding paths caused by misconfiguration.
PathDump [102]	Path verification	IOAM	The switch inserts the path ID, and the user terminal captures and counts the data packet forwarding path.
CherryPick [134]	Path verification	IOAM	The switch inserts the forwarding path ID into packet according to the matching rule, and the controller verifies the actual forwarding path according to the path ID.
Wang [135]	Match rule verification	INT	According to the change degree of the INT telemetry information value, the report rate of the matching situation of the network flow table is adaptively adjusted, saving network bandwidth overhead by more than 39 times.
HPCC [136]	Congestion control	INT	For high-performance data center networks, the terminal calculates the appropriate sending rate based on the fine-grained switch load information, and the rate update is driven by ACK packets.
RCP*[137]	Congestion control	IOAM	The terminal counts the switch ID, queue length, link utilization, and link fair sharing rate, calculates the average queue length of each link, and uses the RCP congestion control algorithm to calculate and asynchronously update the link fair sharing rate.
CLOVE [138]	Routing decision	INT	Uses the reserved field of the encapsulation protocol header to carry INT to detect real-time path utilization, maintain the path weight of the source node, and perform weighted multi-path selection.
SPIDER [139]	Routing decision	IOAM	Designs a stateful forwarding plane, uses heartbeat packets to detect path quality, and uses protocol header fields to mark and specify forwarding paths.
TPP [64]	Traffic Engineering	INT	The terminal periodically telemetry link utilization and queue length, and realizes fair bandwidth through RCP.
HULA [140]	Traffic Engineering	ANT	Updates bandwidth utilization by periodically sending probes to obtain congestion information.
Speedlight [141]	Traffic Engineering	IOAM	Takes 10 ms—accurate state snapshots of the entire network traffic.
Hyun [142]	Network intelligence	INT	Knowledge-defined network prototype solution implemented on the “ONOS+BMv2” platform.
NetworkAI [143]	Network intelligence	INT	Dynamically generates close to optimal control strategies using deep reinforcement learning combined with INT.

4.3. Network troubleshooting

Handigol et al. [131] designed the network fault diagnosis tool NetSight by capturing the forwarding history of data packets on network. NetSight created an event record, namely Postcard, when the data packet passed through the switch. Postcard contains the copy of packet header information, switch ID, output port, switch forwarding status and other information. By collecting the Postcards of the telemetry packet, the NetSight server restored the historical state. Based on the historical status, NetSight deployed four applications: ndb, netwatch, netshark and nprof. Network administrators use regular expression-like packet history filters to perform purposeful network troubleshooting. NetSight also reduced the bandwidth overhead through both data compression and label simplification. Theoretical analysis shows that the optimized NetSight generates a bandwidth load of only 3%–7% compared to the native solution, which has a bandwidth load of 31%.

Xia et al. [132] designed an event-triggered fault detection platform KeySight based on Bloom filter. Compared to the NetSight, KeySight uploads Postcard when it detects anomalous packet behavior, which reduces the frequency of telemetry reporting and enables a balance between detection coverage and scalability.

Tang et al. [145] proposed a policy-aware INT method PAINT for the SDN fault location. In PAINT, network operators use Service Provisioning Language (SPL) to define and deploy in-band network telemetry services. PAINT automatically parsed service policies and inferred causal relationships between service-related network components and end-to-end network symptoms. Based on causality model, PAINT deploys symptom monitoring tools, and used the “symptom-fault-telemetry” model to improve the efficiency and accuracy of SDN fault location.

In modern data center networks, many network failures may happen silently with packets discarded without producing any explicit notification before causing tremendous damage to the network. To troubleshoot these "gray failures", Jia et al. [133] presented a rapid gray failure detection and localization mechanism based on INT. Once a network failure occurs, servers can proactively perform source routing-based fast traffic reroute to avoid massive packet loss and retain uninterrupted quality of experience.

Barefoot Deep Insight [42] is the world’s first commercial-level packet-by-packet status monitoring system, which can capture, analyze and locate the cause and location of packet delay in real time. Combined with machine learning technologies, Barefoot Deep Insight can realize automatic abnormality monitoring of network performance at nanosecond time, including microburst detection, abnormal packet loss detection, abnormal queue detection, etc.

In brief summary, with big data, fault root cause analysis and other technologies, in-band network telemetry is close to meeting the technical requirements of real-time troubleshooting.

4.4. Congestion control

The network source can use the rich network status information provided by in-band network telemetry for new congestion avoidance and control protocols.

HPCC [136] is a data center network load balancing scheme based on INT. By solving the INT telemetry feedback delay and measurement overshoot problems, HPCC obtains accurate link load information and quickly converges network congestion to maximize bandwidth utilization and congestion avoidance. Compared to traditional data center congestion control solutions, HPCC can quickly converge on large networks, reduce dependence on switch caches, ensure flow fairness, and reduce flow completion time by 95%.

Jeyakumar et al. [137] designed a congestion control scheme RCP* based on Rate Control Protocol (RCP) with individual flow state statistics. RCP* deploys rate limiters and rate controllers on end hosts, and assigns memory addresses to each flow to store a fair rate. The rate controller for each flow periodically queries and updates the network state in "collect-compute-update" phases. In the collection phase, the rate controller counts hop-by-hop switch ID, queue length, link utilization and link fair share rate in millisecond. Since RCP* sends a telemetry data packet with fair rate adjustment in each RTT cycle, the overhead of RCP* is similar to that of TCP.

In order to monitor communication performance of high performance interconnection networks, Taffet et al. [146] augmented a packet with three pieces of information: a hop count that represents how many links the packet has traversed so far, a hop reservoir that remembers a link along the packet's path, and a congested bit that indicates whether the link in the hop reservoir was congested when it was encountered. This scheme has essentially no bandwidth overhead, as it stores only a few bits of information in the header of a monitored IP packet, making it practical to monitor every packet.

4.5. Routing decision

In addition to network connections and network hops for routing decisions, in-band network telemetry also provides more detailed network status parameters, such as link delay, packet loss rate, network congestion and link utilization. Therefore, based on these network status parameters, network nodes can develop customized performance routes in combination with new routing methods such as Segment Route, MPLS VPN, etc.

Tactile Internet puts forward the requirement of end-to-end extremely low network latency. Tactile Internet ensures reliable data packet transmission and avoids congestion by performing redundant protection on active path and backup path. Turkovic et al. [147] realized the calculation and tracking of the processing delay and queuing delay of service flows in Tactile Internet based on P4 switches. For each newly arrived tactile flow, the controller will calculate two paths that meet the end-to-end delay requirements, and select one of them for forwarding. When the increase in queuing time is detected, the congestion control program on switch will reduce congestion by changing output port of the tactile flow, and change path in real time.

4.6. Traffic engineering

CONGA [148] is a two-layer leaf-spine topology load balancing scheme based on global network information. By maintaining a path-level measurement table (e.g., link utilization) on each switch, CONGA maps small flows to light-loaded paths to maximize network throughput or minimize max network link utilization. CONGA has two benefit. One is that congestion information is explicitly visible, and the switch marks the quantified congestion information in the header of packet. The second is that load balancing can be completed within one RTT, which can quickly detect and respond to network congestion.

Jeyakumar et al. [64] designed a similar load balancing mechanism based on CONGA. In data center networks running multi-path routing protocols, end hosts are allowed to select a routing path based on packet header. Taking VLAN as an example, the end host selects different forwarding paths by changing the VLAN ID. The end host inserts telemetry instructions consisting of path VLAN ID, link utilization TX-Utilization, and link transmit bytes TX-Bytes in the packets it sends out. The purpose of setting TX-Bytes is to evaluate the path congestion if the link utilization is not updated. The receiver sends telemetry packets back to sender after receiving them. The sender constructs "path-congestion" mapping from the telemetry information. The congestion level is represented by the maximum or sum of hop-by-hop link utilization. The sender selects an appropriate path for forwarding based on the congestion map.

Aiming at the problem of low bandwidth utilization of data center network with multi-root tree topology, Katta et al. [140] proposed a distributed traffic load balancing strategy HULA. HULA maintains the next-hop bandwidth utilization table on each switch as a basis for packet routing. HULA constructs periodic probe messages to update bandwidth utilization table entries and routing table entries. The HULA probe contains the ToR switch ID and the minimum bandwidth utilization minUtil. The switch collects bandwidth utilization and faulty link information on all links through multicast. The switch does not need to save the complete path information to a destination address, but only records the next hop information with the optimal bandwidth utilization.

