IPv6+ Network Architecture for Deep Space

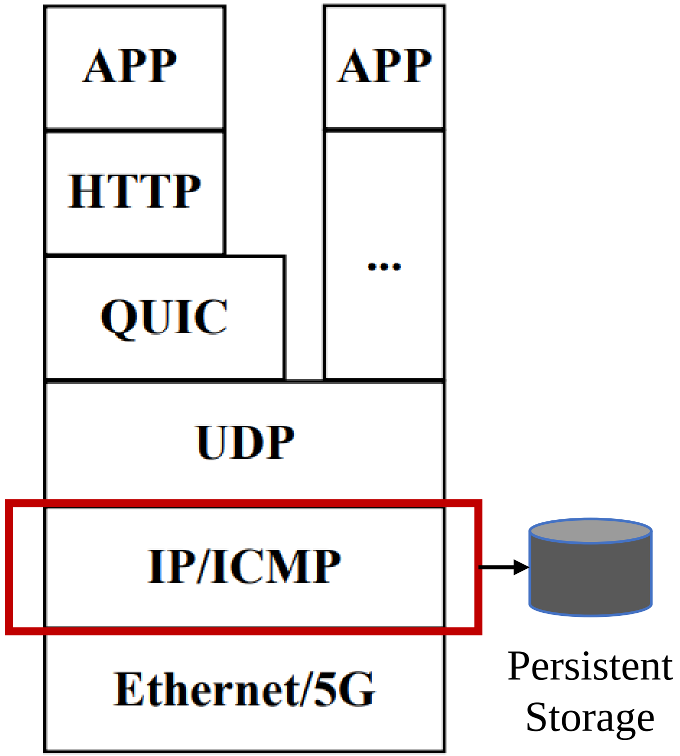
Yihan Zhu, Jun Liu, Kanglian Zhao
School of Electronic Science And Engineering
Institute of Space Terrestrial Intelligent Networks
Nanjing University, P. R. China

2024.11.04

Introduction

- At last IETF 120 Side meeting, we introduced a new **SRv6 based store-carry-and-forward networking** for deep space, and got some valuable results after our prototype validation. First, we will review the main ideas and conclusions.
- At this meeting, we will give an whole **IPv6+ network architecture** for deep space.
- First, we will introduce our **test of QUIC and SRv6 based store-carry-and-forward interoperation**;
- Next, we will present a **CGR-based routing** approach to generate an SRv6 segment list for link handover;
- At last, we will give the **new scenarios for next-generation deep space networks** and we will give the design selection why we choose IPv6+ network architecture for deep space.

Review: SRv6 Based Store-Carry-and-Forward Solution



IP layer: Extend the IP forwarding plane to **handle disruptions**.

Our solution: the store-carry-and-forward paradigm based on SRv6 at IP layer.

Main ideas:

1. Use **Segment Routing** to complete **hop by hop network path planning**;

2. Implement **hop-by-hop forwarding** through SRv6;

3. Leverage the programmable capabilities of SRv6, a new SID with store-carry-and-forward function is added for **disruption tolerance**;

4. Extend ICMPv6 message types to implement the **reliable retransmission** and deletion of stored packets over long and lossy deep space links.

Review: The store-carry-and-forward based on SRv6 paradigm

Basic SR forwarding paradigm

The store-carry-and-forward based on SRv6 paradigm :

