The Latency Characteristics of GTP-U and SRv6 Stateless Translation on VPP Software Router

Chunghan Lee1), Naoyuki Mori2) Yasuhiro Ohara3)
Tetsuya Murakami4), Shogo Asaba5), Satoru Matsushima6)

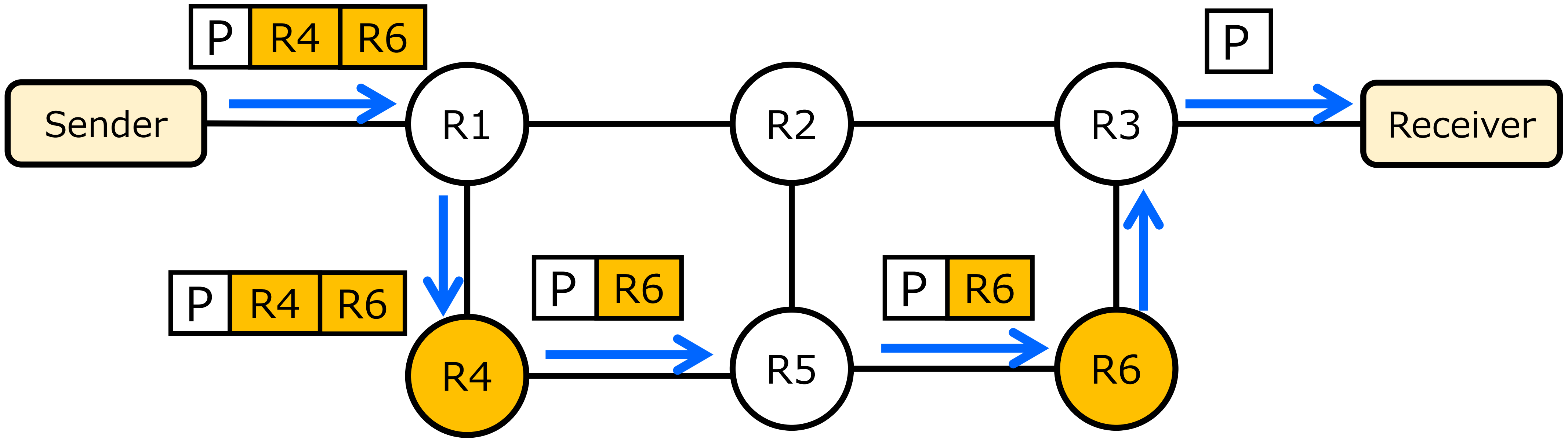
IEEE ADMNET 2021

July.12th.2021

Toyota Motor Corporation1), Intel Corporation2), NTT Communications Corporation3)
Arrcus, Inc.4), NEC corporation5), SoftBank Corp.6)

Introduction

- Segment routing IPv6 (SRv6)
  - Encode the path in each packet
  - The SRv6 based on source routing has many advantages: stateless traffic steering, state reduction, network programming, and so on



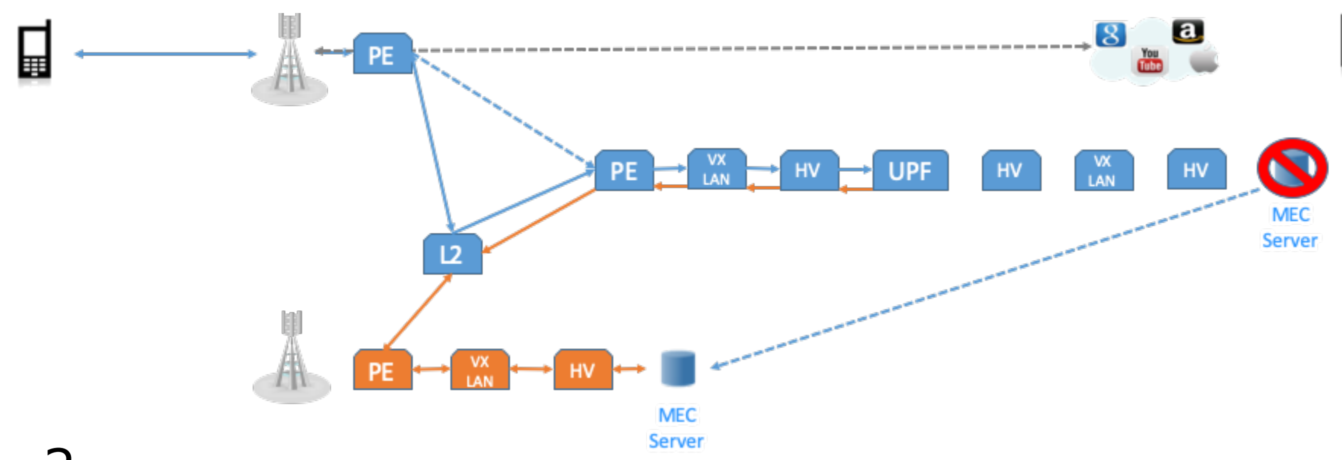
The SRv6 is promising for the reduction of complexity and dependency on mobile network

Actual use case

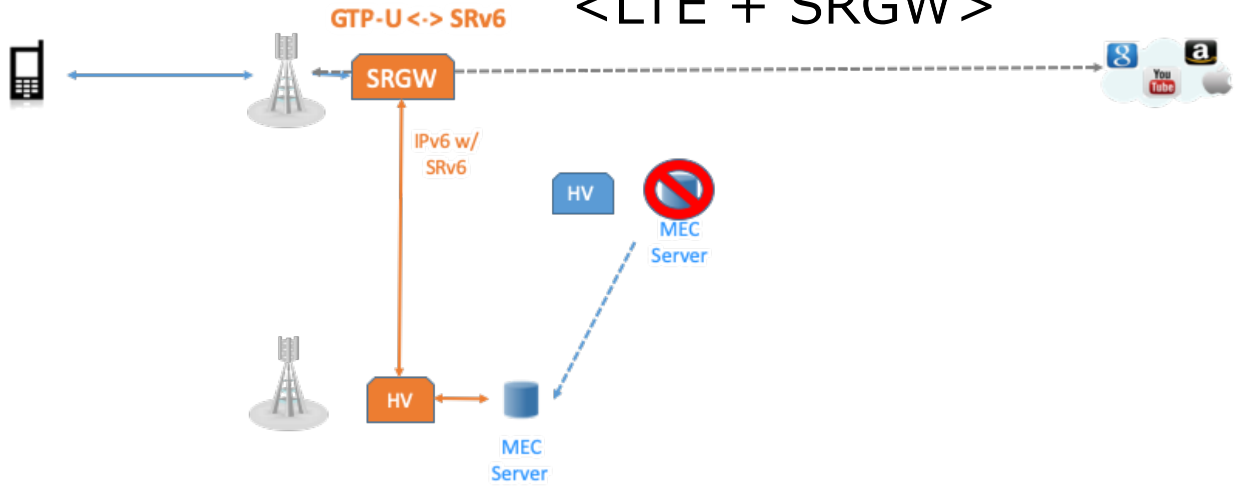
GTP-U/SRv6 translation for mobile network

- Multiple stacking headers on the data plane of mobile network
  - Mobile devices are only communicated with UPF (GTP-U) on LTE network
  - After network failure, the efficient network path would not be selected due to the state of stacking headers
- Segment routing gateway (SRGW)
  - The stacking headers can be embedded to IPv6 address space and Segment Routing Header (SRH) and the efficient path can be selected