On the basis of HULA, Katta et al. [138] proposed CLOVE, a terminal-based state-aware load balancing scheme. The source node deploys both Clove-ECN and Clove-INT mechanisms through a software virtual switch. The Clove-ECN captures the binary congestion state of the network through ECN notification messages. Clove-INT uses the reserved field of the protocol header to carry telemetry information to collect path utilization. According to source node's path weight table, Clove implements short-flow load balancing on each path through a weighted round-robin method.

To address the problem that the cumulative delay of hungry flows increases due to the competition of data flows with the same priority, Cugini et al. [149] designed a dynamic flow priority scheduling algorithm based on INT. The data packet selects node ID, outport ID and queuing delay as INT telemetry option, and confirms whether the cumulative queuing delay exceeds a threshold during transmission. If it exceeds, the flow priority to the packet is increased. This scheme reduces the end-to-end accumulated delay and jitter of data packets by increasing the priority of the flow with high accumulated queuing delay, and ensures that the delay is bounded.

4.7. Data plane validation

Due to the continuous changes of data plane forwarding rules, it becomes extremely difficult to verify the consistency between the control plane routing view and the actual forwarding state.

NetSight [131] detected the true path of a packet by actively sending probes. When the probe passes through the switch, the switch sends the event record as a postcard to the measurement server. Each postcard contains the packet header, switch ID and outport ID. The measurement server comprehensively analyzes all postcard and network topology information to reconstruct the actual forwarding path of the packet in network.

CherryPick [134] recovered the packet path by storing the path information in the packet header. It takes advantage of the Fat-Tree topology, and censors the non-essential links in the network by edge coloring algorithm, which effectively reduces the number of measurement rules to be deployed.

In short, since in-band network telemetry can collect richer switch device information hop by hop, it can be further optimized for data plane verification.

4.8. Network intelligence

In addition to intelligence algorithms and computing power, data is also an important part of network intelligent system. In-band network telemetry can provide large amount of real-time fine-grained network data for intelligent network management and control.

Hyun et al. [142] considered SDN, network telemetry and knowledge-defined network (KDN) as the basic building blocks of closed-loop network management. Using in-band network telemetry, the knowledge-defined network prototype solution was implemented on "ONOS+BMv2" platform, and the realization ideas of traffic engineering and abnormal detection based on KDN were briefly described.

Yao et al. [143] proposed a SDN monitoring technique, namely NetworkAI, based on INT and reinforcement learning. NetworkAI designed the network status upload channel and decision-making channel in closed-loop control of network, which directly combines network decisions with network state, and generates approximately optimal decisions in real time through deep reinforcement learning. The data plane adopts in-band network telemetry to add topology, delay, queue utilization and other network state information into the user's packet header, and then exports to NetworkAI data analysis platform via Kafka/IPFIX.

Singh et al. [150] evaluated the protocol overhead and processing latency of in-band network telemetry at 40 GE data rate. The overall INT protocol header overhead (including protocol overhead such as Ethernet and IP) for a packet size of 305 bytes (containing a 231 bytes payload) is about 24%, which decreases as the payload length increases. The average switch processing delay is 360μs. Singh et al. concluded that the low latency and high throughput of in-band network telemetry can meet the metadata acquisition needs for network machine learning.

Kong et al. [151] designed a fine-grained performance monitoring, troubleshooting and real-time visualization neural network system (NNS) for elastic optical networks. The NNS consists of a multi-layer in-band network telemetry system. Based on the telemetry data collection of electrical layer, the optical performance indicator measurement is realized by inserting the optical performance monitor in optical layer. The hybrid centralized/distributed processing is used to realize automatic network control and management.

Isolani et al. [129] presented an SDN-based framework for slice orchestration using INT monitoring techniques. This framework is capable of fine-grained statistics gathering via INT-enabled nodes and periodic statistics analysis against current application QoS requirements.

In summary, network telemetry has become an indispensable component of network intelligence. In addition to "Telemetry for Network AI", "AI for Network Telemetry" is also an important research content of network intelligence. Although there is not much related research work now. But in the near future, "AI for Network Telemetry" will empower network telemetry.

5. Future challenges and research directions

In-band network telemetry has received increasing attention from academia and industry. The academia pays more attention to whether in-band network telemetry can bring new solutions to network closed-loop control, and the industry has already developed mature network supervision products. However, the existing network telemetry researches are mostly focused on the telemetry architecture and applications, and the research results are at the initial stage of "just could measure", which lack of work for cross-heterogeneous networks, universal telemetry models, and high-performance telemetry data query and so on.

5.1. General model

In-band network telemetry has the characteristics of PDP, path-associated measurement, reconfigurable, etc. How to use these characteristics to solve the situation that traditional network measurement cannot be universally applied is of great significance to the improvement of network telemetry research.

At present, the academia and industry lack of research on a universal telemetry model for integrated converged network [152] scenarios. Designing an abstract model that separates network-wide measurement query and basic measurement behaviors, using the flexibility and scalability of the programmable data plane to design an on-route telemetry solution based on heterogeneous network, and selecting reasonable network state parameters to form a unified network state view are important applications of in-band network telemetry technology. Although some new telemetry technologies that integrate the advantages of different telemetry technologies have been proposed [127], the integration of telemetry technology is also worth studying

In addition, existing telemetry query primitives [46,104,153] and models [78] still need to be optimized. In-band network telemetry needs to design and simplify telemetry data query and operation primitives in a unified manner.

5.2. High performance telemetry

The amount of telemetry metadata inserted into business packets by INT and IOAM is limited by the original size of the data packet and the maximum transmission unit of the network. In-band network telemetry will consume part of the network bandwidth. Most encapsulation protocols such as VXLAN-GPE [81], Geneve [82], NSH [79], and TCP contain checksum, and the insertion of meta-measurements requires the checksum fields to be updated, which will increase the processing cost of the switch. At present, there is a lack of evaluation research on the impact of in-band network telemetry on network performance loss. Huang et al. [154] re-architected network telemetry with resource efficiency and full accuracy, which is a very valuable research idea. Based on the quantitative representation of performance loss, designing stateful switch processing and efficient telemetry deployment strategy will be an effective way to make up for the loss of network performance.

5.3. Flow analysis

Compared to other measurement schemes, in-band network telemetry can capture more detailed individual flow status information (e.g., delay, buffer size, sub-path characteristics of multi-path transmission, etc.). Therefore, in-band network telemetry needs to face the problem of how to make full use of the massive data in the real network environment. In addition, network administrators can quickly and intuitively determine the trend of network performance changes through tracking, statistics, analysis and evaluation, and locate network failures in time, even solve these problems before network performance deteriorates. In short, establishing accurate statistical analysis models (e.g., time series analysis, regression analysis, correlation analysis, etc.) for each flow and designing optimization solutions for network management has become an important task for future research.

5.4. Telemetry data aggregation

In-band network telemetry may generate a large amount of telemetry data [3], which increases the burden of data collection, storage and analysis on server. Taking INT as an example, the total telemetry data length of a packet that needs to carry is proportional to the number of telemetry points. Assuming that a switch will add dozens of bytes of telemetry data to each packet, these accumulated tracking data may even exceed the size of the original packet. The amount of telemetry data generated by the data plane is related to the amount of

telemetry metadata, traffic, network scale, etc. The processing capacity of telemetry server may be the performance bottleneck of telemetry system. Therefore, when telemetry endpoint reports telemetry data, it should consider pre-compression, filtering and aggregation of the telemetry data to reduce the pressure on telemetry server.

The existing metadata field can only store some indispensable limited data. With the development of network automation, virtualization and convergence technology, in-band network telemetry should collect and provide more network data on demand. Therefore, future research needs to fully consider the flexibility and scalability of data field definition, aggregation, acquisition and filtering.

Since business packets may be lost due to various reasons, INT telemetry information will inevitably be lost. But the telemetry mechanism makes INT unable to detect the network packet loss. There is no feasible packet loss measurement solution for INT.[1] There are some high-performance packet loss measurement solutions, but none of them have been integrated into INT. It is worth noting that there is already scheme [133] using INT to measure packet loss rate, but it belongs to active measurement. In other words, the reason it can measure the packet loss rate is not because of INT, but because it constructs active network probes. Besides, it measures the packet loss rate of the probe instead of per-flow. Therefore, INT itself should support packet loss detection and localization.