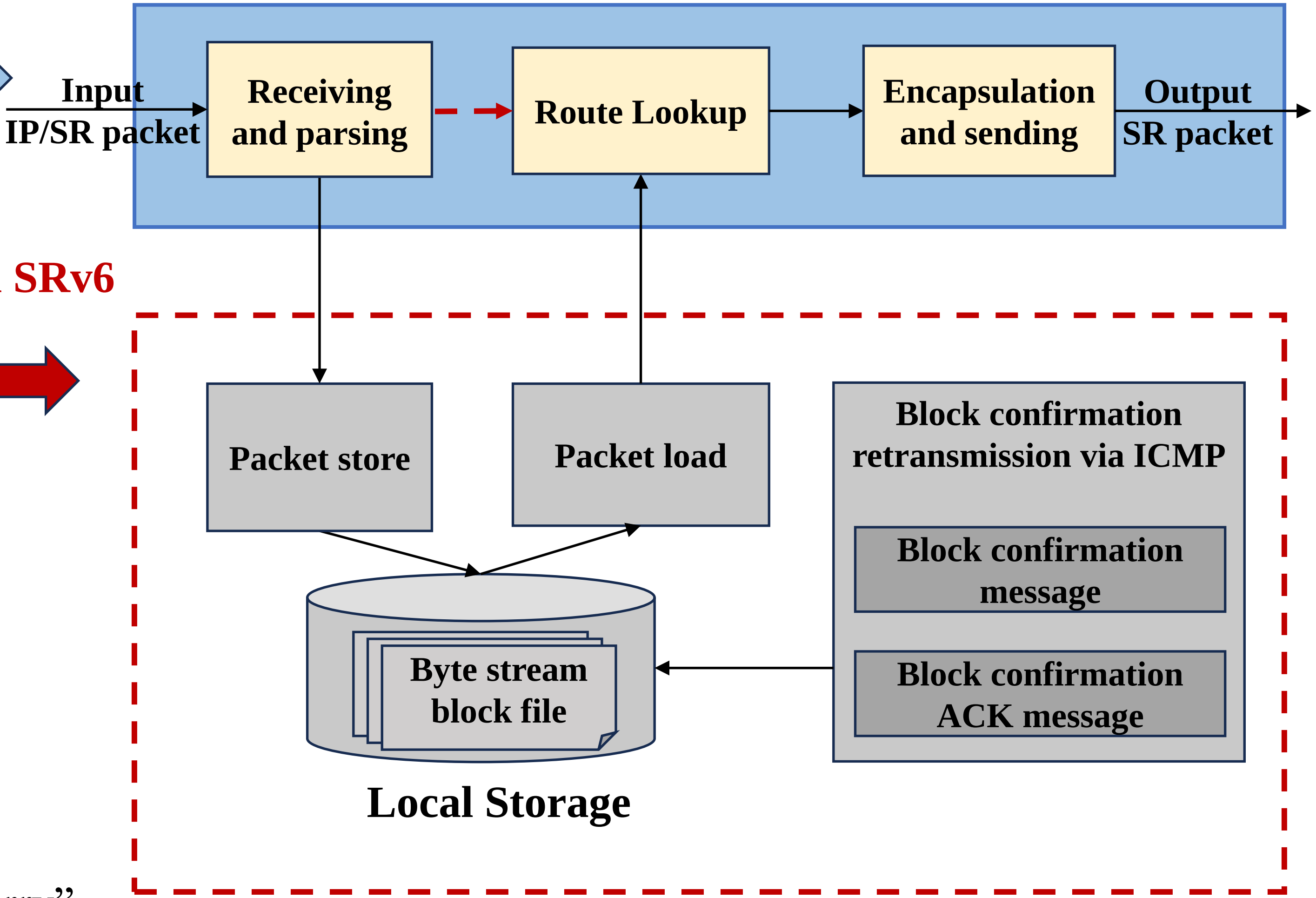
(1) Add **packet store** function

(2) Add **packet load** function

· **Packet reading drive** mechanism

· **Packet recovery** mechanism

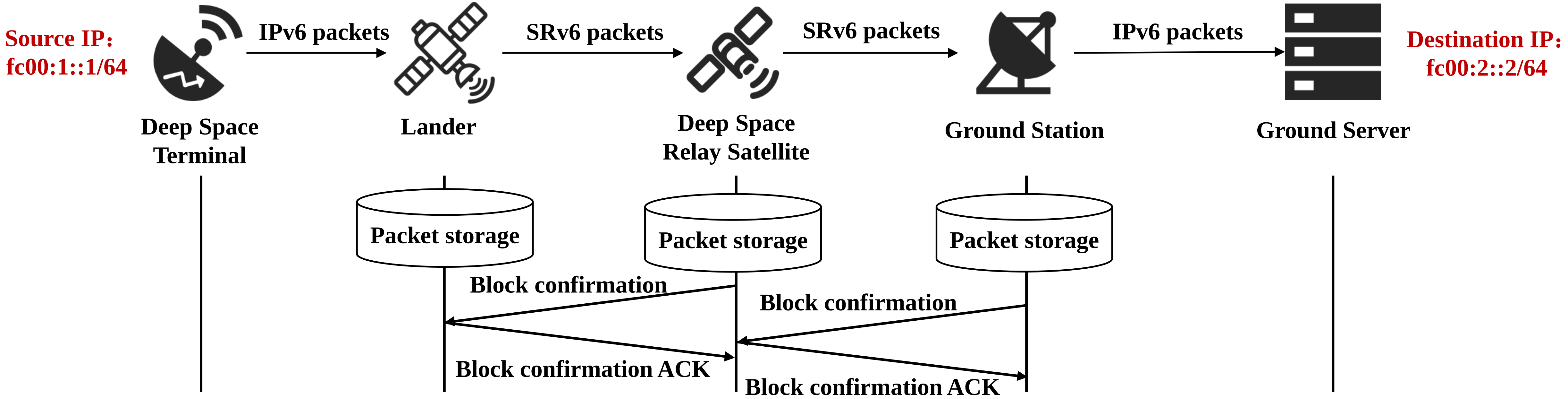
(3) Add an optional **reliable block retransmission** mechanism with **ICMP confirmation message**, to notify the "carry" node to delete the successfully sent packets



* SR packet: IPv6 packet with SRH header

Review: Implementation and Experiment

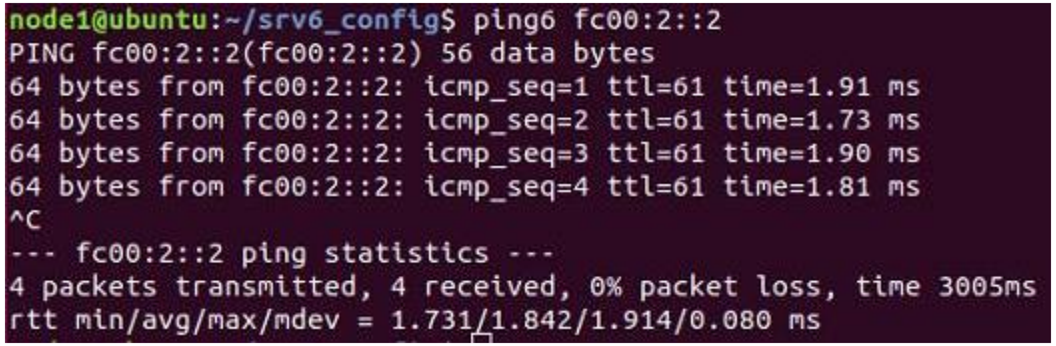
We have implemented **SRv6 Based Store-Carry-and-Forward** by using **Linux kernel** TCP/IP protocol stack codes, and completed a prototype validation.