<Current LTE >

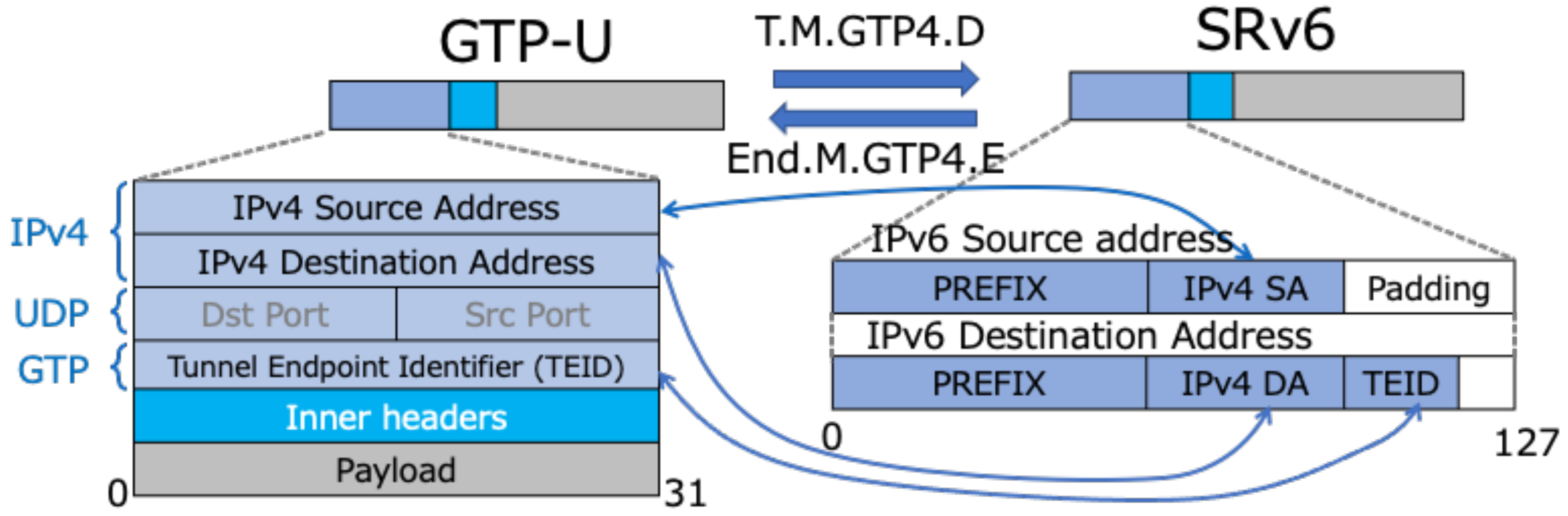


<LTE + SRGW>



What is GTP-U and SRv6 Stateless Translation?

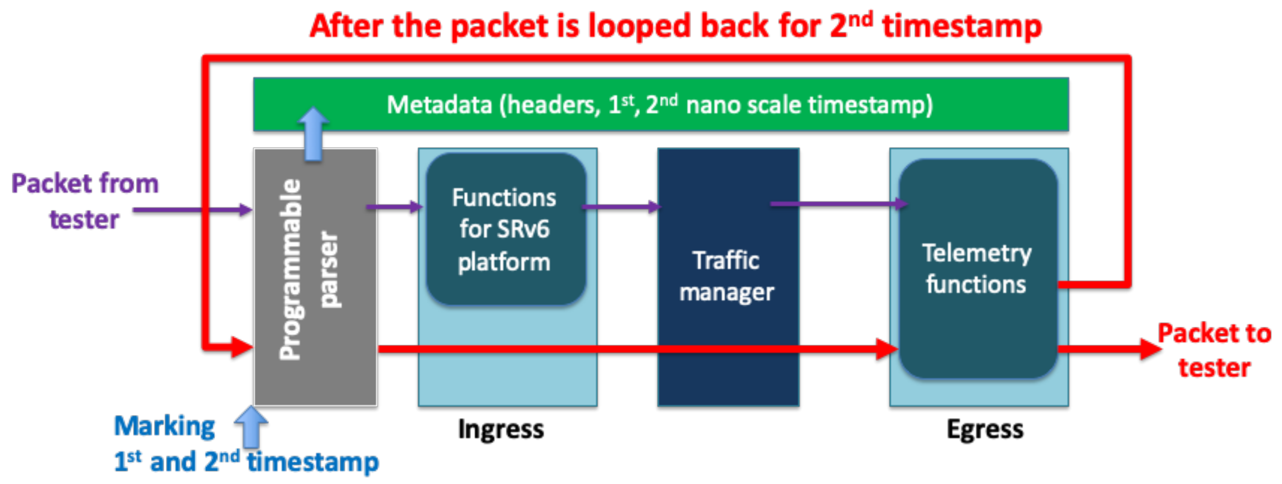
- All identifiers of a GTP-U tunnel (IPv4 addresses and TEID) can be embedded into the IPv6 address space
  - We can also use the SRH for multiple GTP-U headers
- The translation method (SRv6 for Mobile UserPlane) has been proposed in IETF
  - https://datatracker.ietf.org/doc/html/draft-ietf-dmm-srv6-mobile-uplane-13



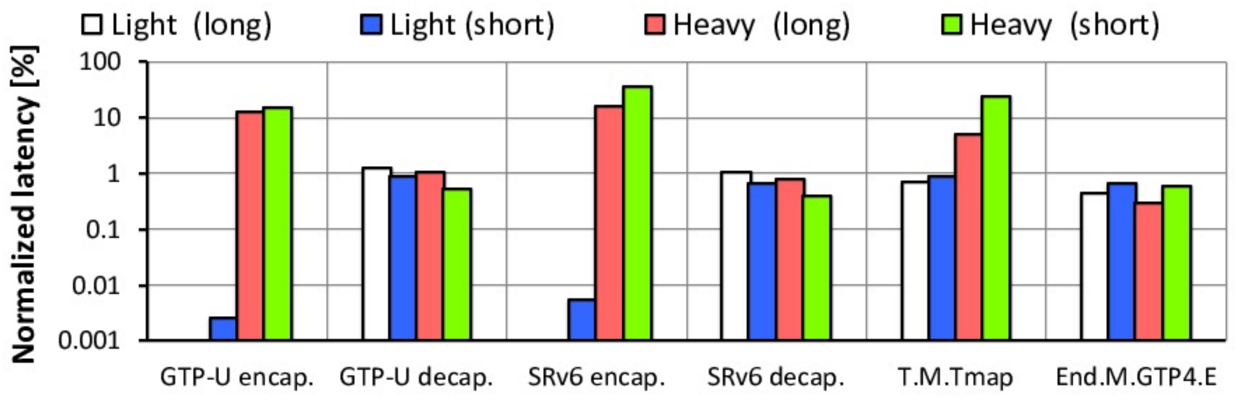
Our previous work[1]…

- We implemented the translation functions on the H/W P4 switch and measured the latency on the H/W P4 switch

However, the software-based SRv6 Mobile UserPlane would be better to deploy Edge/MEC or local 5G network
(We consider other platforms, such as an DPDK-based software router)


After the packet is looped back for 2nd timestamp
Metadata (headers, 1st, 2nd nano scale timestamp)
Packet from tester
Programmable parser
Functions for SRv6 platform
Traffic manager
Telemetry functions
Packet to tester
Marking 1st and 2nd timestamp
Ingress
Egress

< SRv6 functions on P4 switch >



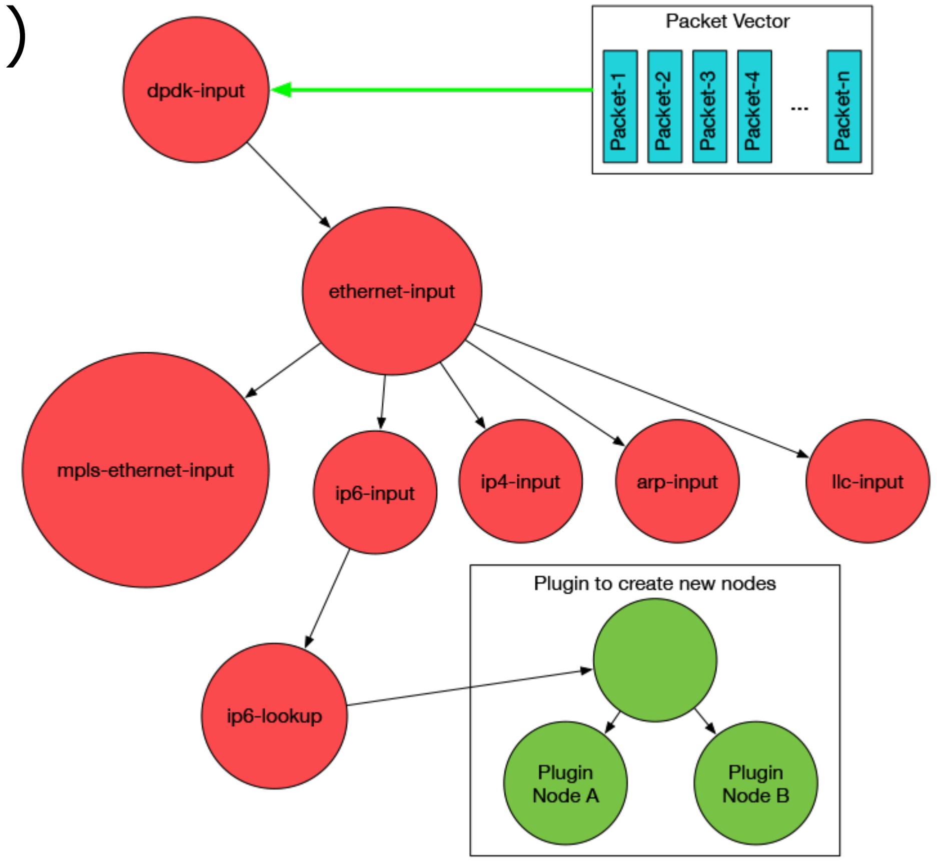
< Normalized latency at SR functions >

1) "Performance Evaluation of GTP-U and SRv6 Stateless Translation", 2nd Workshop on Segment Routing and Service Function Chaining (SR+SFC 2019)