5.5. Orchestration of telemetry tasks

Compared to traditional network measurement, in-band network telemetry task orchestration is more complicated [113]. The monitoring application only cares about the data itself, instead of how to obtain the data. Therefore, a telemetry task scheduling layer must be used to achieve efficient task distribution and data acquisition. In addition to upper-level monitoring applications and telemetry projects, the factors participating in the orchestration of in-band network telemetry tasks include real-time and changing network flows [114]. How to achieve high-quality network measurement at low cost according to the existing network status will be a key issue of in-band network telemetry that continue to be studied.

5.6. Telemetry security

The data plane programmability of in-band network telemetry may lead to potential software vulnerabilities, backdoors, viruses, and other security issues. Malicious INT and IOAM source nodes continuously construct telemetry packets to carry out malicious and illegal attacks, occupying network bandwidth, consuming node processing capabilities. These attacks may cause the exhaustion of available network resources. In-band network telemetry technology requires packet-level operations such as decapsulation, insertion, and encapsulation. These operations may threaten the confidentiality and security of user information. Therefore, in-band network telemetry systems need to have node authentication functions. Pan et al. [155] used vector homomorphic encryption to design a lightweight telemetry data encryption scheme. By hash-based signature encoding of the encrypted INT data, The telemetry server can verify the integrity of INT data, thus eliminating the threat of telemetry data security due to data tampering.

[1] The latest version of INT specification [61] has divided INT into three types: INT-XD, INT-MX and INT-MD. The first two draw on the ideas of iFIT [39] and IOAM [37], and export metadata hop by hop, so they can be directly used for loss measurement. This survey only discuss the third type of INT here.

6. Summary

In this survey, we researched in-band network telemetry from various aspects, i.e., development history, research situation, application and key technologies. Technical challenges and future research directions were also proposed.

Existing researches have documented that in-band network telemetry technology has important research value and broad application prospects. In-band network telemetry technology is in the ascendant, and there is still a lot of room for research and innovation, and it is worth continuing to explore in academia and industry.

CRediT authorship contribution statement

Lizhuang Tan: Conceptualization of this study, Writing - original draft. **Wei Su:** Project administration, Funding acquisition. **Wei Zhang:** Writing - review & editing. **Jianhui Lv:** Writing - original draft. **Zhenyi Zhang:** Data curation. **Jingying Miao:** Writing - original draft, Data curation. **Xiaoxi Liu:** Writing - original draft. **Na Li:** Writing - review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the research projects funded by the National Key R&D Program of China under Grant No. 2018YFB1800305, the Fundamental Research Funds for the Central Universities of China under Grant No. 2020YJS013, the National Natural Science Foundation of China under Grant No. 61802233, the Natural Science Foundation of Shandong Provincial under Grant No. ZR2019LZH013 and the Natural Science Foundation of Shandong Provincial under Grant No. ZR2020LZH010.

References

[1] H. Zhang, Z. Cai, Y. Li, An overview of software-defined network measurement technologies, SCI. SIN. Inf. 48 (3) (2018) 293–314, http://dx.doi.org/10.1360/N112017-00203.

[2] A. Morton, RFC 7799: Active and passive metrics and methods (with hybrid types in-between), 2016, http://dx.doi.org/10.17487/RFC7799.

[3] C. Guo, L. Yuan, D. Xiang, Y. Dang, R. Huang, D. Maltz, Z. Liu, V. Wang, B. Pang, H. Chen, et al., Pingmesh: A large-scale system for data center network latency measurement and analysis, in: Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, London, UK, 2015, pp. 139–152, http://dx.doi.org/10.1145/2785956.2787496.

[4] K. Agarwal, E. Rozner, C. Dixon, J. Carter, SDN traceroute: Tracing SDN forwarding without changing network behavior, in: Proceedings of the 3rd Workshop on Hot Topics in Software Defined Networking, Chicago, Illinois, USA, 2014, pp. 145–150, http://dx.doi.org/10.1145/2620728.2620756.

[5] R. Sommer, A. Feldmann, Netflow: Information loss or win? in: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment, Marseille, France, 2002, pp. 173–174, http://dx.doi.org/10.1145/637201.637226.

[6] P. Phaal, S. Panchen, N. McKee, RFC 3176: InMon corporation's sFlow: A method for monitoring traffic in switched and routed networks, 2001, http://dx.doi.org/10.17487/RFC3176.

[7] J. Quittek, T. Zseby, B. Claise, S. Zander, RFC 3917: Requirements for IP flow information export (IPFIX), 2004, http://dx.doi.org/10.17487/RFC3917.

[8] R.B. Basat, G. Einziger, J. Gong, J. Moraney, D. Raz, Q-MAX: A unified scheme for improving network measurement throughput, in: Proceedings of the Internet Measurement Conference, Amsterdam, Netherlands, 2019, pp. 322–336, http://dx.doi.org/10.1145/3355369.3355569.

[9] M. Roughan, Fundamental bounds on the accuracy of network performance measurements, ACM SIGMETRICS Perform. Eval. Rev. 33 (1) (2005) 253–264, http://dx.doi.org/10.1145/1071690.1064241.

[10] B.A.A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, T. Turletti, A survey of software-defined networking: Past, present, and future of programmable networks, IEEE Commun. Surv. Tutor. 16 (3) (2014) 1617–1634, http://dx.doi.org/10.1109/SURV.2014.012214.00180.

[11] R. Bifulco, G. Rétvári, A survey on the programmable data plane: Abstractions, architectures, and open problems, in: Proceedings of the 2018 IEEE 19th International Conference on High Performance Switching and Routing, Bucharest, Romania, Romania, 2018, pp. 1–7, http://dx.doi.org/10.1109/HPSR.2018.8850761.

[12] Z. Su, T. Wang, M. Hamd, COSTA: cross-layer optimization for sketch-based software defined measurement task assignment, in: Proceedings of the 2015 IEEE 23rd International Symposium on Quality of Service, Portland, OR, USA, IEEE, 2015, pp. 183–188, http://dx.doi.org/10.1109/IWQoS.2015.7404731.

[13] P. Megyesi, A. Botta, G. Aceto, A. Pescapé, S. Molnár, Challenges and solution for measuring available bandwidth in software defined networks, Comput. Commun. 99 (2017) 48–61, http://dx.doi.org/10.1016/j.comcom.2016.12.004.

[14] X. Zhang, Y. Wang, J. Zhang, L. Wang, Y. Zhao, A two-way link loss measurement approach for software-defined networks, in: Proceedings of the IEEE/ACM 25th International Symposium on Quality of Service, IWQoS, Vilanova i la Geltru, Spain, IEEE/ACM, 2017, pp. 1–10, http://dx.doi.org/10.1109/IWQoS.2017.7969164.

[15] W. Queiroz, M.A. Capretz, M. Dantas, An approach for SDN traffic monitoring based on big data techniques, J. Netw. Comput. Appl. 131 (2019) 28–39, http://dx.doi.org/10.1016/j.jnca.2019.01.016.

[16] C. Yu, C. Lumezanu, A. Sharma, Q. Xu, G. Jiang, H.V. Madhyastha, Software-defined latency monitoring in data center networks, in: Proceedings of the International Conference on Passive and Active Network Measurement, Springer, 2015, pp. 360–372, http://dx.doi.org/10.1007/978-3-319-15509-8_27.

[17] S. Wang, J. Zhang, T. Huang, J. Liu, Y.-j. Liu, F.R. Yu, FlowTrace: measuring round-trip time and tracing path in software-defined networking with low communication overhead, Front. Inf. Technol. Electron. Eng. 18 (2) (2017) 206–219, http://dx.doi.org/10.1631/FITEE.1601280.

[18] Q. Li, X. Zou, Q. Huang, J. Zheng, P.P. Lee, Dynamic packet forwarding verification in SDN, IEEE Trans. Dependable Secure Comput. 16 (6) (2018) 915–929, http://dx.doi.org/10.1109/TDSC.2018.2810880.

[19] A. AlGhadhban, B. Shihada, Flight: A fast and lightweight elephant-flow detection mechanism, in: Proceedings of the 2018 IEEE 38th International Conference on Distributed Computing Systems, 2018, pp. 1537–1538, http://dx.doi.org/10.1109/ICDCS.2018.00161.