Node	SID	Locator	Function
Ground Station	2001:DB8:300:1:300::/80	2001:DB8:300:1::/64	300
Deep Space Relay Satellite	2001:DB8:200:1:200::/80	2001:DB8:200:1::/64	200
Lander	2001:DB8:100:1:200::/80	2001:DB8:100:1::/64	200

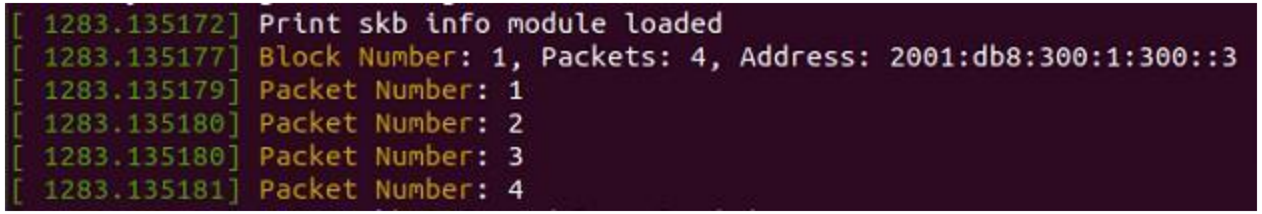
Segment List

Review: Prototype Validation



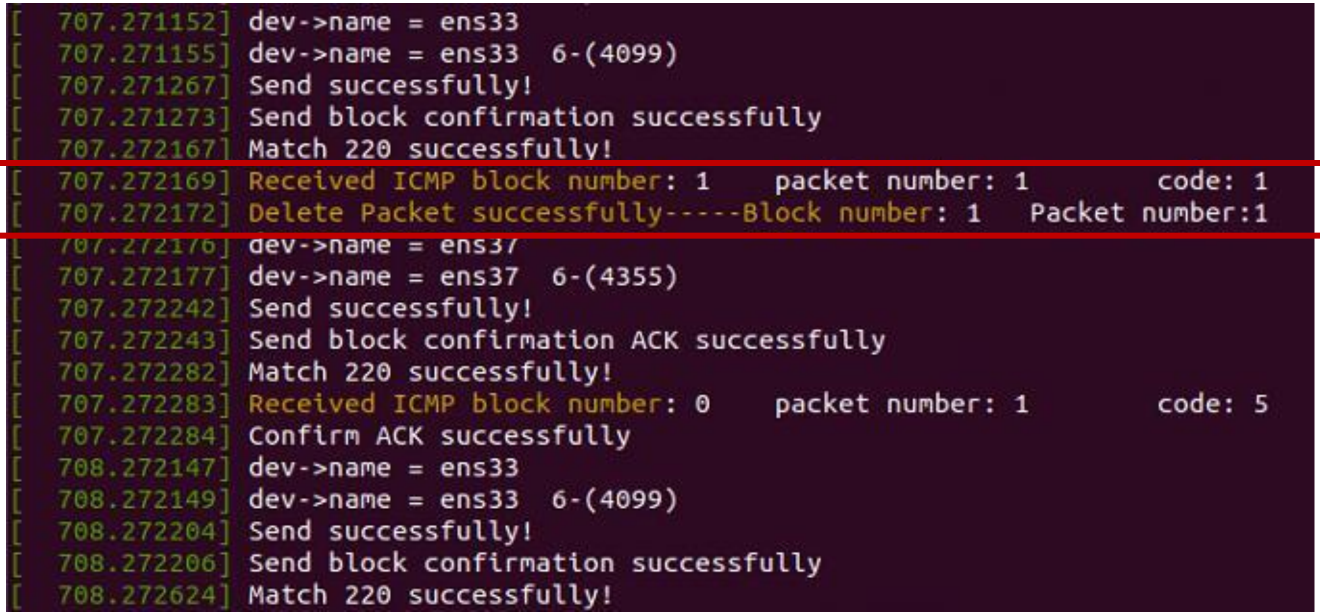
```
node1@ubuntu:~/srv6_config$ ping6 fc00:2::2
PING fc00:2::2(fc00:2::2) 56 data bytes
64 bytes from fc00:2::2: icmp_seq=1 ttl=61 time=1.91 ms
64 bytes from fc00:2::2: icmp_seq=2 ttl=61 time=1.73 ms
64 bytes from fc00:2::2: icmp_seq=3 ttl=61 time=1.90 ms
64 bytes from fc00:2::2: icmp_seq=4 ttl=61 time=1.81 ms
^C
--- fc00:2::2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 1.731/1.842/1.914/0.080 ms
```