DPDK-based software router (VPP)

FD.io VPP

- It is based on Vector Packet Processing (VPP) and an DPDK-based software router
  - Multiple packets are processed at one batch processing
  - The batch processing is called as ***vectors/call***
- It is an open source software (https://fd.io/)
  - It is easy to extend additional network functions (e.g., SRv6 functions)
- The VPP router is integrated to ODL, OPNFV, OpenStack, and K8s
  - It is also used as LB in production (Yahoo! Japan)
- **There are no evaluation results on the latency of the SRv6 for Mobile UserPlane**



< VPP graph nodes >

We select the VPP router as our measurement platform for the translation latency

Research goal

- Evaluate the quantitative performance of SRv6 Mobile User Plane on the VPP software router

- Observe the latency characteristics on the VPP software router

How to measure the accurate latency on the VPP software router?

1. How can we accurately measure the latency when a receiving packet type is changed from the corresponding packet type sent?

   → **Use the nano scale timestamp injection in the P4 switch with a pair of unique inner headers**

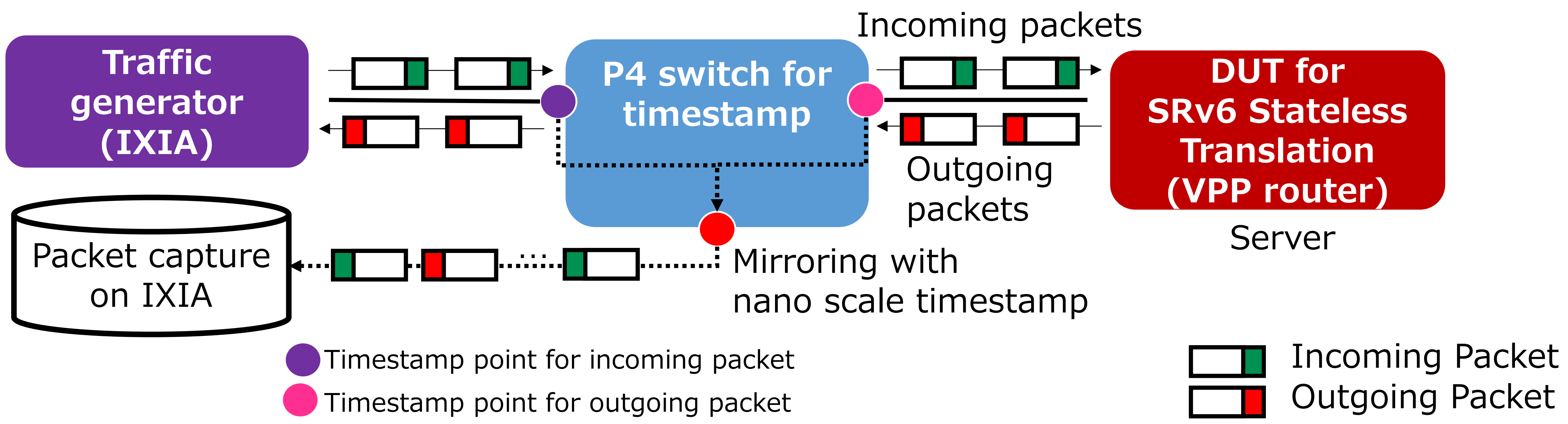
2. What kind of latency characteristics do we observe on the software router?

   → **Analyze evaluation results in detail**

Overview

Experiments on VPP software router

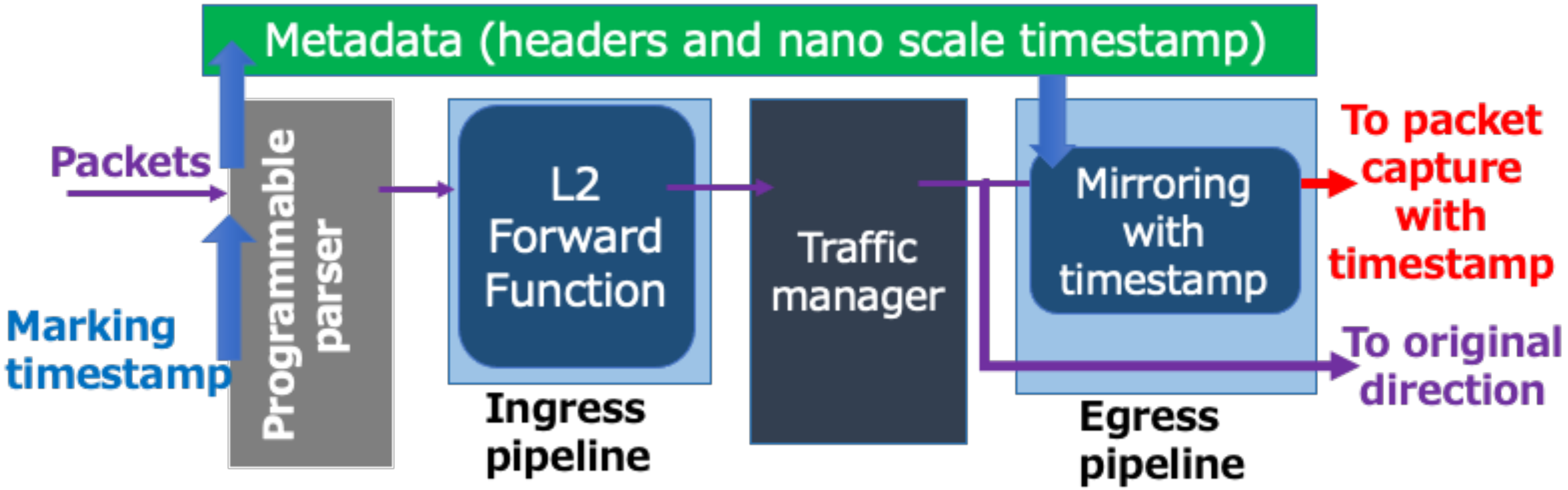
- SRv6 translation functions are used for the latency measurement
  - **DUT** : the VPP software router
  - **Traffic generator (IXIA)** : a hardware-based commercial traffic generator (40Gbps)
  - **P4 switch** : write a packet timestamp to source mac address at mirrored packets
  - **Packet capture** : capture the mirrored packets without packet loss


Traffic generator (IXIA)
P4 switch for timestamp
Incoming packets
DUT for SRv6 Stateless Translation (VPP router)
Outgoing packets
Server
Packet capture on IXIA
Mirroring with nano scale timestamp
Timestamp point for incoming packet
Timestamp point for outgoing packet
Incoming Packet
Outgoing Packet

Accurate latency measurement (P4 switch)

