SRv6 Based Store-Carry-and-Forward Networking 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

2024.07.24

Deep Space Network Open Problems

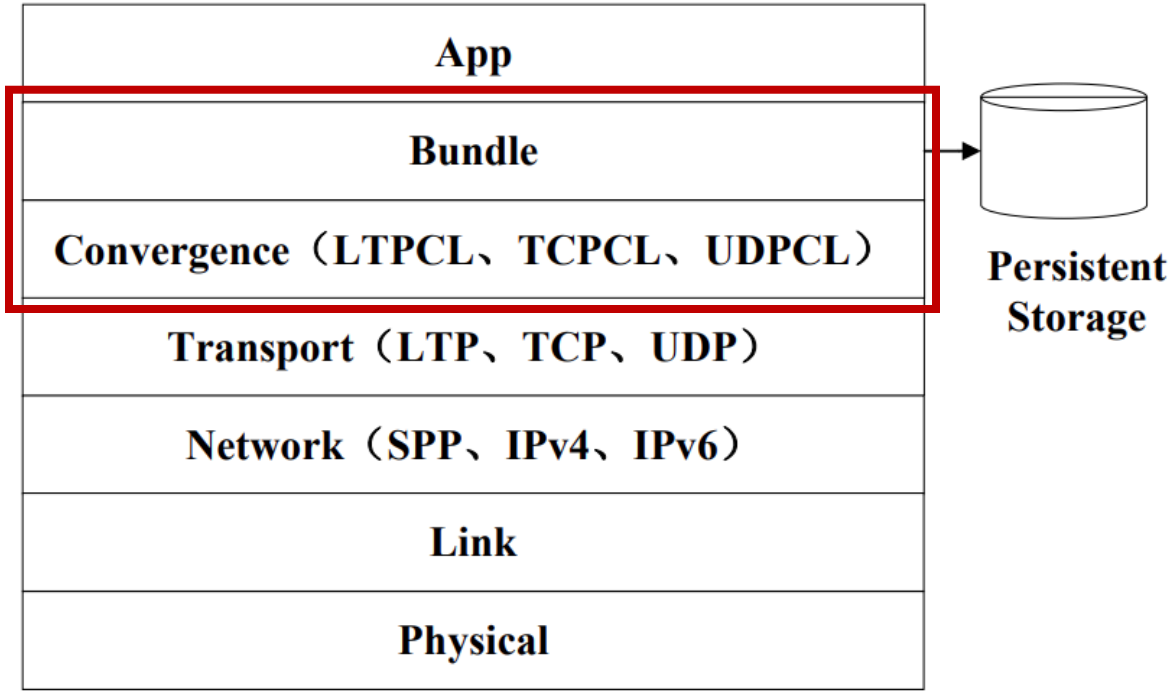
· tolerate ultra-long propagation delay;

· tolerate link disruption;

· tolerate asymmetric round-trip data rate;

· tolerate high bit error rate.

DTN Solution for Deep Space Network

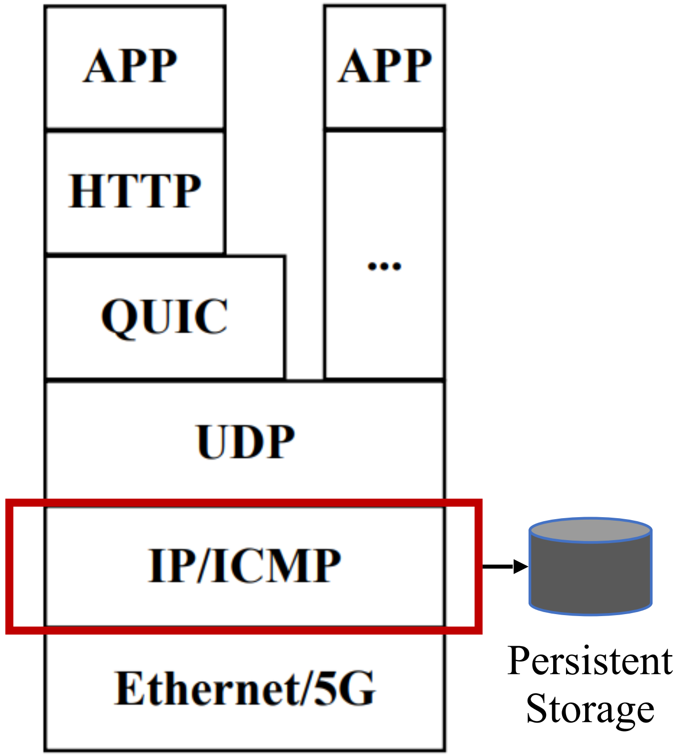


DTN Architecture

BP for Store-Carry-and-Forward

- **Hop-by-hop transmission** via BP
- When a **link disruption** occurs during the Bundle transmission process, the Bundle will be **saved in the relay node** and **wait for the next transmission opportunity** to arrive.
- When the link is **transmittable** again, the Bundle will be **sent from the relay node to the next node**.

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.