The deep space terminal can ping the ground server successfully.



```
[ 1283.135172] Print skb info module loaded
[ 1283.135177] Block Number: 1, Packets: 4, Address: 2001:db8:300:1:300::3
[ 1283.135179] Packet Number: 1
[ 1283.135180] Packet Number: 2
[ 1283.135180] Packet Number: 3
[ 1283.135181] Packet Number: 4
```

The local storage function is implemented successfully.



```
[  707.271152] dev->name = ens33
[  707.271155] dev->name = ens33  6-(4099)
[  707.271267] Send successfully!
[  707.271273] Send block confirmation successfully
[  707.272167] Match 220 successfully!
[  707.272169] Received ICMP block number: 1    packet number: 1        code: 1
[  707.272172] Delete Packet successfully-----Block number: 1   Packet number:1
[  707.272176] dev->name = ens37
[  707.272177] dev->name = ens37  6-(4355)
[  707.272242] Send successfully!
[  707.272243] Send block confirmation ACK successfully
[  707.272282] Match 220 successfully!
[  707.272283] Received ICMP block number: 0    packet number: 1        code: 5
[  707.272284] Confirm ACK successfully
[  708.272147] dev->name = ens33
[  708.272149] dev->name = ens33  6-(4099)
[  708.272204] Send successfully!
[  708.272206] Send block confirmation successfully
[  708.272624] Match 220 successfully!
```

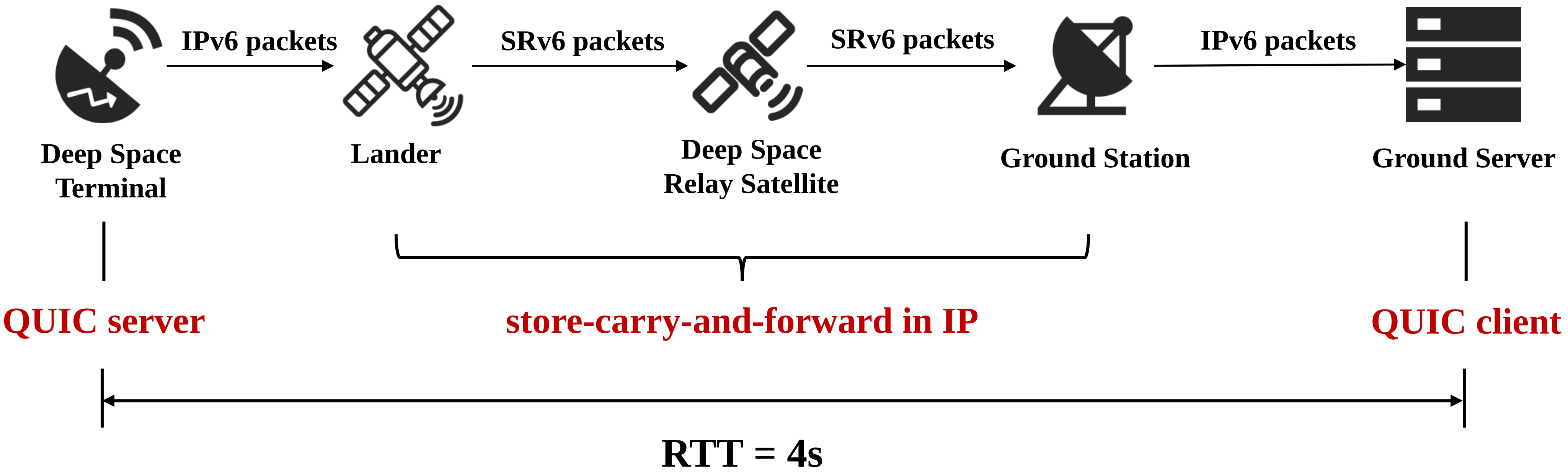
After receiving the ICMPv6 confirmation message, the local stored packets are released successfully.

Review: DTN BP/LTP vs Deep space IP

Target	DTN Solution	IP Solution
Hop-by-hop transmission	√ (BP)	√ (SRv6)
Store-Carry-and-Forward	√ (BP)	√ (SRv6 End.XS)
Confirmation retransmission	√ (LTP)	√ (ICMP)
Adaptive Routing	√ (CGR)	√ (TVR+CGR, FRR)
Distinguishing between reliable and unreliable transmission	√ (LTP Red and Green segments)	√ (ACL, Link config)
Parallel transfer	√ (LTP Session)	√ (ECMP, Multiple-threads)
Socket interface for APP	X (under development)	√
Forwarding performance	Packet processing is complex, too many Queues, larger packet headers	**Better (**Packet Processing is simple, same AQM as Standard IP)