[20] Y. Zhao, P. Zhang, Y. Jin, Netography: Troubleshoot your network with packet behavior in SDN, in: Proceedings of the 2016 IEEE/IFIP Network Operations and Management Symposium, Istanbul, Turkey, IEEE, 2016, pp. 878–882, http://dx.doi.org/10.1109/NOMS.2016.7502919.

[21] M. Yu, Network telemetry: towards a top-down approach, ACM SIGCOMM Comput. Commun. Rev. 49 (1) (2019) 11–17, http://dx.doi.org/10.1145/3314212.3314215.

[22] A. Yaar, A. Perrig, D. Song, Pi: A path identification mechanism to defend against DDoS attacks, in: Proceedings of the 2003 Symposium on Security and Privacy, Berkeley, CA, USA, IEEE, 2003, pp. 93–107, http://dx.doi.org/10.1109/SECPRI.2003.1199330.

[23] D. Katabi, M. Handley, C. Rohrs, Congestion control for high bandwidth-delay product networks, in: Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Pittsburgh, Pennsylvania, USA, ACM, 2002, pp. 89–102, http://dx.doi.org/10.1145/633025.633035.

[24] M. Luckie, A. McGregor, IPMP: IP measurement protocol, in: Passive and Active Measurement Workshop, 2002.

[25] R. Mahajan, N. Spring, D. Wetherall, T. Anderson, User-level internet path diagnosis, in: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, Bolton Landing, NY, USA, ACM, 2003, pp. 106–119, http://dx.doi.org/10.1145/945445.945456.

[26] D.P. Pezaros, D. Hutchison, J.S. Sventek, F.J. García, R.D. Gardner, In-line service measurements: an ipv6-based framework for traffic evaluation and network operations, in: Proceedings of the 2004 IEEE/IFIP Network Operations and Management Symposium, Vol. 1, Seoul, South Korea, IEEE, 2004, pp. 497–510, http://dx.doi.org/10.1109/NOMS.2004.1317736.

[27] D.P. Pezaros, K. Georgopoulos, D. Hutchison, High-speed, in-band performance measurement instrumentation for next generation IP networks, Comput. Netw. 54 (18) (2010) 3246–3263, http://dx.doi.org/10.1016/j.comnet.2010.06.014.

[28] C. Kim, A. Sivaraman, N. Katta, A. Bas, A. Dixit, L.J. Wobker, In-band network telemetry via programmable dataplanes, in: Proceedings of the ACM SIGCOMM 2015 Conference Posters and Demos, London, UK, 2015.

[29] F. Brockners, S. Bhandari, S. Dara, et al., Requirements for in-situ OAM, in: Working Draft, Internet-Draft Draft-Brockners-Inband-Oam-Requirements-03, 2017.

[30] G. Fioccola, A. Capello, M. Cociglio, L. Castaldelli, M.G. Chen, L. Zheng, G. Mirsky, T. Mizrahi, RFC 8321: Alternate-marking method for passive and hybrid performance monitoring, 2018, http://dx.doi.org/10.17487/RFC8321.

[31] T. Pan, E. Song, Z. Bian, X. Lin, X. Peng, J. Zhang, T. Huang, B. Liu, Y. Liu, INT-path: Towards optimal path planning for in-band network-wide telemetry, in: Proceedings of the IEEE Conference on Computer Communications, Paris, France, IEEE, 2019, pp. 487–495, http://dx.doi.org/10.1109/INFOCOM.2019.8737529.

[32] IETF IP Performance Measurement (ippm), https://datatracker.ietf.org/group/ippm.

[33] IETF IP flow information export (ipfix), https://datatracker.ietf.org/wg/ipfix.

[34] IETF Operations and Management Area (ops), https://datatracker.ietf.org/group/ops/.

[35] IETF Remote Network Monitoring (rmonmib), https://datatracker.ietf.org/wg/rmonmibs.

[36] IETF Packet Sampling (psamp), https://datatracker.ietf.org/wg/psamp.

[37] IETF IOAM, https://datatracker.ietf.org/wg/ioam.

[38] H. Song, T. Zhou, Z. Li, J. Shin, K. Lee, Postcard-based on-path flow data telemetry, in: Internet-Draft Draft-Song-Ippm-Postcard-Based-Telemetry-06, 2019.

[39] B. Lu, L. Xu, Y. Song, L. Dai, M. Liu, T. Zhou, Z. Li, H. Song, IFIT: Intelligent flow information telemetry, in: Proceedings of the ACM SIGCOMM 2019 Conference Posters and Demos, Beijing, China, ACM, 2019, pp. 15–17, http://dx.doi.org/10.1145/3342280.3342292.

[40] P4, https://p4.org/.

[41] https://www.opennetworking.org/wp-content/uploads/2018/12/Data-Plane-Telemetry-ONF-Connect-Public.pdf.

[42] Barefoot Deep Insight, https://barefootnetworks.com/products/brief-deep-insight.

[43] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, D. Walker, P4: Programming protocol-independent packet processors, ACM SIGCOMM Comput. Commun. Rev. 44 (3) (2014) 87–95, http://dx.doi.org/10.1145/2656877.2656890.

[44] Y. Geng, S. Liu, Z. Yin, A. Naik, B. Prabhakar, M. Rosenblum, A. Vahdat, SIMON: A simple and scalable method for sensing, inference and measurement in data center networks, in: Proceedings of the 16th USENIX Symposium on Networked Systems Design and Implementation, Boston, MA, USENIX Association, 2019, pp. 549–564, URL https://www.usenix.org/conference/nsdi19/presentation/geng.

[45] N. Van Tu, J. Hyun, G.Y. Kim, J.-H. Yoo, J.W.-K. Hong, Intcollector: A high-performance collector for in-band network telemetry, in: Proceedings of the 2018 14th International Conference on Network and Service Management, Rome, Italy, IEEE, 2018, pp. 10–18.

[46] Z. Liu, J. Bi, Y. Zhou, Y. Wang, Y. Lin, NetVision: Towards network telemetry as a service, in: Proceedings of the 2018 IEEE 26th International Conference on Network Protocols, Cambridge, UK, IEEE, 2018, pp. 247–248, http://dx.doi.org/10.1109/ICNP.2018.00036.

[47] Q. Huang, X. Jin, P.P.C. Lee, R. Li, L. Tang, Y. Chen, G. Zhang, Sketchvisor: Robust network measurement for software packet processing, in: Proceedings of the Conference of the ACM Special Interest Group on Data Communication, Los Angeles, CA, USA, ACM, 2017, pp. 113–126, http://dx.doi.org/10.1145/3098822.3098831.

[48] S. Tang, J. Kong, B. Niu, Z. Zhu, Programmable multilayer INT: An enabler for AI-assisted network automation, IEEE Commun. Mag. 58 (1) (2020) 26–32, http://dx.doi.org/10.1109/MCOM.001.1900365.

[49] VPP Inband OAM (IOAM), https://docs.fd.io/vpp/17.04/ioam_plugin_doc.html.

[50] Broadcom Tomahawk, https://www.broadcom.cn/products/ethernet-connectivity/switching/strataxgs/bcm56960-series.

[51] T. Mizrahi, V. Vovnoboy, M. Nisim, G. Navon, A. Soffer, Network telemetry solutions for data center and enterprise networks, Marvell, White Paper, 2018, URL https://www.marvell.com/content/dam/marvell/en/public-collateral/switching/marvell-telemetry-white-paper-2018-03.pdf.

[52] Arista, https://www.arista.com/en/solutions/telemetry-analytics.

[53] What-just-happened, https://www.mellanox.com/products/what-just-happened/.

[54] Junos telemetry interface, https://www.juniper.net/documentation/en_US/junos/topics/concept/junos-telemetry-interface-oveview.html.

[55] Traceflow, https://www.oreilly.com/library/view/learning-vmware-nsx/9781788398985/cd260a21-62d6-4da4-bb39-8978d8dc1739.xhtml.

[56] C. Fang, H. Liu, M. Miao, J. Ye, L. Wang, W. Zhang, D. Kang, B. Lyv, P. Cheng, J. Chen, Vtrace: Automatic diagnostic system for persistent packet loss in cloud-scale overlay network, in: Proceedings of the 2020 ACM Conference on SIGCOMM, New York City, USA, ACM, 2020, pp. 1–10.

[57] FabricInsight, https://e.huawei.com/cn/material/networking/networkanalyzers/6865c8bf1f1d4893943dfe71e8eda765.