Nano scale timestamp using P4 switch

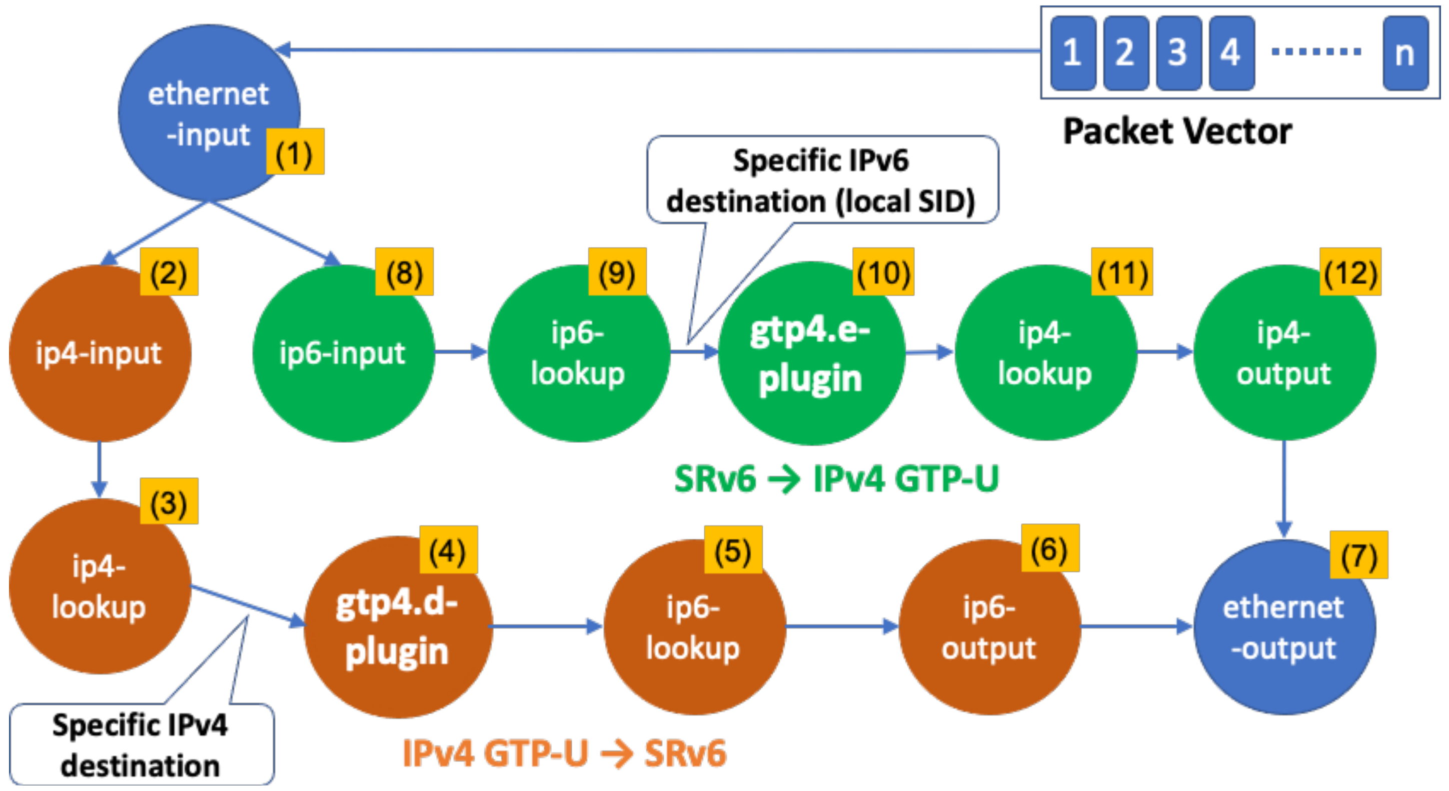
- During the packet mirroring, the nano scale timestamp is stored in the source MAC address field
  - The P4 switch overwrites and abandons the original source MAC address



VPP software router

GTP-U/SRv6 translation functions

- VPP graph nodes for GTP-U/SRv6 stateless translation
  - There are two IP address lookups from IPv4 to IPv6 (vice versa)
    - **GTP-U→SRv6 (*gtp4.d-plugin (4)*)** : translates a GTP-U over IPv4 to an SRv6
    - **SRv6→GTP-U (*gtp4.e-plugin (10)*)** : translates an SRv6 to a GTP-U over IPv4
  - The translation functions are already merged to the VPP master branch**



** https://github.com/FDio/vpp

Measurement scenarios

- We prepared the following conditions for the accurate latency measurement
  - One CPU core for packet processing thread with one NIC port
  - Input traffic loads : 1Mpps ~ 5Mpps (NO drop condition)
  - Two stateless translation functions
    - **GTP-U→SRv6 (*T.M.GTP4.D*)**
    - **SRv6→GTP-U (*End.M.GTP4.E*)**
  - Three types of packet sizes
    - The short size : no payload
    - The middle size : intermediate size with the payload
    - The long size : the throughput (5 Mpps) is equal to the link capacity (40 Gbps)

	Short [bytes]	Middle [bytes]	Long [bytes]
GTP-U (IPv4)	94	508	976
SRv6 (IPv6)	98	512	980

<The packet sizes for the translation latency measurement>

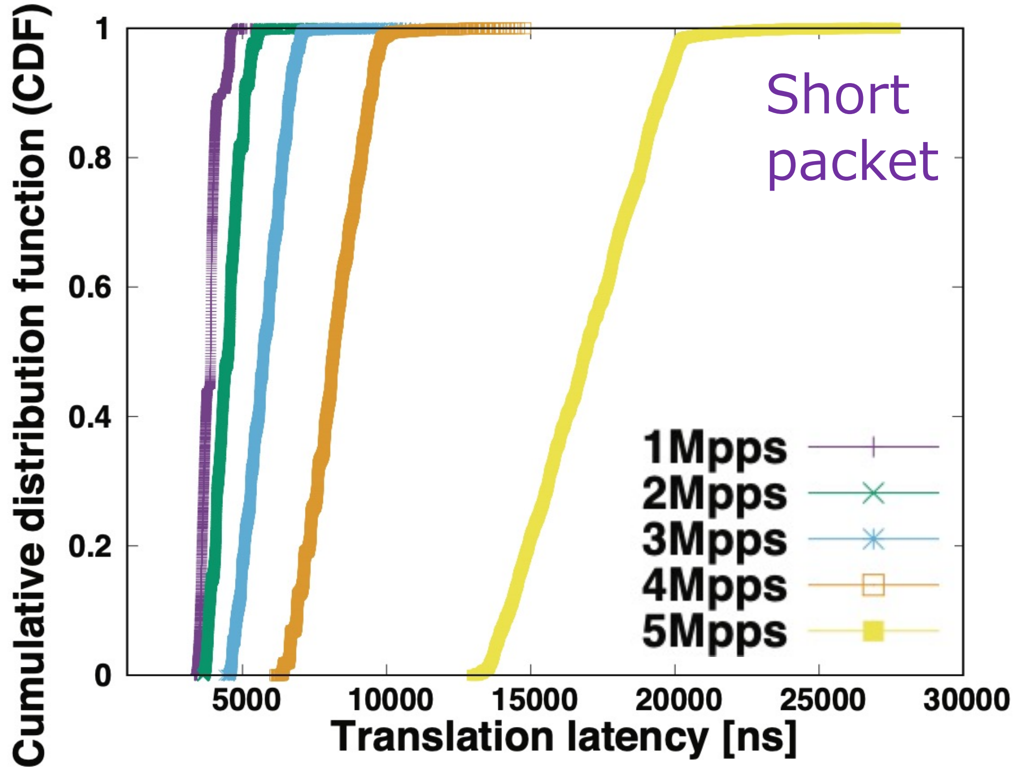
Latency measurement results on VPP router

The translation latency

- The translation latency is in the feasible range (**roughly 3–30 μs**)
- As the packet size is increased from the short to the long, the latency tends to increase slightly
- We also observe that the input load (Mpps) impacts the latency

	Input load [**1Mpps**]			Input load [**5Mpps**]		
	Min [μs]	Mean [μs]	Max [μs]	Min [μs]	Mean [μs]	Max [μs]
Short	3.3	3.9	8.5	12.9	17.0	27.6
Middle	3.5	4.1	8.7	13.8	17.5	27.5
Long	3.8	4.3	8.9	15.5	19.5	29.0