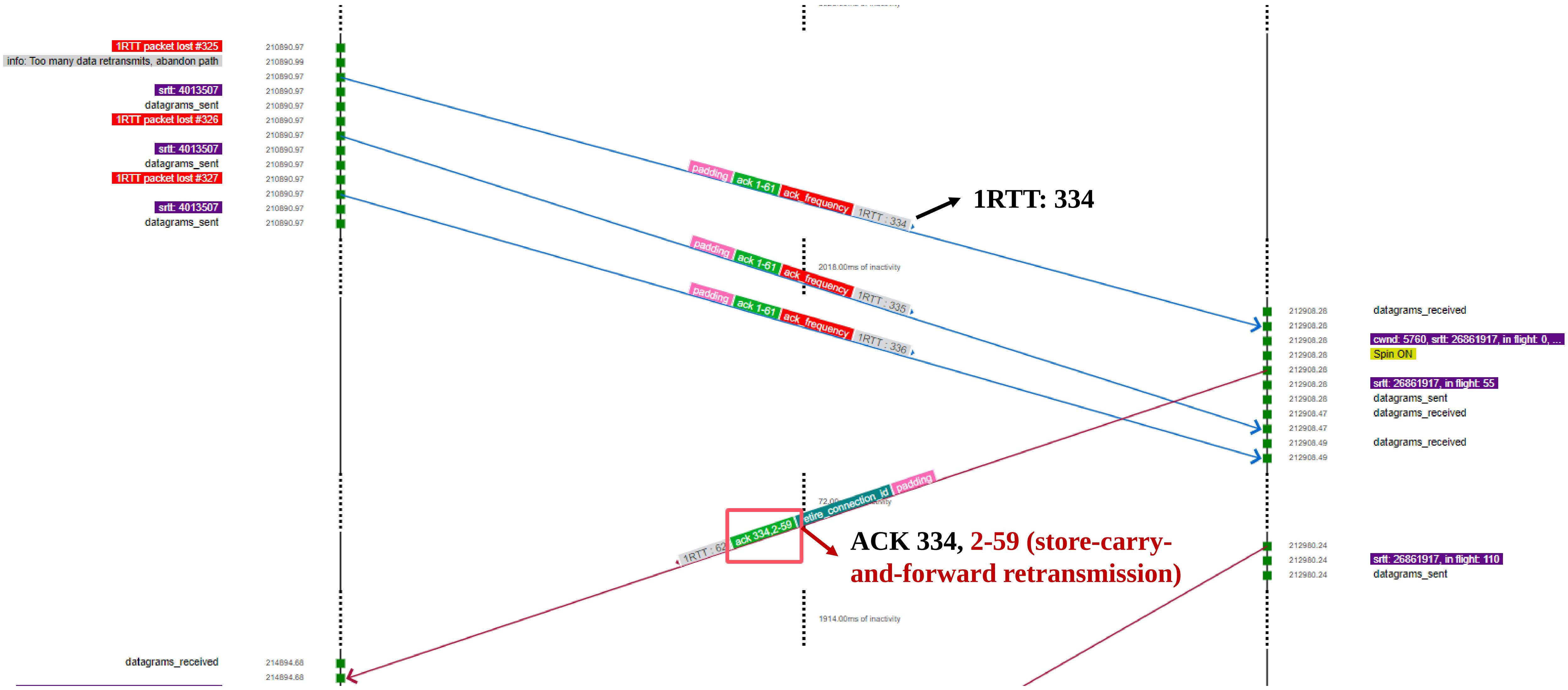
SRv6 based IP and DTN have consistent functions, making interconnection easier.

Test of QUIC and SRv6 based store-carry-and-forward interoperation



Purpose: In scenarios with link interruptions, files can be successfully delivered relying on the IP layer's store-carry-and-forward paradigm, without depending on the QUIC timeout retransmission mechanism.