[58] H. Zhang, Z. Cai, Q. Liu, Q. Xiao, Y. Li, C.F. Cheang, A survey on security-aware measurement in SDN, Secur. Commun. Netw. 2018 (2018) 1–14, http://dx.doi.org/10.1155/2018/2459154.

[59] P.-W. Tsai, C.-W. Tsai, C.-W. Hsu, C.-S. Yang, Network monitoring in software-defined networking: A review, IEEE Syst. J. 12 (4) (2018) 3958–3969, http://dx.doi.org/10.1109/JSYST.2018.2798060.

[60] D. Zhou, Z. Yan, Y. Fu, Z. Yao, A survey on network data collection, J. Netw. Comput. Appl. 116 (2018) 9–23.

[61] In-band network telemetry (INT) dataplane specification v2.1, 2020, URL https://github.com/p4lang/p4-applications/blob/master/docs.

[62] F. Brockners, S. Bhandari, C. Pignataro, et al., Data fields for in-situ OAM, in: Working Draft, Internet-Draft Draft-Ietf-Ippm-Ioam-Data-09, https://tools.ietf.org/html/draft-ietf-ippm-ioam-data-09.

[63] D.L. Tennenhouse, J.M. Smith, W.D. Sincoskie, D.J. Wetherall, G.J. Minden, A survey of active network research, IEEE Commun. Mag. 35 (1) (1997) 80–86, http://dx.doi.org/10.1109/35.568214.

[64] V. Jeyakumar, M. Alizadeh, Y. Geng, C. Kim, D. Mazières, Millions of little minions: Using packets for low latency network programming and visibility, in: Proceedings of the 2014 ACM Conference on SIGCOMM, Chicago, Illinois, USA, ACM, 2014, pp. 3–14, http://dx.doi.org/10.1145/2619239.2626292.

[65] INT, https://p4.org/p4/inband-network-telemetry.

[66] M. Anand, R. Subrahmaniam, R. Valiveti, POINT: An intent-driven framework for integrated packet-optical in-band network telemetry, in: Proceedings of the 2018 IEEE International Conference on Communications, Kansas City, MO, USA, IEEE, 2018, pp. 1–6, http://dx.doi.org/10.1109/ICC.2018.8422785.

[67] A. Gulenko, M. Wallschläger, O. Kao, A practical implementation of in-band network telemetry in open vswitch, in: 2018 IEEE 7th International Conference on Cloud Networking, Tokyo, Japan, IEEE, 2018, pp. 1–4, http://dx.doi.org/10.1109/CloudNet.2018.8549431.

[68] A. Karaagac, E. De Poorter, J. Hoebeke, In-band network telemetry in industrial wireless sensor networks, IEEE Trans. Netw. Serv. Manag. 17 (1) (2020) 517–531, http://dx.doi.org/10.1109/TNSM.2019.2949509.

[69] S. Tang, D. Li, B. Niu, J. Peng, Z. Zhu, Sel-INT: A runtime-programmable selective in-band network telemetry system, IEEE Trans. Netw. Serv. Manag. 17 (2) (2020) 708–721, http://dx.doi.org/10.1109/TNSM.2019.2953327.

[70] B. Pfaff, J. Pettit, T. Koponen, E. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, P. Shelar, et al., The design and implementation of open vswitch, in: Proceedings of the 12th USENIX Symposium on Networked Systems Design and Implementation, Oakland, CA, USENIX Association, 2015, pp. 117–130, URL https://www.usenix.org/conference/nsdi15/technical-sessions/presentation/pfaff.

[71] M. Shahbaz, S. Choi, B. Pfaff, C. Kim, N. Feamster, N. McKeown, J. Rexford, PISCES: A programmable, protocol-independent software switch, in: Proceedings of the 2016 ACM SIGCOMM Conference, Florianopolis, Brazil, ACM, 2016, pp. 525–538, http://dx.doi.org/10.1145/2934872.2934886.

[72] Behavioral-model, https://github.com/p4lang/behavioral-model.

[73] B. Niu, J. Kong, S. Tang, Y. Li, Z. Zhu, Visualize your IP-over-optical network in realtime: a P4-based flexible multilayer in-band network telemetry (ML-INT) system, IEEE Access 7 (2019) 82413–82423, http://dx.doi.org/10.1109/ACCESS.2019.2924332.

[74] T. Mizrahi, N. Sprecher, E. Bellagamba, Y. Weingarten, RFC 7276: An overview of operations, administration, and maintenance (OAM) tools, 2014, http://dx.doi.org/10.17487/RFC7276.

[75] S. Bhandari, F. Brockners, C. Pignataro, et al., In-situ OAM IPv6 Options, in: Working Draft, Internet-Draft Draft-Ietf-Ippm-Ioam-Ipv6-Options-02, https://tools.ietf.org/html/draft-ietf-ippm-ioam-ipv6-options-02.

[76] T. Mizrahi, F. Brockners, S. Bhandari, et al., In-situ OAM Flags, in: Working Draft, Internet-Draft Draft-Ietf-Ippm-Ioam-Flags-01, https://tools.ietf.org/html/draft-ietf-ippm-ioam-flags-01.

[77] T. Mizrahi, F. Brockners, S. Bhandari, et al., In Situ OAM profiles, in: Working Draft, Internet-Draft Draft-Mizrahi-Ippm-Ioam-Profile-02, https://tools.ietf.org/html/draft-mizrahi-ippm-ioam-profile-02.

[78] T. Zhou, J. Guichard, F. Brockners, et al., A YANG data model for in-situ OAM, in: Working Draft, Internet-Draft Draft-Zhou-Ippm-Ioam-Yang-07, https://tools.ietf.org/html/draft-zhou-ippm-ioam-yang-07.

[79] F. Brockners, S. Bhandari, Network Service Header (NSH) encapsulation for in-situ OAM (IOAM) Data, in: Working Draft, Internet-Draft Draft-Ietf-Sfc-Ioam-Nsh-03, https://tools.ietf.org/html/draft-ietf-sfc-ioam-nsh-03.

[80] R. Gandhi, Z. Ali, C. Filsfils, et al., MPLS data plane encapsulation for in-situ OAM data, in: Working Draft, Internet-Draft Draft-Gandhi-Mpls-Ioam-Sr-02, https://tools.ietf.org/html/draft-gandhi-mpls-ioam-sr-02.

[81] F. Brockners, S. Bhandari, V. Govindan, et al., VXLAN-GPE encapsulation for in-situ OAM data, in: Working Draft, Internet-Draft Draft-Brockners-Ippm-Ioam-Vxlan-Gpe-03, https://tools.ietf.org/html/draft-brockners-ippm-ioam-vxlan-gpe-03.

[82] F. Brockners, S. Bhandari, V. Govindan, et al., Geneve encapsulation for In-situ OAM data, in: Working Draft, Internet-Draft Draft-Brockners-Ippm-Ioam-Geneve-00, https://tools.ietf.org/html/draft-brockners-ippm-ioam-geneve-00.

[83] B. Weis, F. Brockners, C. Hill, et al., Ethertype protocol identification of in-situ OAM Data, in: Working Draft, Internet-Draft Draft-Weis-Ippm-Ioam-Eth-02, https://tools.ietf.org/html/draft-weis-ippm-ioam-eth-02.

[84] M. Spiegel, F. Brockners, S. Bhandari, et al., In-situ OAM raw data export with IPFIX, in: Working Draft, Internet-Draft Draft-Spiegel-Ippm-Ioam-Rawexport-03, https://tools.ietf.org/html/draft-spiegel-ippm-ioam-rawexport-03.

[85] H. Song, B. Gafni, T. Zhou, et al., In-situ OAM Direct Exporting, in: Working Draft, Internet-Draft Draft-Ietf-Ippm-Ioam-Direct-Export-00, https://tools.ietf.org/html/draft-ietf-ippm-ioam-direct-export-00.

[86] F. Brockners, S. Bhandari, T. Mizrahi, et al., Proof of transit, in: Working Draft, Internet-Draft Draft-Ietf-Sfc-Proof-of-Transit-04, https://tools.ietf.org/html/draft-ietf-sfc-proof-of-transit-04.

[87] IOAM for IPv6, https://github.com/IurmanJ/kernel_ipv6_ioam.

[88] OpenDaylight, https://git.opendaylight.org.