<The statistics of translation latency(SRv6→GTP-U)>

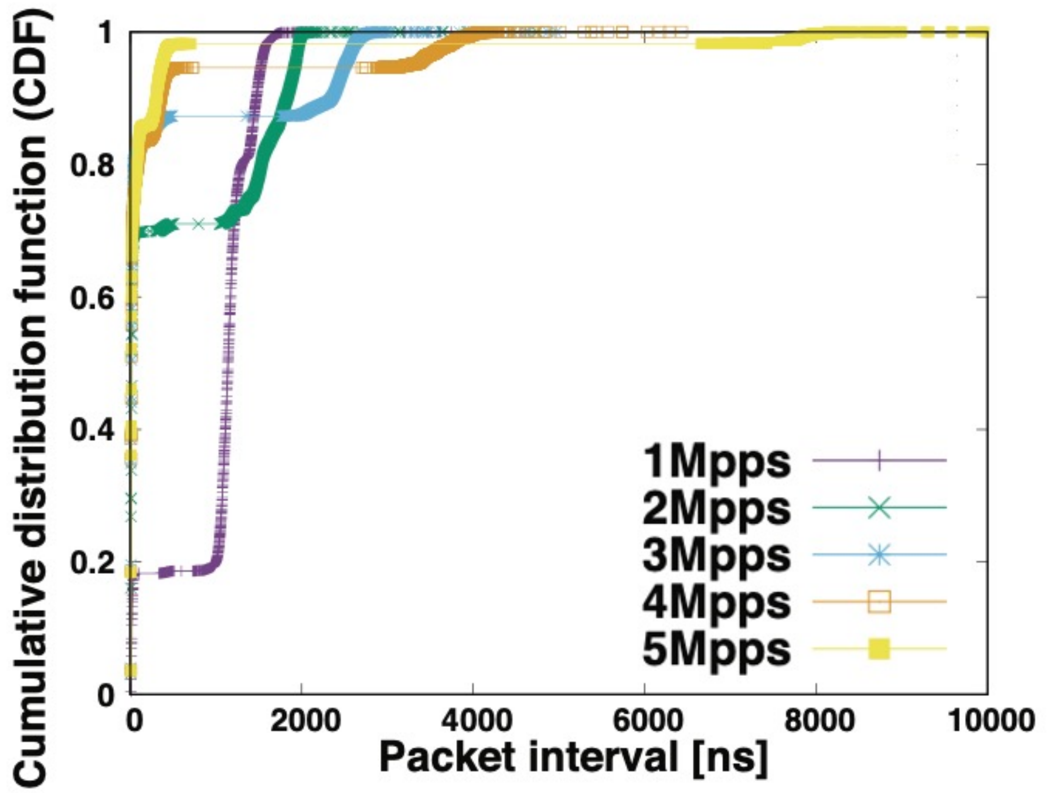


<The CDF of translation latency(SRv6→GTP-U)>

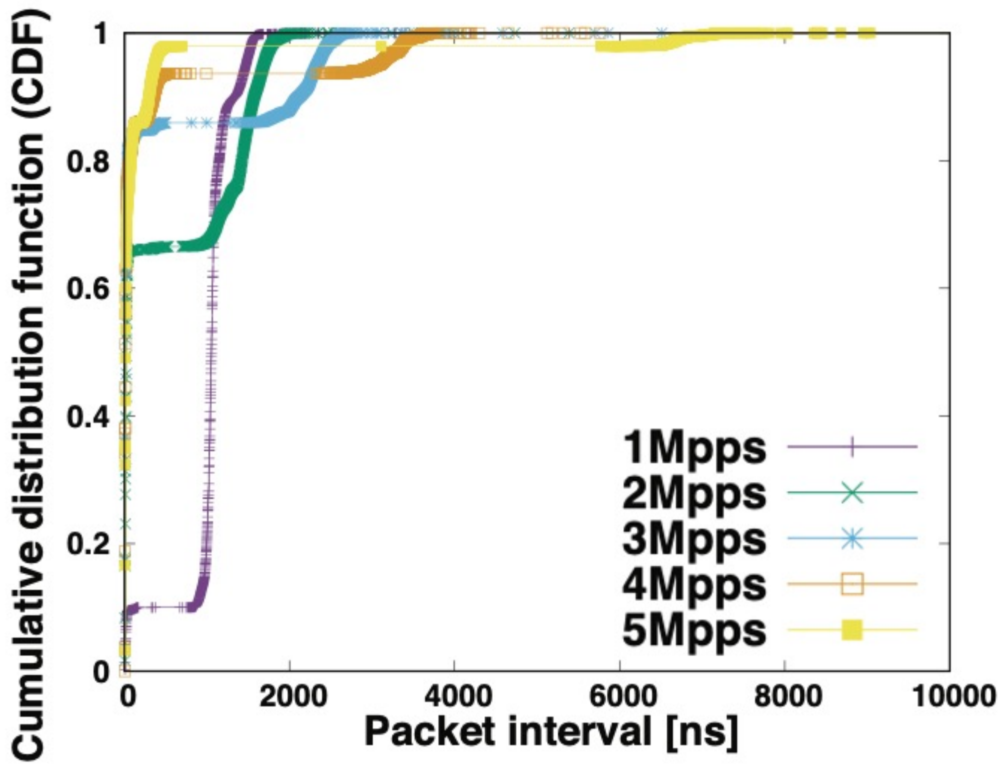
Latency measurement results on VPP router

Outgoing packet interval of translation latency

- The packet interval is a time period between packets on the VPP router
- When the input traffic load is increased to 5 Mpps, multiple packets are simultaneously queued and they are processed in a bulk manner
  - The approximately 80% packet intervals are close to zero (a few nanoseconds)



<The CDF of packet interval (GTP-U→SRv6)>



<The CDF of packet interval (SRv6→GTP-U)>

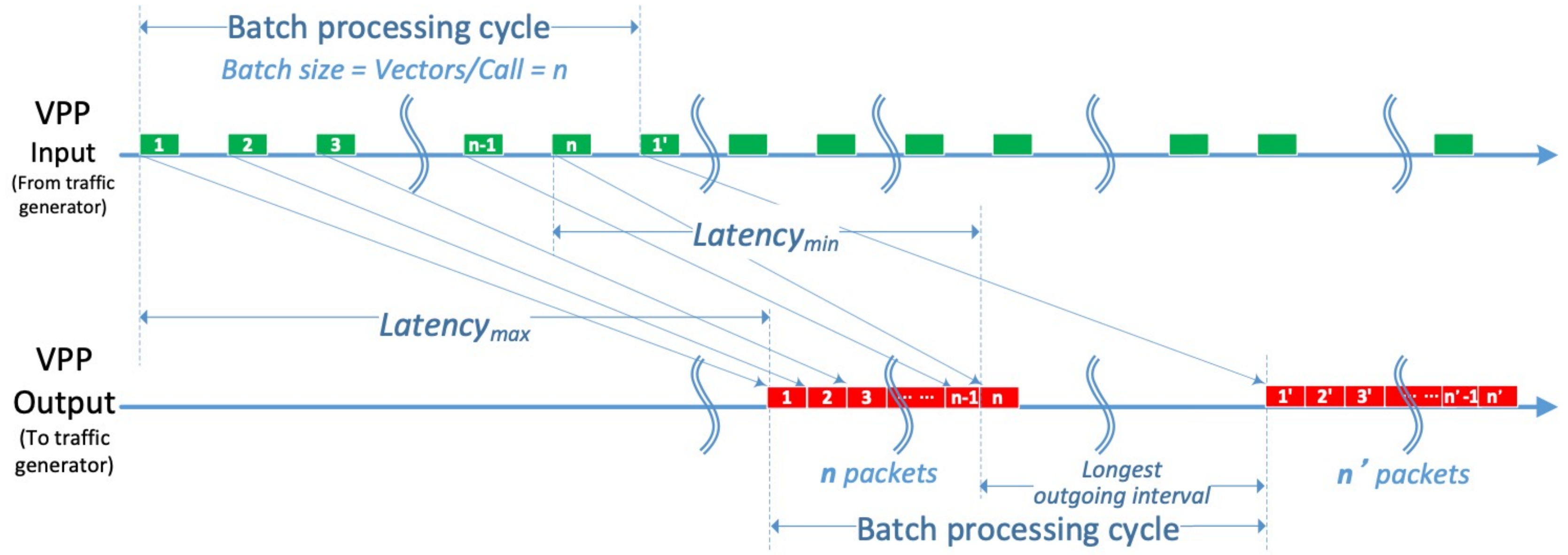
Latency characteristics on VPP router

Batch processing model

- ***Vectors/Call*** : how many packets are processed in one batch processing cycle
- The jitter on the batch model is defined as follow

$$Jitter = Latency_{max} - Latency_{min}$$

  - To eliminate the unexpected latency fluctuation, we determine 95%tile latency (***p95***) as $Latency_{max}$ and 5%tile latency (***p5***) as $Latency_{min}$
  - The jitter depends on the *Vectors/Call*



<Input/output latency model on batch processing>

Latency characteristics on VPP router

Contribution factors of *Vectors/Call* – (1)

- We calculated the correlation coefficient (***r***) between the *Vectors/Call* and the following contribution factors
  - We mainly present the contribution factor (1)