Test of QUIC and SRv6 based store-carry-and-forward interoperation



Test of QUIC and SRv6 based store-carry-and-forward interoperation

25s: Link down

210s: Link up

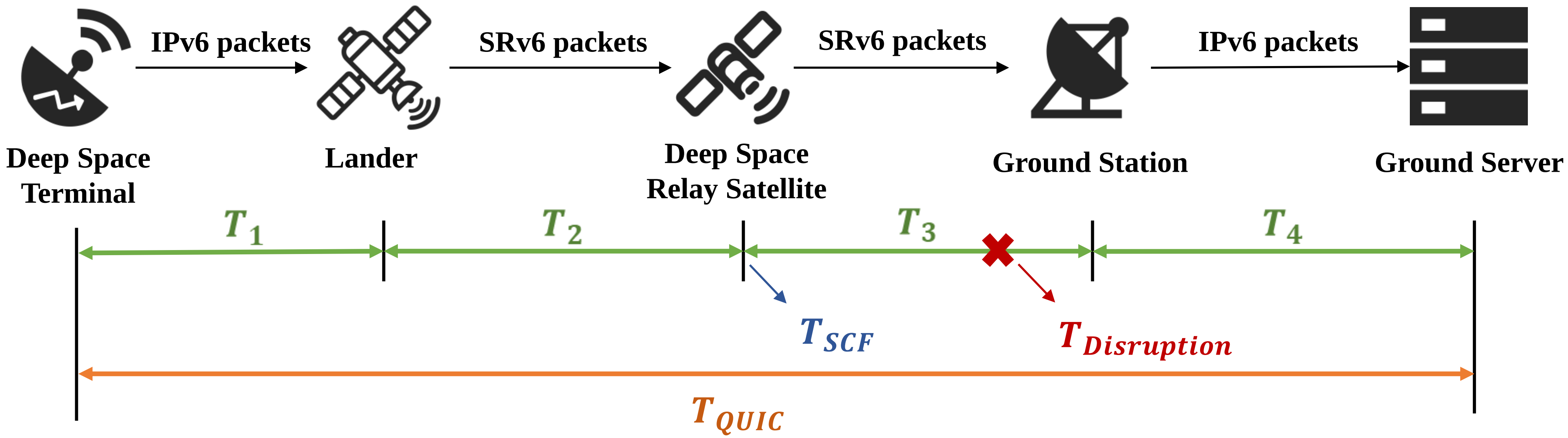
212s: First ACK reply

```
125 10.464882913  fc04::2                     fc00:1::1     QUIC   117 Protected P
126 17.884423358  fc04::2                     fc00:1::1     QUIC  1502 Protected P
127 25.303725575  fc04::2                     fc00:1::1     QUIC  1462 Protected P
128 91.817233562  fe80::1e22:2f4d:5bb7:9daa   ff02::fb      MDNS   107 Standard qu
129 91.817254431  fc02::2                     ff02::fb      MDNS   107 Standard qu
130 91.817258080  172.16.158.135              224.0.0.251   MDNS    87 Standard qu
131 91.817416534  172.16.158.134              224.0.0.251   MDNS    87 Standard qu
132 208.906118471 fe80::e2f1:d796:b044:4532   ff02::fb      MDNS   107 Standard qu
133 209.010413447 172.16.158.132              224.0.0.251   MDNS    87 Standard qu
134 209.035770170 172.16.158.131              224.0.0.251   MDNS    87 Standard qu
135 209.054124580 172.16.158.130              224.0.0.251   MDNS    87 Standard qu
136 210.140769688 fc00:1::1                   fc04::2       QUIC   213 Protected P
137 210.140966355 fc00:1::1                   fc04::2       QUIC   213 Protected P
138 210.141139226 fc00:1::1                   fc04::2       QUIC   213 Protected P
139 212.142471441 fc04::2                     fc00:1::1     QUIC   117 Protected P
140 212.214251906 fc04::2                     fc00:1::1     QUIC   117 Protected P
141 214.144506417 fc00:1::1                   fc04::2       QUIC  1390 Protected P
142 214.144733640 fc00:1::1                   fc04::2       QUIC   213 Protected P
143 214.144943374 fc00:1::1                   fc04::2       QUIC   213 Protected P

> Frame 136: 213 bytes on wire (1704 bits), 213 bytes captured (1704 bits) on interface ens38, id 0
> Ethernet II, Src: VMware_12:22:b5 (00:0c:29:12:22:b5), Dst: VMware_ec:d5:f4 (00:0c:29:ec:d5:f4)
> Internet Protocol Version 6, Src: fc00:1::1, Dst: 2001:db8:300:1:300:0:1:14f
> Internet Protocol Version 6, Src: fc00:1::1, Dst: fc04::2
v User Datagram Protocol, Src Port: 4433, Dst Port: 4434
    Source Port: 4433
    Destination Port: 4434
    Length: 63
    Checksum: 0x4986 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 0]
  v [Timestamps]
      [Time since first frame: 210.140769688 seconds]
      [Time since previous frame: 184.837044113 seconds]
    UDP payload (55 bytes)
v QUIC IETF
  v QUIC Connection information
      [Connection Number: 0]
    [Packet Length: 55]
  > QUIC Short Header DCID=9d6771d42d9d18c6
    Remaining Payload: be3a1278ea268829c646b595fcc9ff728161887991904ca911b13fce91f03b4440aea29d3d5a9e7eebaddf889b62
```

store-carry-and-forward retransmission

Disruption Time Modelling for layers Interoperations



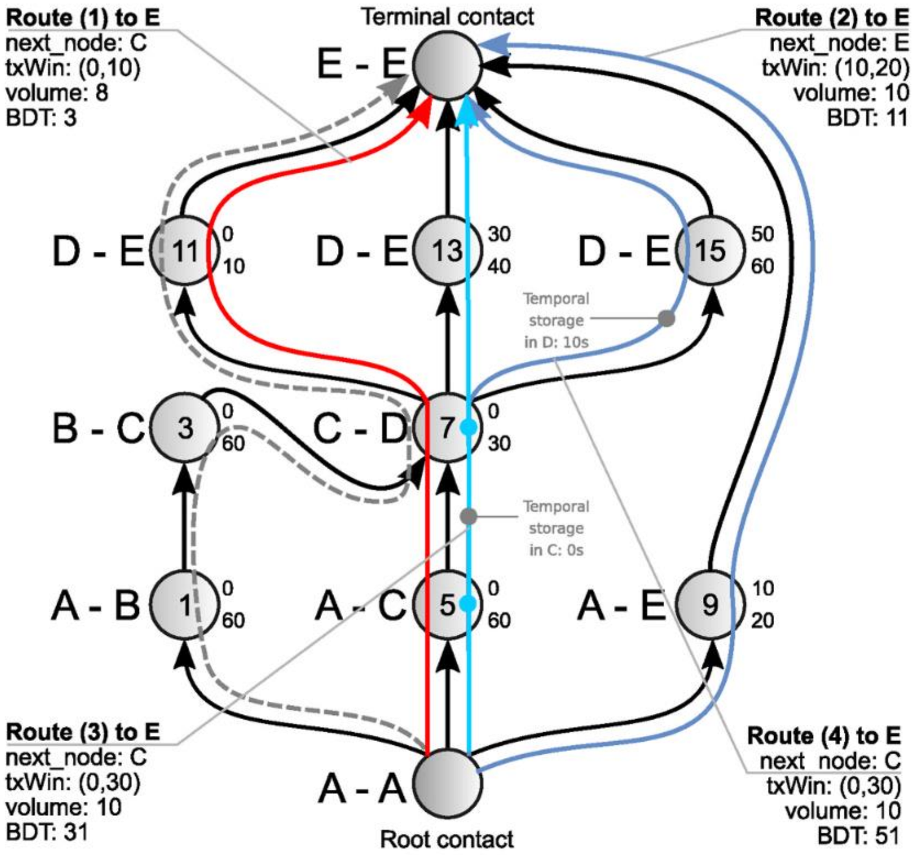
Problem: QUIC retransmissions over a store-carry-and-forward mechanism may result in invalid **duplicate packets**. How can we reconcile the reliability relationship between these two?