[89] H. Song, T. Zhou, Z. Li, et al., Postcard-based on-Path flow data telemetry, in: Working Draft, Internet-Draft Draft-Song-Ippm-Postcard-Based-Telemetry-06, https://tools.ietf.org/html/draft-song-ippm-postcard-based-telemetry-06.

[90] R. Ballamajalu, S. Anand, M. Hegde, Co-iOAM: In-situ telemetry metadata transport for resource constrained networks within IETF standards framework, in: Proceedings of the 2018 10th International Conference on Communication Systems & Networks, Bengaluru, India, IEEE, 2018, pp. 573–576, http://dx.doi.org/10.1109/COMSNETS.2018.8328276.

[91] T. Mizrahi, G. Navon, G. Fioccola, M. Cociglio, M. Chen, G. Mirsky, AM-PM: Efficient network telemetry using alternate marking, IEEE Netw. 33 (4) (2019) 155–161, http://dx.doi.org/10.1109/MNET.2019.1800152.

[92] A. Riesenberg, Y. Kirzon, M. Bunin, E. Galili, G. Navon, T. Mizrahi, Time-multiplexed parsing in marking-based network telemetry, in: Proceedings of the 12th ACM International Conference on Systems and Storage, Haifa, Israel, ACM, 2019, pp. 80–85, http://dx.doi.org/10.1145/3319647.3325837.

[93] A. Karaagac, E. De Poorter, J. Hoebeke, Alternate marking-based network telemetry for industrial WSNs, in: Proceedings of the 2020 16th IEEE International Conference on Factory Communication Systems, 2020, pp. 1–8, http://dx.doi.org/10.1109/WFCS47810.2020.9114490.

[94] Y. Zhu, N. Kang, J. Cao, A. Greenberg, G. Lu, R. Mahajan, D. Maltz, L. Yuan, M. Zhang, B.Y. Zhao, H. Zheng, Packet-level telemetry in large datacenter networks, ACM SIGCOMM Comput. Commun. Rev. 45 (4) (2015) 479–491, http://dx.doi.org/10.1145/2829988.2787483.

[95] Y. Lin, Y. Zhou, Z. Liu, K. Liu, Y. Wang, M. Xu, J. Bi, Y. Liu, J. Wu, Netview: Towards on-demand network-wide telemetry in the data center, Comput. Netw. (2020) 107386, http://dx.doi.org/10.1016/j.comnet.2020.107386.

[96] Google remote procedure calls, https://www.grpc.io/.

[97] R. Enns, M. Bjorklund, J. Schoenwaelder, A. Bierman, Network configuration protocol (NETCONF), 2011, RFC 6241.

[98] O. Michel, J. Sonchack, E. Keller, J.M. Smith, PIQ: Persistent interactive queries for network security analytics, in: Proceedings of the ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization, Richardson, Texas, USA, ACM, 2019, pp. 17–22, http://dx.doi.org/10.1145/3309194.3309197.

[99] T. Pan, E. Song, C. Jia, W. Cao, T. Huang, B. Liu, Lightweight Network-Wide Telemetry Without Explicitly Using Probe Packets, in: IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS, 2020, pp. 1–2.

[100] D. Suh, S. Jang, S. Han, S. Pack, X. Wang, Flexible sampling-based in-band network telemetry in programmable data plane, ICT Express 6 (1) (2020) 62–65, http://dx.doi.org/10.1016/j.icte.2019.08.005.

[101] R. Ben Basat, S. Ramanathan, Y. Li, G. Antichi, M. Yu, M. Mitzenmacher, PINT: Probabilistic in-band network telemetry, in: Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication, SIGCOMM '20, ACM, 2020, pp. 662–680, http://dx.doi.org/10.1145/3387514.3405894.

[102] P. Tammana, R. Agarwal, M. Lee, Simplifying datacenter network debugging with pathdump, in: Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation, USA, USENIX Association, 2016, pp. 233–248.

[103] S. Narayana, A. Sivaraman, V. Nathan, P. Goyal, V. Arun, M. Alizadeh, V. Jeyakumar, C. Kim, Language-directed hardware design for network performance monitoring, in: Proceedings of the Conference of the ACM Special Interest Group on Data Communication, Los Angeles, CA, USA, ACM, 2017, pp. 85–98, http://dx.doi.org/10.1145/3098822.3098829.

[104] Y. Ran, X. Wu, P. Li, C. Xu, Y. Luo, L.-M. Wang, Equery: Enable event-driven declarative queries in programmable network measurement, in: Proceedings of the 2018 IEEE/IFIP Network Operations and Management Symposium, Taipei, Taiwan, IEEE, 2018, pp. 1–7, http://dx.doi.org/10.1109/NOMS.2018.8406142.

[105] Y. Zhou, D. Zhang, K. Gao, C. Sun, J. Cao, Y. Wang, M. Xu, J. Wu, Newton: Intent-driven network traffic monitoring, in: Proceedings of the 16th International Conference on Emerging Networking EXperiments and Technologies, CoNEXT '20, ACM, 2020, pp. 295–308, http://dx.doi.org/10.1145/3386367.3431298.

[106] INTCollector, https://gitlab.com/tunv_ebpf/BPFCollector.

[107] J. Hyun, N. Van Tu, J.-H. Yoo, J.W.-K. Hong, Real-time and fine-grained network monitoring using in-band network telemetry, Int. J. Netw. Manage. 29 (6) (2019) e2080, http://dx.doi.org/10.1002/nem.2080.

[108] Y. Kim, D. Suh, S. Pack, Selective in-band network telemetry for overhead reduction, in: Proceedings of the 2018 IEEE 7th International Conference on Cloud Networking, Tokyo, Japan, IEEE, 2018, pp. 1–3, http://dx.doi.org/10.1109/CloudNet.2018.8549351.

[109] J. Kučera, D.A. Popescu, H. Wang, A. Moore, J. Kořenek, G. Antichi, Enabling event-triggered data plane monitoring, in: Proceedings of the Symposium on SDN Research, San Jose, CA, USA, ACM, 2020, pp. 14–26, http://dx.doi.org/10.1145/3373360.3380830.

[110] J. Vestin, A. Kassler, D. Bhamare, K.-J. Grinnemo, J.-O. Andersson, G. Pongracz, Programmable event detection for in-band network telemetry, in: Proceedings of the 2019 IEEE 8th International Conference on Cloud Networking, Coimbra, Portugal, IEEE, 2019, pp. 1–6, http://dx.doi.org/10.1109/CloudNet47604.2019.9064137.

[111] Y. Zhou, Z. Xi, D. Zhang, Y. Wang, J. Wang, M. Xu, J. Wu, Hypertester: High-performance network testing driven by programmable switches, in: Proceedings of the 15th International Conference on Emerging Networking Experiments and Technologies, Orlando, Florida, ACM, 2019, pp. 30–43, http://dx.doi.org/10.1145/3359989.3365406.

[112] Y. Zhou, J. Bi, T. Yang, K. Gao, J. Cao, D. Zhang, Y. Wang, C. Zhang, Hypersight: Towards scalable, high-coverage, and dynamic network monitoring queries, IEEE J. Sel. Areas Commun. 38 (6) (2020) 1147–1160, http://dx.doi.org/10.1109/JSAC.2020.2986690.

[113] J.A. Marques, M.C. Luizelli, R.I.T. da Costa Filho, L.P. Gaspary, An optimization-based approach for efficient network monitoring using in-band network telemetry, J. Internet Serv. Appl. 10 (1) (2019) 12.

[114] R. Hohemberger, A.F. Lorenzon, F. Rossi, M.C. Luizelli, Optimizing distributed network monitoring for NFV service chains, IEEE Commun. Lett. 23 (8) (2019) 1332–1336, http://dx.doi.org/10.1109/LCOMM.2019.2922184.

[115] D. Bhamare, A. Kassler, J. Vestin, M.A. Khoshkholghi, J. Taheri, Intopt: In-band network telemetry optimization for NFV service chain monitoring, in: Proceedings of the 2019 IEEE International Conference on Communications, Shanghai, China, IEEE, 2019, pp. 1–7, http://dx.doi.org/10.1109/ICC.2019.8761722.

[116] M. Cociglio, G. Fioccola, G. Marchetto, A. Sapio, R. Sisto, Multipoint passive monitoring in packet networks, IEEE/ACM Trans. Netw. 27 (6) (2019) 2377–2390, http://dx.doi.org/10.1109/TNET.2019.2950157.