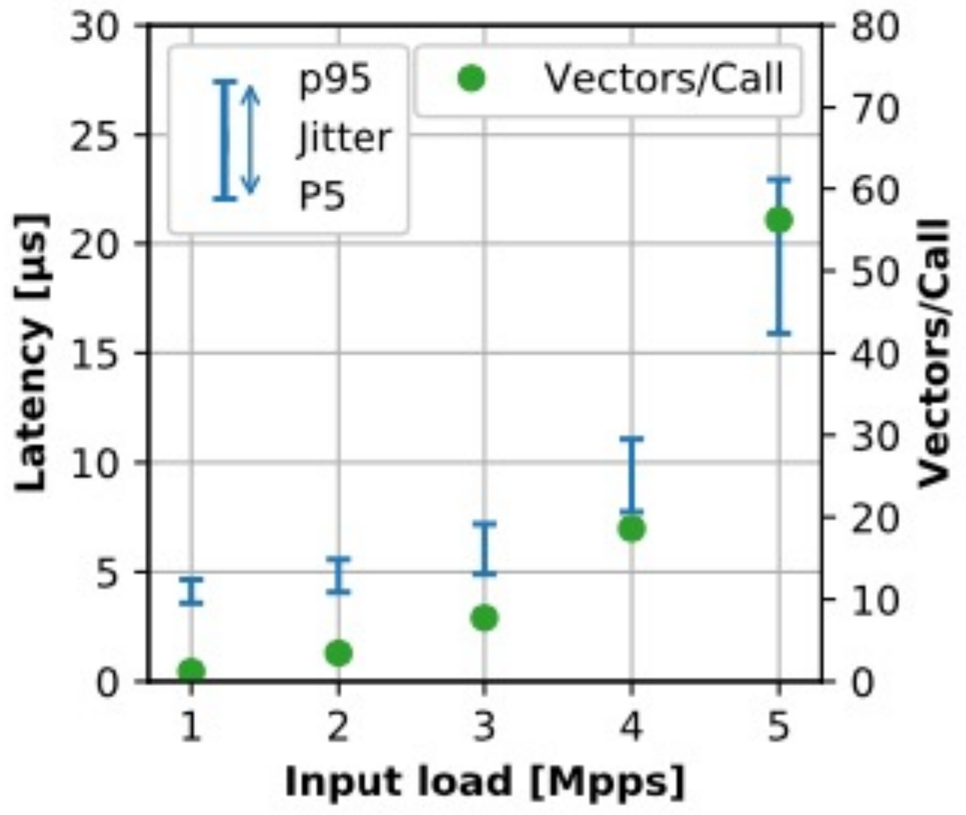
A list of contribution factor	**Relationship using (*r*)**
1.Input traffic load	Strong
2.CPU frequency	Strong
3.Translation function type	Strong
4.Packet sizes	No relationship

Refer to the contribution factors in the paper ☺

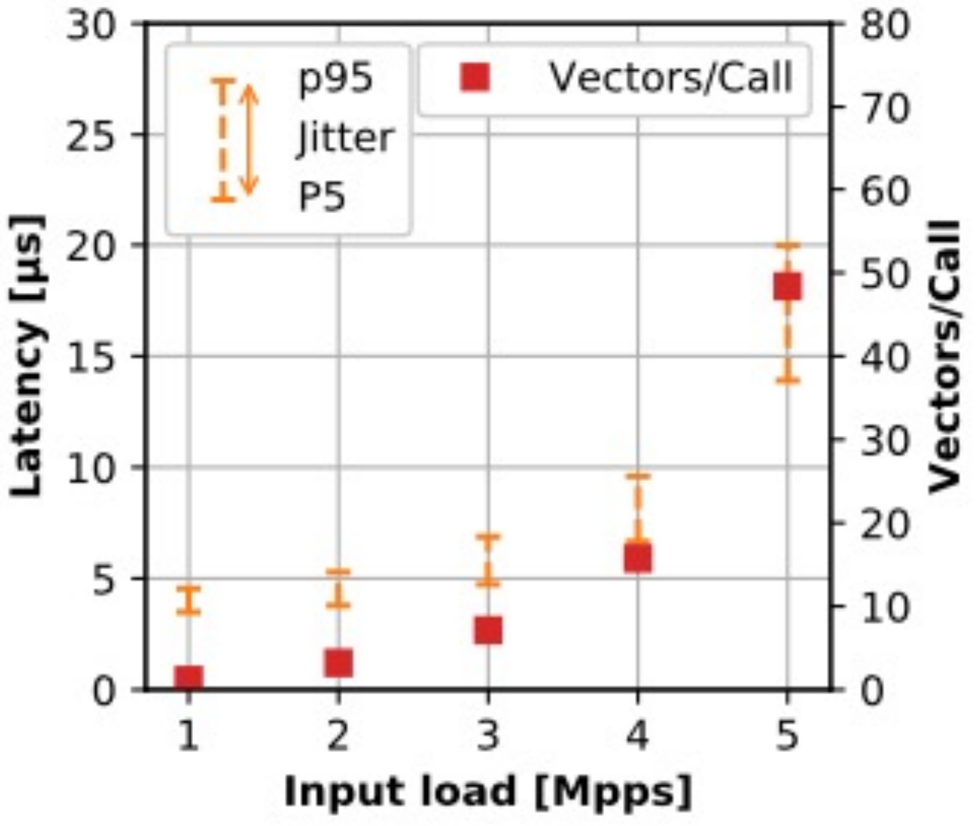
Latency characteristics on VPP router

Contribution factors of *Vectors/Call* – (2)

- Relation between the *Vectors/call* and the input traffic load
  - As the input traffic load linearly increases (e.g., 1 - 5 Mpps), the *Vectors/Call* and the jitter seem to increase exponentially



< GTP-U→SRv6 (short packet) >

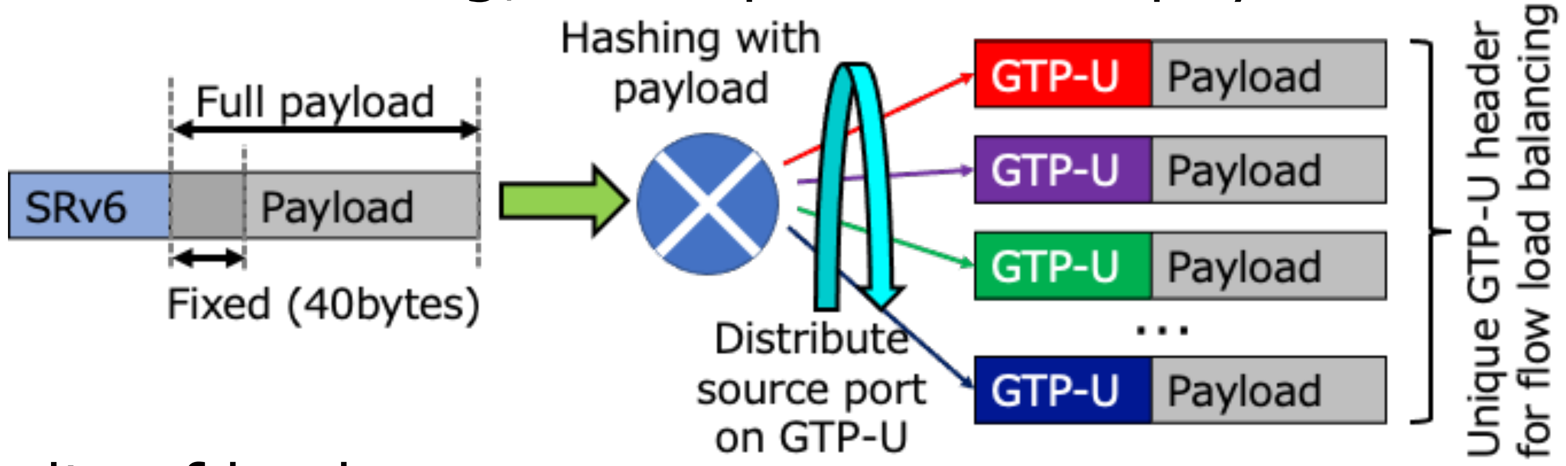


< SRv6→ GTP-U (short packet) >

Experiences on VPP router

The effect of hash space

- For flow load balancing, we implement the payload hashing



- The results of hash space
  - We encountered the significant performance degradation on the middle and long sizes
  - We fixed the hashing space as 40 bytes (The inner IPv4 and TCP headers only)
    - The CPU time of hash function is maintained as approximately 15%
    - The source code is already contributed to the main official VPP (v20.05.1)