$$T_{QUIC} \geq T_1 + T_2 + T_3 + T_4 + T_{Disruption} + T_{SCF}$$

· **Dynamically Adjust QUIC Retransmission Timer**

CGR based routing to create SRv6 segment list

In deep space networks, the communication windows (contacts) between nodes are **pre-planned**, allowing the optimal transmission paths between any nodes to be theoretically inferred from these contact schedules.



Contact graph

· **Node:** a contact in the contact plan that allows data transmission between two nodes;

· **Link:** the necessary storage time for data at the sending node while waiting for the next contact transmission.

Every route includes:

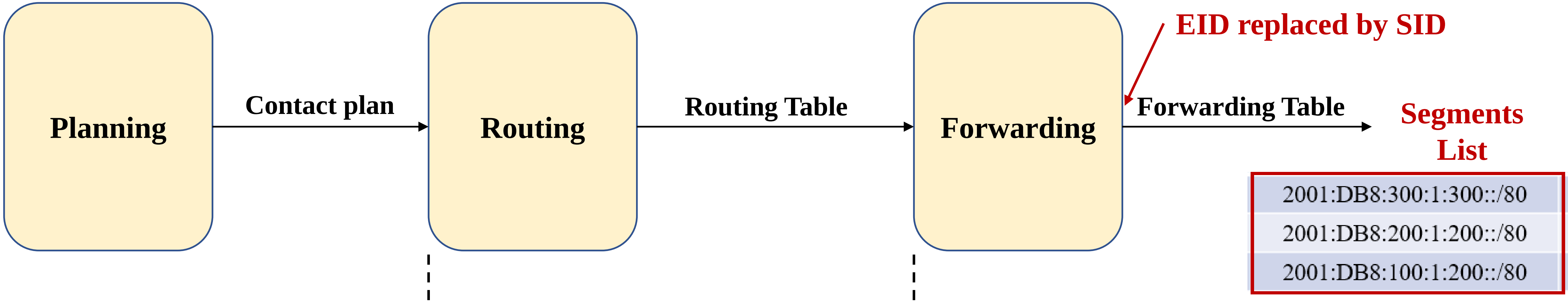
· **Temporal storage:** data storage time at the node;

· **txWin (transmission window):** [the start time of the first hop, the earliest end time among the contacts traversed]

· **volume = (end-start) * rate**;

· **BDT (best delivery time):** the earliest time at which the destination node can be reached.

CGR based routing to create SRv6 segment list



A centralized entity computes contact plans based on the estimation of future episodes of communications.

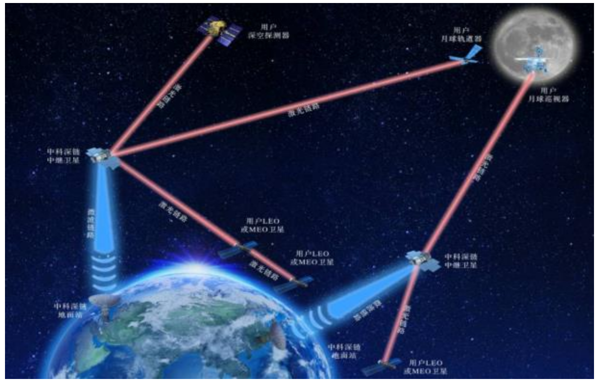
Contact Graph Dijkstra Search: compute a path with the best delivery time in a contact graph;

Contact Graph Yen's algorithm: compute the best K routes in the contact plan.

Utilize a four-step filtering process to generate a candidate routing table, and select based on the following criteria:

1. Shortest BDT;
2. Fewest hops;
3. Latest route end time;
4. Smallest EID.

New scenarios for next-generation deep space networks



Earth-Moon Communication

Future Lunar Communication

With the continuous development of manned lunar exploration programs and deep space exploration, the scale and applications of deep space networks are increasingly resembling **the IP-centric internet**:

· The manned lunar exploration network involves not only remote sensing data transmission addressed by DTN;

· Also includes **astronaut-to-ground** communication, **human-machine** communication with lunar sensor networks, and **machine-to-machine** communication within lunar sensor networks.