[117] W.L. da Costa Cordeiro, J.A. Marques, L.P. Gaspary, Data plane programmability beyond openflow: Opportunities and challenges for network and service operations and management, J. Netw. Syst. Manage. 25 (4) (2017) 784–818, http://dx.doi.org/10.1007/s10922-017-9423-2.

[118] S. Chole, A. Fingerhut, S. Ma, A. Sivaraman, S. Vargaftik, A. Berger, G. Mendelson, M. Alizadeh, S.-T. Chuang, I. Keslassy, et al., DRMT: Disaggregated programmable switching, in: Proceedings of the Conference of the ACM Special Interest Group on Data Communication, Los Angeles, CA, USA, ACM, 2017, pp. 1–14, http://dx.doi.org/10.1145/3098822.3098823.

[119] M. Dobrescu, N. Egi, K. Argyraki, B.-G. Chun, K. Fall, G. Iannaccone, A. Knies, M. Manesh, S. Ratnasamy, Routebricks: Exploiting parallelism to scale software routers, in: Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles, Big Sky, Montana, USA, ACM, 2009, pp. 15–28, http://dx.doi.org/10.1145/1629575.1629578.

[120] S. Han, K. Jang, K. Park, S. Moon, Packetshader: A GPU-accelerated software router, in: Proceedings of the ACM SIGCOMM 2010 Conference, New Delhi, India, ACM, 2010, pp. 195–206, http://dx.doi.org/10.1145/1851182.1851207.

[121] Barefoot Tofino, https://www.barefootnetworks.com/products/brief-tofino-2/.

[122] PostgreSQL, https://www.postgresql.org/.

[123] TimescaleDB, https://www.timescale.com/.

[124] N. Van Tu, J. Hyun, J.W.-K. Hong, Towards ONOS-based SDN monitoring using in-band network telemetry, in: Proceedings of the 2017 19th Asia-Pacific Network Operations and Management Symposium, Seoul, South Korea, IEEE, 2017, pp. 76–81, http://dx.doi.org/10.1109/APNOMS.2017.8094182.

[125] Y. Zhu, N. Kang, J. Cao, A. Greenberg, G. Lu, R. Mahajan, D. Maltz, L. Yuan, M. Zhang, B.Y. Zhao, H. Zheng, Packet-level telemetry in large datacenter networks, ACM SIGCOMM Comput. Commun. Rev. 45 (4) (2015) 479–491, http://dx.doi.org/10.1145/2829988.2787483.

[126] N. Kagami, R.I.T. da Costa Filho, L.P. Gaspary, CAPEST: Offloading network capacity and available bandwidth estimation to programmable data planes, IEEE Trans. Netw. Serv. Manag. 17 (1) (2020) 175–189, http://dx.doi.org/10.1109/TNSM.2019.2934316.

[127] In-band network function telemetry.

[128] N. Choi, L. Jagadeesan, Y. Jin, N.N. Mohanasamy, M.R. Rahman, K. Sabnani, M. Thottan, Run-time performance monitoring, verification, and healing of end-to-end services, in: Proceedings of the 2019 IEEE Conference on Network Softwarization, Paris, France, IEEE, 2019, pp. 30–35, http://dx.doi.org/10.1109/NETSOFT.2019.8806660.

[129] P.H. Isolani, J. Haxhibeqiri, I. Moerman, J. Hoebeke, J.M. Marquez-Barja, L.Z. Granville, S. Latré, An SDN-based framework for slice orchestration using in-band network telemetry in IEEE 802.11, in: Proceedings of the 2020 IEEE Conference on Network Softwarization, Ghent, Belgium, 2020, pp. 1–3, http://dx.doi.org/10.1109/NetSoft48620.2020.9165358.

[130] R. Hohemberger, A.G. Castro, F.G. Vogt, R.B. Mansilha, A.F. Lorenzon, F.D. Rossi, M.C. Luizelli, Orchestrating in-band data plane telemetry with machine learning, IEEE Commun. Lett. 23 (12) (2019) 2247–2251, http://dx.doi.org/10.1109/LCOMM.2019.2946562.

[131] N. Handigol, B. Heller, V. Jeyakumar, D. Mazières, N. McKeown, I know what your packet did last hop: Using packet histories to troubleshoot networks, in: Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation, USA, USENIX Association, 2014, pp. 71–85, URL https://www.usenix.org/system/files/conference/nsdi14/nsdi14-paper-handigol.pdf.

[132] Z. Xia, J. Bi, Y. Zhou, C. Zhang, Keysight: A scalable troubleshooting platform based on network telemetry, in: Proceedings of the Symposium on SDN Research, Los Angeles, CA, USA, ACM, 2018, http://dx.doi.org/10.1145/3185467.3190787.

[133] C. Jia, T. Pan, Z. Bian, X. Lin, et al., Rapid detection and localization of gray failures in data centers via in-band network telemetry, in: NOMS 2020, Budapest, Hungary, IEEE, 2020, pp. 1–9, http://dx.doi.org/10.1109/NOMS47738.2020.9110326.

[134] P. Tammana, R. Agarwal, M. Lee, Cherrypick: Tracing packet trajectory in software-defined datacenter networks, in: Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research, Santa Clara, California, ACM, 2015, http://dx.doi.org/10.1145/2774993.2775066.

[135] S.-Y. Wang, Y.-R. Chen, J.-Y. Li, H.-W. Hu, J.-A. Tsai, Y.-B. Lin, A bandwidth-efficient INT system for tracking the rules matched by the packets of a flow, in: Proceedings of the 2019 IEEE Global Communications Conference, Waikoloa, HI, USA, USA, IEEE, 2019, pp. 1–6, http://dx.doi.org/10.1109/GLOBECOM38437.2019.9013581.

[136] Y. Li, R. Miao, H.H. Liu, Y. Zhuang, F. Feng, L. Tang, Z. Cao, M. Zhang, F. Kelly, M. Alizadeh, M. Yu, HPCC: High precision congestion control, in: Proceedings of the ACM Special Interest Group on Data Communication, Beijing, China, ACM, 2019, pp. 44–58, http://dx.doi.org/10.1145/3341302.3342085.

[137] V. Jeyakumar, M. Alizadeh, C. Kim, D. Mazières, Tiny packet programs for low-latency network control and monitoring, in: Proceedings of the Twelfth ACM Workshop on Hot Topics in Networks, College Park, Maryland, ACM, 2013, pp. 1–7, http://dx.doi.org/10.1145/2535771.2535780.

[138] N. Katta, A. Ghag, M. Hira, I. Keslassy, A. Bergman, C. Kim, J. Rexford, Clove: Congestion-aware load balancing at the virtual edge, in: Proceedings of the 13th International Conference on Emerging Networking EXperiments and Technologies, Incheon, Republic of Korea, ACM, 2017, pp. 323–335, http://dx.doi.org/10.1145/3143361.3143401.

[139] C. Cascone, D. Sanvito, L. Pollini, A. Capone, B. Sanso, Fast failure detection and recovery in SDN with stateful data plane, Int. J. Netw. Manage. 27 (2) (2017) e1957, http://dx.doi.org/10.1002/nem.1957.

[140] N. Katta, M. Hira, C. Kim, A. Sivaraman, J. Rexford, HULA: Scalable load balancing using programmable data planes, in: Proceedings of the Symposium on SDN Research, Santa Clara, CA, USA, ACM, 2016, http://dx.doi.org/10.1145/2890955.2890968.

[141] N. Yaseen, J. Sonchack, V. Liu, Synchronized network snapshots, in: Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication, Budapest, Hungary, ACM, 2018, pp. 402–416, http://dx.doi.org/10.1145/3230543.3230552.

[142] J. Hyun, J.W.-K. Hong, Knowledge-defined networking using in-band network telemetry, in: Proceedings of the 2017 19th Asia-Pacific Network Operations and Management Symposium, Seoul, South Korea, IEEE, 2017, pp. 54–57, http://dx.doi.org/10.1109/APNOMS.2017.8094178.

[143] H. Yao, T. Mai, X. Xu, P. Zhang, M. Li, Y. Liu, NetworkAI: An intelligent network architecture for self-learning control strategies in software defined networks, IEEE Internet Things J. 5 (6) (2018) 4319–4327, http://dx.doi.org/10.1109/JIOT.2018.2859480.