	Short packet	Middle packet	Long packet	**Fixed (40 bytes)**
Traffic load [Mpps]	5.00	2.72	1.69	**5.00**
CPU time ratio of hash function using *Intel VTune* [%]	15.1	55.7	72.9	**15**

Conclusion and future work

- **Conclusion**
  - Evaluate the latency of translation functions on the VPP router using nano-scale measurement with P4 switch
  - **Latency characteristics**
    - The translation latency is in the feasible range (**roughly 3–30 μs**)
    - The batch processing impacts the latency characteristics and the *Vectors/Call* is a key factor
    - The three major contribution factors of *Vectors/Call*:

      Input traffic load, CPU frequency, and translation function types
  - Our findings can help to improve the performance of software-based network system and to design beyond-5G mobile network systems

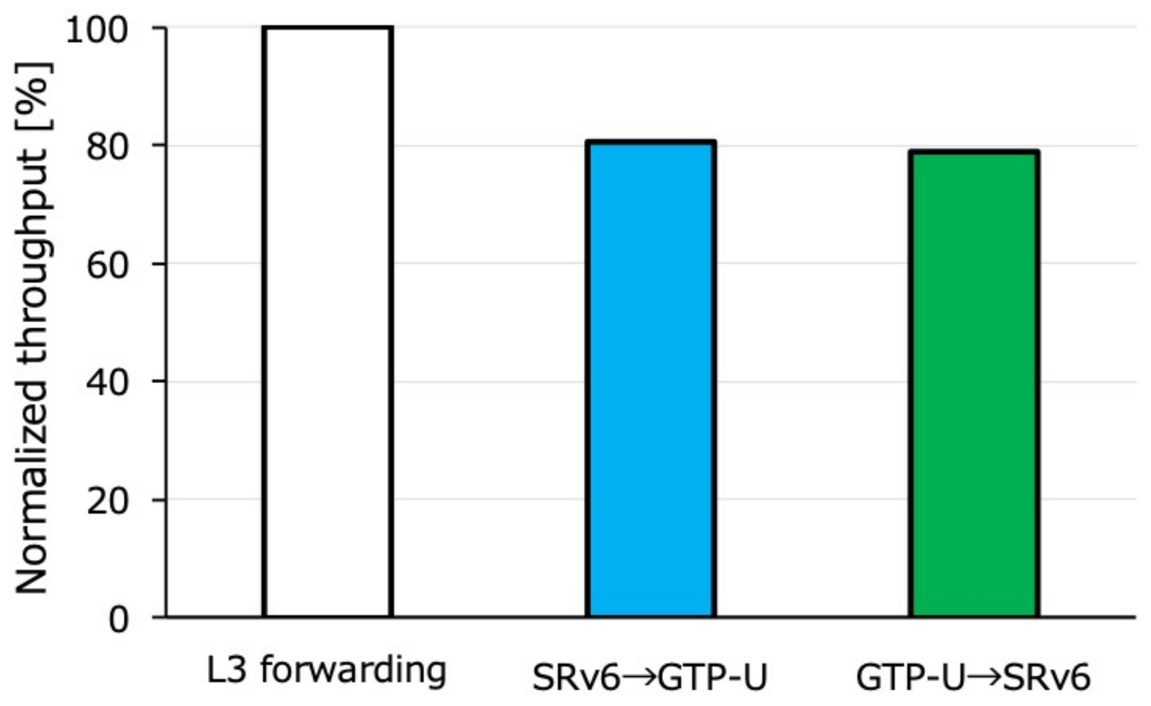
- **Future work**
  - Measure the latency on more complicated cases, such as multiple routing entries and multiple CPU cores at maximum performance
  - Evaluate the translation latency on other software routers

Backup slide

L3 forwarding vs Translation

Comparison of maximum throughput [Mpps]

- Measure the maximum throughput (L3 forwarding [IPv4] vs Translation)
  - There is **NO** packet loss and the single CPU core is used
  - L3 forwarding [IPv4]: 6.7 Mpps (100%)
  - SRv6/GTP-U Translation : 5.4 Mpps (79.1%)
  - GTP-U/SRv6 Translation : 5.3 Mpps (80.5%)

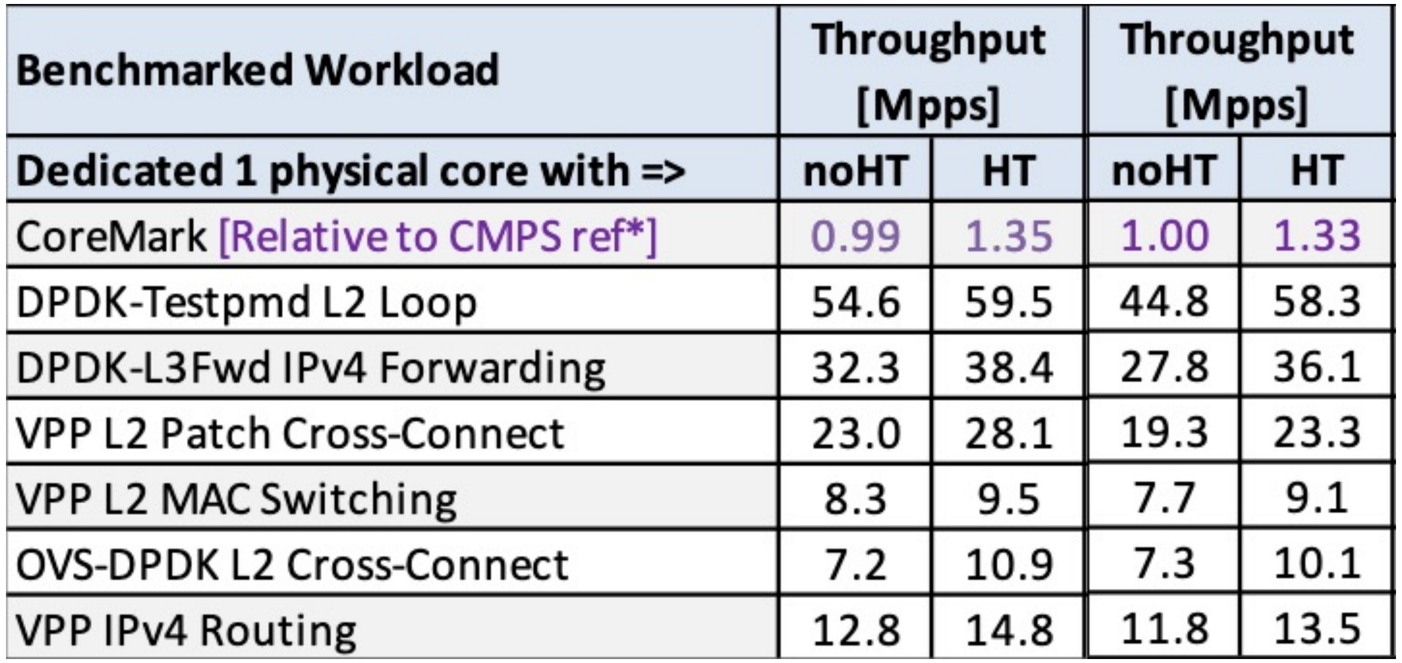


The DPDK-based software router (VPP)

Performance measurement

- Although there are extensive results on throughput, there seem to be little latency evaluation on the VPP software router
- Moreover, there are no evaluation results on latency for the SRv6 for Mobile UserPlane