Why IPv6+ network architecture for deep space

SRv6 technology enables IPv6 networks to be programmable, allowing them to meet more complex and flexible communication requirements. The **design philosophy** for next-generation deep space network architecture includes:

· **Support for Large Networks:** The model is closer to an IP-centric terrestrial internet;

· **Scalability:** Supports dynamic routing, allowing terminals and network nodes to join and leave freely;

· **Open Interconnectivity:** Forming a seamless interstellar network that connects lunar, Martian, and Venusian networks with terrestrial networks, sharing applications of the terrestrial internet;

· **Resilience:** Addresses reliability issues, enabling rapid rerouting capabilities;

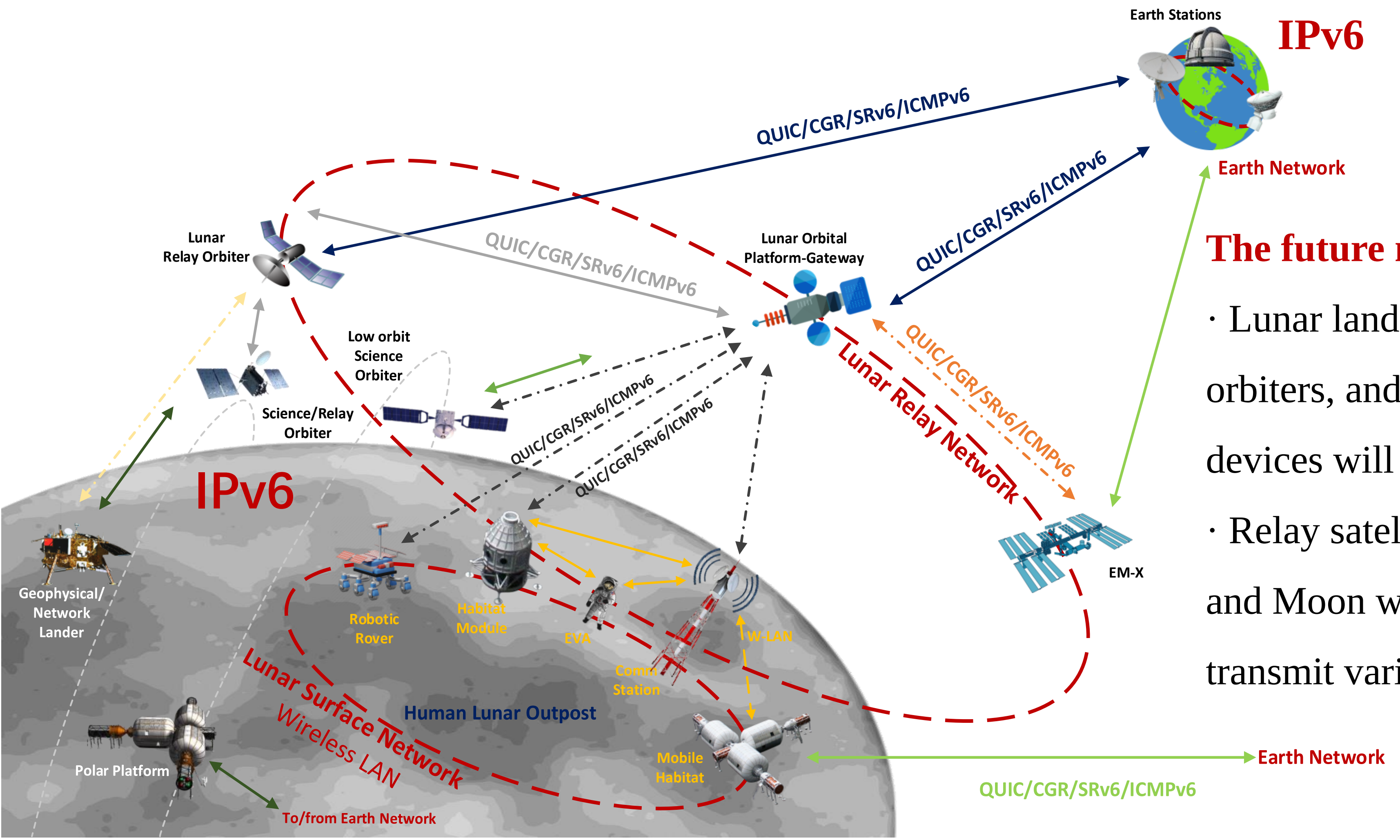
· **Programmability:** Equipped with SDN capabilities, supporting differentiated service provisioning and QoS guarantees, allowing for flexible service policies and traffic engineering;

· **Simplified Protocol Stack:** Enhances network efficiency and simplifies network operations;

· **Automation:** Improves network operation efficiency.

IPv6+ Network Architecture for Deep Space = IPv6 + SRv6 +ICMPv6 + QUIC + CGR

IPv6+ Networking : the next-generation deep space networks



The future manned lunar exploration network :

· Lunar landers, rovers, intelligent robots, astronauts, orbiters, and various remote sensing and telemetry devices will be interconnected using **IPv6**;

· Relay satellites and ground stations between Earth and Moon will utilize an enhanced **QUIC+SRv6** to transmit various data.