[144] R. Joshi, T. Qu, M.C. Chan, B. Leong, B.T. Loo, BurstRadar: Practical real-time microburst monitoring for datacenter networks, in: Proceedings of the 9th Asia-Pacific Workshop on Systems, Jeju Island, Republic of Korea, 2018, pp. 1–8, http://dx.doi.org/10.1145/3265723.3265731.

[145] Y. Tang, Y. Wu, G. Cheng, Z. Xu, Intelligence enabled SDN fault localization via programmable in-band network telemetry, in: Proceedings of the 2019 IEEE 20th International Conference on High Performance Switching and Routing, Xi'An, China, IEEE, 2019, pp. 1–6, http://dx.doi.org/10.1109/HPSR.2019.8808121.

[146] P. Taffet, J. Mellor-Crummey, Lightweight, packet-centric monitoring of network traffic and congestion implemented in P4, in: Proceedings of the 2019 IEEE Symposium on High-Performance Interconnects, Santa Clara, CA, USA, IEEE, 2019, pp. 54–58, http://dx.doi.org/10.1109/HOTI.2019.00026.

[147] B. Turkovic, F. Kuipers, N. van Adrichem, K. Langendoen, Fast network congestion detection and avoidance using P4, in: Proceedings of the 2018 Workshop on Networking for Emerging Applications and Technologies, Budapest, Hungary, ACM, 2018, pp. 45–51, http://dx.doi.org/10.1145/3229574.3229581.

[148] M. Alizadeh, T. Edsall, S. Dharmapurikar, R. Vaidyanathan, K. Chu, A. Fingerhut, V.T. Lam, F. Matus, R. Pan, N. Yadav, et al., CONGA: Distributed congestion-aware load balancing for datacenters, in: Proceedings of the 2014 ACM Conference on SIGCOMM, Chicago, Illinois, USA, 2014, pp. 503–514, http://dx.doi.org/10.1145/2619239.2626316.

[149] F. Cugini, P. Gunning, F. Paolucci, P. Castoldi, A. Lord, P4 in-band telemetry (INT) for latency-aware VNF in metro networks, in: Proceedings of the Optical Fiber Communication Conference, San Diego, CA, USA, Optical Society of America, 2019, pp. M3Z–6, http://dx.doi.org/10.1364/OFC.2019.M3Z.6.

[150] H. Singh, C. Huang, M. Sicard-Gagne, G. Shami, M. Lyonnais, D. Fedorov, R. Wilson, INT-SDN: Evaluation of various P4 parameters using optical telemetry having reconfigurable data plane on 40 Gbps line rate, in: Proceedings of the 2019 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, Victoria, BC, Canada, IEEE, 2019, pp. 1–6, http://dx.doi.org/10.1109/PACRIM47961.2019.8985117.

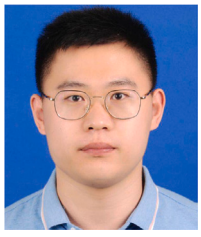
[151] J. Kong, B. Niu, S. Tang, Y. Li, H. Fang, W. Lu, Z. Zhu, Network nervous system: When multilayer telemetry meets AI-assisted service provisioning, in: Proceedings of the 2019 18th International Conference on Optical Communications and Networks, Huangshan, China, IEEE, 2019, pp. 1–3, http://dx.doi.org/10.1109/ICOCN.2019.8934891.

[152] C. Ranaweera, E. Wong, C. Lim, A. Nirmalathas, Next generation optical-wireless converged network architectures, IEEE Netw. 26 (2) (2012) 22–27, http://dx.doi.org/10.1109/MNET.2012.6172271.

[153] S. Narayana, M. Tahmasbi, J. Rexford, D. Walker, Compiling path queries, in: Proceedings of the 13th USENIX Symposium on Networked Systems Design and Implementation, Santa Clara, CA, 2016, pp. 207–222, http://dx.doi.org/10.5555/2930611.2930626.

[154] Q. Huang, H. Sun, P. P. C. Lee, W. Bai, F. Zhu, Y. Bao, Omnimon: Re-architecting network telemetry with resource efficiency and full accuracy, in: Proceedings of the 2020 ACM Conference on SIGCOMM, ACM, Virtual Event, USA, 2020, pp. 404–421, http://dx.doi.org/10.1145/3387514.3405877.

[155] X. Pan, S. Tang, S. Liu, J. Kong, X. Zhang, D. Hu, J. Qi, Z. Zhu, Privacy-preserving multilayer in-band network telemetry and data analytics: For safety, please do not report plaintext data, J. Lightwave Technol. 38 (21) (2020) 5855–5866, http://dx.doi.org/10.1109/JLT.2020.3007491.

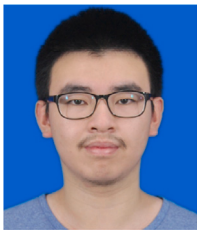


Lizhuang Tan received his B.S. degree in Communication Engineering from College of Information Science and Engineering, Shandong Normal University, Ji'nan, P.R. China in 2017. He is currently pursuing his Ph.D. degree at the National Engineering Laboratory for Next Generation Internet Interconnection Devices (NGIID), Beijing Jiaotong University, P.R. China. He is a student member of IEEE and CCF. His research interests include software-defined networking (SDN) and deterministic network (DetNet).

Wei Su received his B.S., M.S., and Ph.D. degrees in communication and information systems from Beijing Jiaotong University, P. R. China in 2001, 2004 and 2008. He is a full professor at the School of Electronic and Information Engineering, Beijing Jiaotong University. He has studied the Internet for more than 20 years. He was selected for the Beijing Young Talents Program in 2013. He won the second prize of National Technology Invention Award in 2014. He was a visiting scholar with Future University, Hakodate, Japan in 2015. He has published more than 50 research papers and 4 monographs in the areas of communications and computer networks. His research interests are next generation network and Mobile Internet.

Wei Zhang received the B.E. degree from Zhejiang University in 2004, the M.S. degree from Liaoning University in 2008, and the Ph.D. degree from Shandong University of Science and Technology in 2018. He is currently an Associate Professor with the Shandong Computer Science Center (National Supercomputer Center in Ji'nan), Qilu University of Technology (Shandong Academy of Sciences). His research interests include future generation network architectures, edge computing and edge intelligence.

Jianhui Lv received B.S. degree in mathematics and applied mathematics from Jilin Institute of Chemical Technology, Jilin, China in 2012, and M.S. & Ph.D degree in computer science from Northeastern University, Shenyang, China in 2014 & 2017. He is currently an assistant research fellow at Tsinghua University. His research interests include ICN, in-network caching enabled networks, IoT, bio-inspired networking, edge computing, etc. He has published more than 35 journals (such as IEEE Network, Elsevier Information Science, Elsevier COMNET, Elsevier ASOC, Elsevier JNCA and IEEE COML) and conference papers (such as IEEE INFOCOM, IEEE/ACM IWQoS and IEEE ICPADS). He has served as the leader guest editors (LGE) in four international journals (Wireless Communications and Mobile Computing, Internet Technology Letters, Recent Advances in Computer Science and Communications, and International Journal of Distributed Systems and Technologies).



Zhenyi Zhang received his B.S. degree in Communication Engineering from Beijing Jiaotong University, P.R. China in 2019. Now he is pursuing her M.S. degree at National Engineering Laboratory for Next Generation Internet Interconnection Devices (NGIID), Beijing Jiaotong University. His research interests are software-defined networking and data plane programmable technology.

Jingying Miao received her B.S. degree in Communication Engineering from Beijing Jiaotong University, P.R. China in 2019. Now she is pursuing her M.S. degree at National Engineering Laboratory for Next Generation Internet Interconnection Devices (NGIID), Beijing Jiaotong University. She has extensive P4 programming experience. Her research interests are software-defined networking (SDN) and data plane programmable technology.

Xiaoxi Liu received her B.S. degree in Communication Engineering from Beijing Jiaotong University, P.R. China in 2020. Now she is pursuing her M.S. degree at National Engineering Laboratory for Next Generation Internet Interconnection Devices (NGIID), Beijing Jiaotong University. She is going to pursue M.S. degree at Beijing Jiaotong University. Her research interests is software-defined networking (SDN).

Na Li graduated from College of Information Science and Engineering, Ocean University of China with her M.S. degree in communication and information system in 2019. Now she works at the Shandong Branch of National Computer network Emergency Response technical Team/Coordination Center(CNCERT/SD). Her research interests are 5G, software-defined networking (SDN) and communication security.