Skylake *Broadwell*



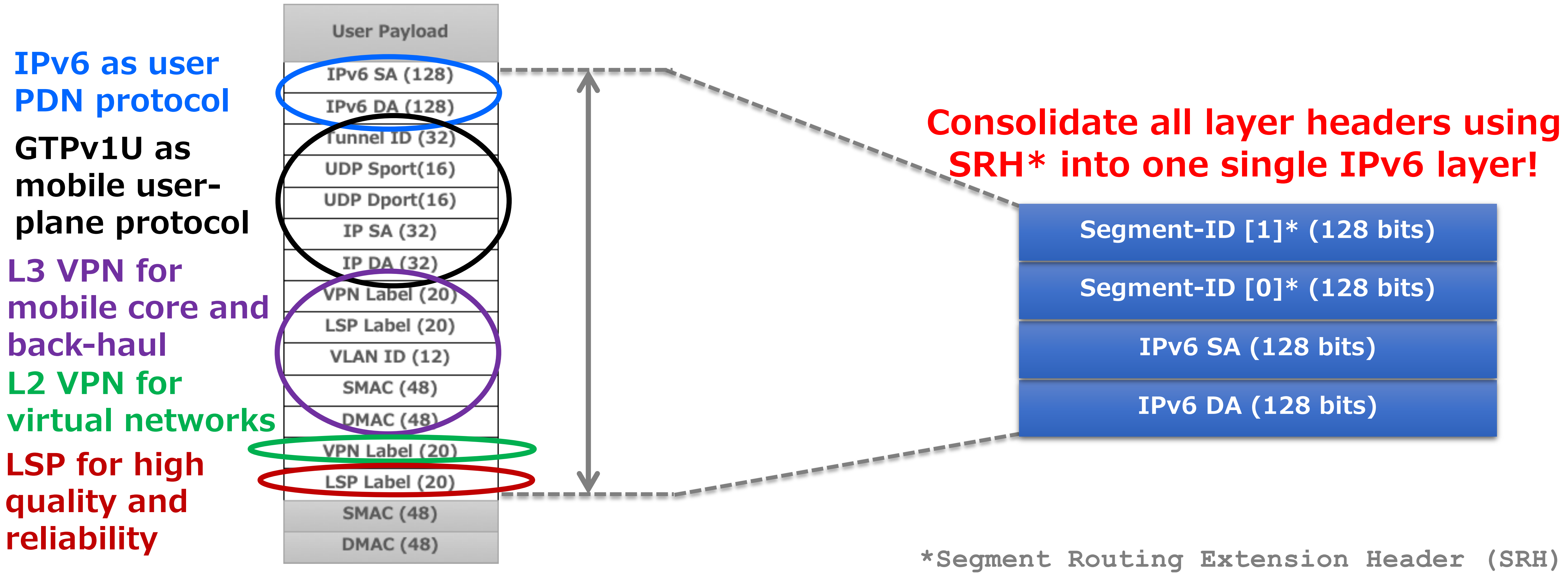
Benchmarked Workload	Throughput [Mpps]		Throughput [Mpps]	
Dedicated 1 physical core with =>	**noHT**	**HT**	**noHT**	**HT**
CoreMark [Relative to CMPS ref*]	0.99	1.35	1.00	1.33
DPDK-Testpmd L2 Loop	54.6	59.5	44.8	58.3
DPDK-L3Fwd IPv4 Forwarding	32.3	38.4	27.8	36.1
VPP L2 Patch Cross-Connect	23.0	28.1	19.3	23.3
VPP L2 MAC Switching	8.3	9.5	7.7	9.1
OVS-DPDK L2 Cross-Connect	7.2	10.9	7.3	10.1
VPP IPv4 Routing	12.8	14.8	11.8	13.5

< Throughput[2] of network applications >

2) Benchmarking Software Data Planes Intel® Xeon® Skylake vs. Broadwell

Why the translation is required?

- Multiple stacking headers on the data plane of mobile network
  - Stacking multiple small IDs to fulfill the requirements of reliability
  - After network failure, the efficient network path would not be selected due to the state of stacking headers


IPv6 as user PDN protocol
GTPv1U as mobile user-plane protocol
L3 VPN for mobile core and back-haul
L2 VPN for virtual networks
LSP for high quality and reliability
User Payload
IPv6 SA (128)
IPv6 DA (128)
Tunnel ID (32)
UDP Sport(16)
UDP Dport(16)
IP SA (32)
IP DA (32)
VPN Label (20)
LSP Label (20)
VLAN ID (12)
SMAC (48)
DMAC (48)
VPN Label (20)
LSP Label (20)
SMAC (48)
DMAC (48)
Consolidate all layer headers using SRH* into one single IPv6 layer!
Segment-ID [1]* (128 bits)
Segment-ID [0]* (128 bits)
IPv6 SA (128 bits)
IPv6 DA (128 bits)

*Segment Routing Extension Header (SRH)

Latency measurement setup (server)

- H/W specification
  - One CPU core is used for latency measurement

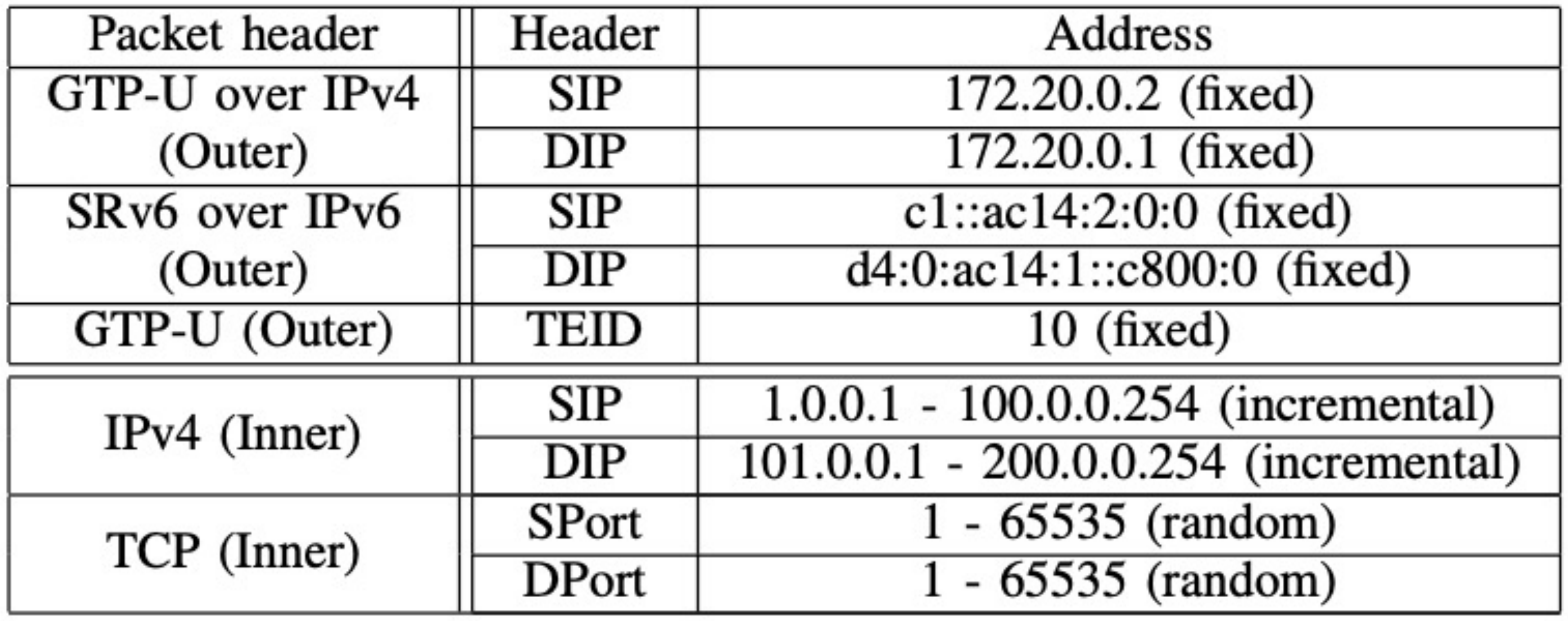
CPU	Intel Xeon Gold 6126 (2.6 GHz) [19.25 MB L3 cache]
Memory	384 GB
NIC	Mellanox ConnectX-4
Bandwidth	40 Gbps

- S/W configuration
  - C state/P state are disable
  - CPU frequency is fixed (2.6 GHz)
    - Intel TurboBoost is also disable

OS	Ubuntu 18.04.2 LTS
Kernel	5.3.0-28
VPP	v20.05.1 (stable)

How to generate unique packets for latency measurement?

- We fixed outer headers and changed unique inner packet headers randomly and sequentially



Outer headers

Inner headers

Packet header	Header	Address
GTP-U over IPv4 (Outer)	SIP	172.20.0.2 (fixed)
	DIP	172.20.0.1 (fixed)
SRv6 over IPv6 (Outer)	SIP	c1::ac14:2:0:0 (fixed)
	DIP	d4:0:ac14:1::c800:0 (fixed)
GTP-U (Outer)	TEID	10 (fixed)
IPv4 (Inner)	SIP	1.0.0.1 - 100.0.0.254 (incremental)
	DIP	101.0.0.1 - 200.0.0.254 (incremental)
TCP (Inner)	SPort	1 - 65535 (random)
	DPort	1 - 65535 (random)