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

Basic SR forwarding paradigm

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

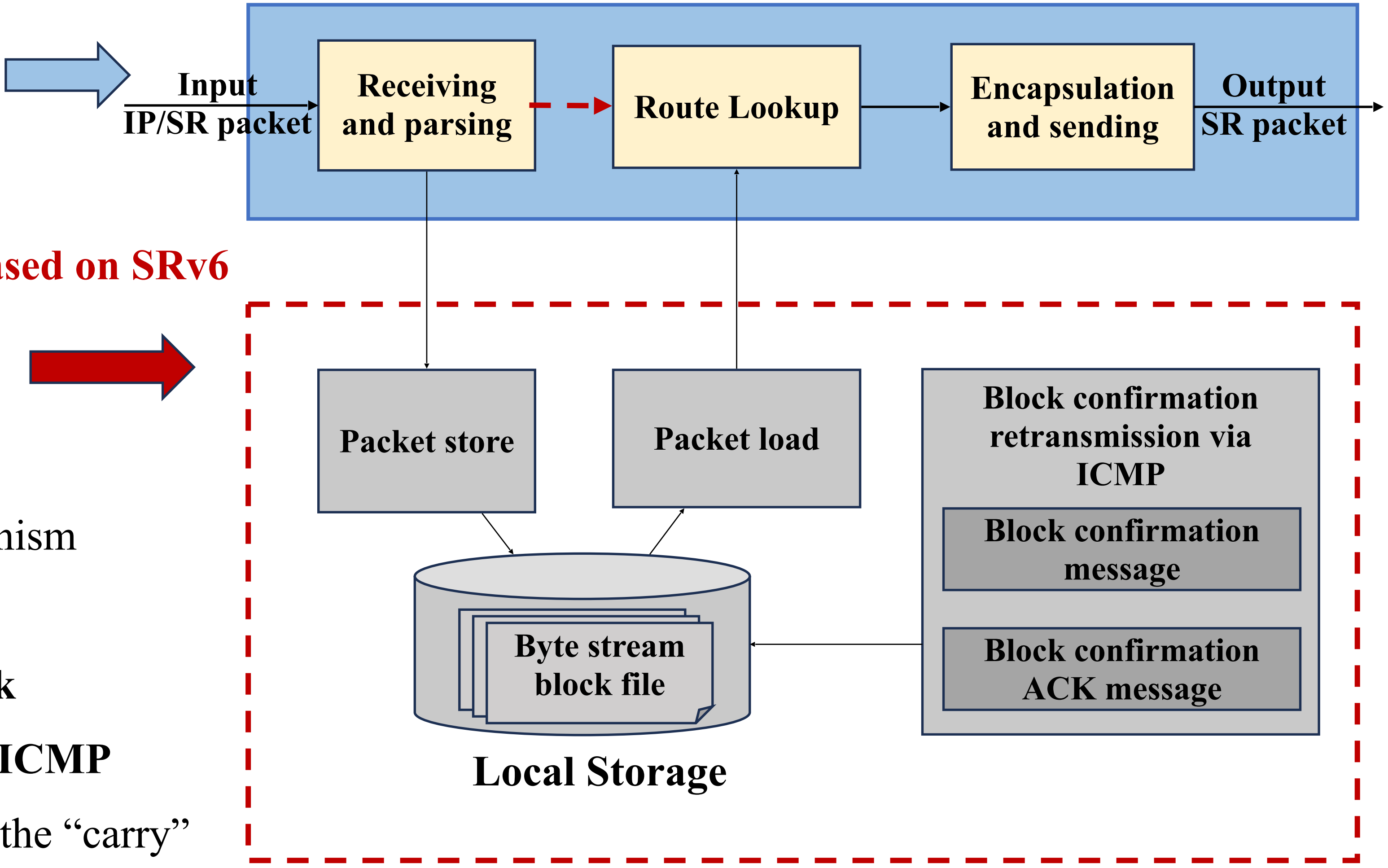
(1) Add **packet store** function

(2) Add **packet load** function

· **Packet reading drive** mechanism

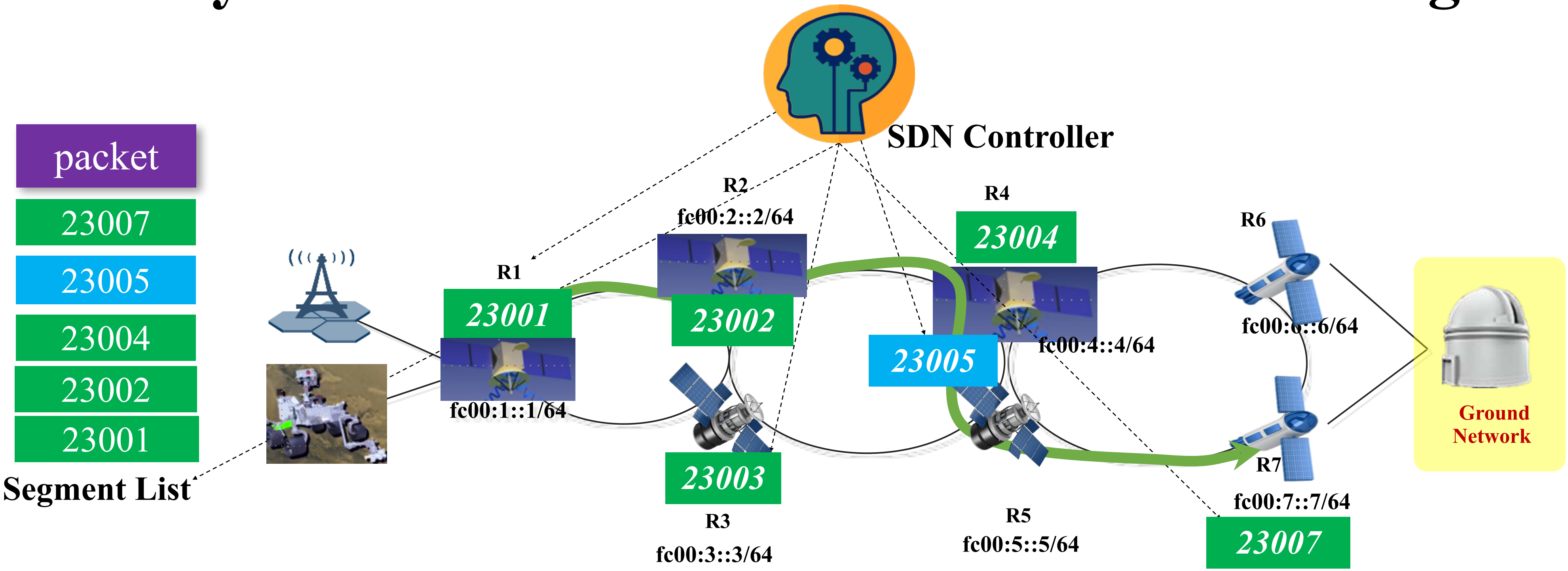
· **Packet recovery** mechanism

(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

Store-Carry-and-Forward Based on SRv6 Networking Flow



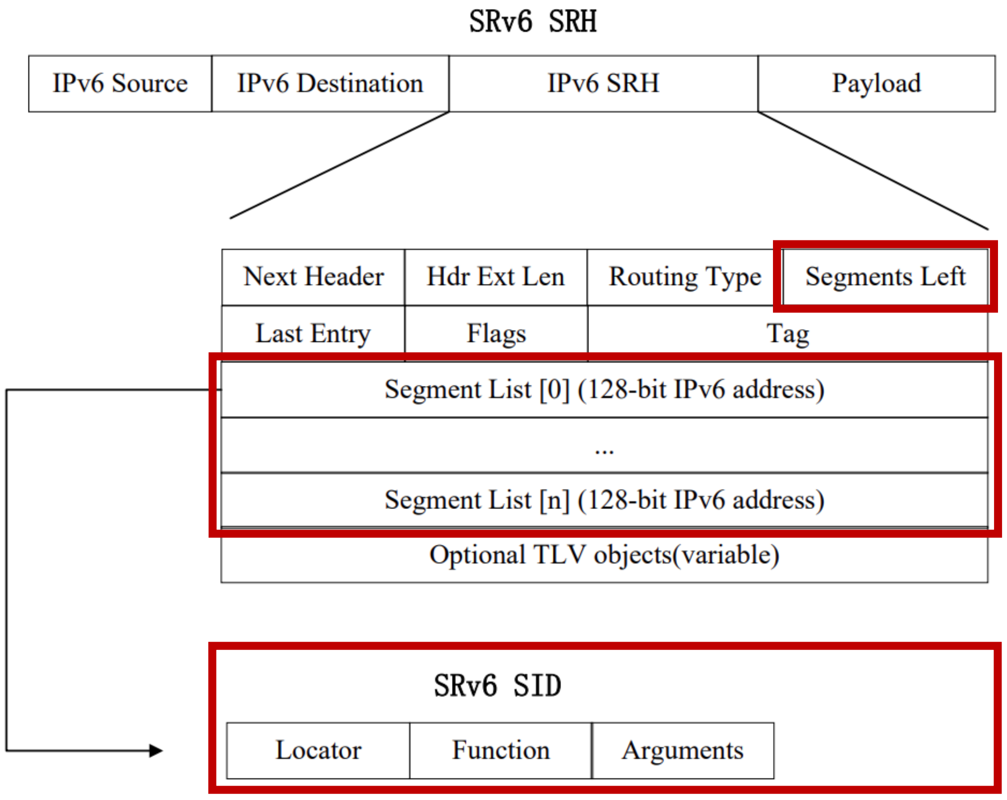
· Segment Routing (**SR**) leverages the source routing paradigm. SRv6 divides the network path into segments and assigns segment identifiers(**SIDs**);

· Control plane, i.e, SDN controller, calculates the network path to the destination and converts the path into an ordered **Segment list;**

· At the source node, forwarding plane encapsulates the Segment List into the SRH;

· The intermediate nodes of the path only need to forward packet hop by hop by the Segment List in the SRH.

SID and Segment List encapsulation

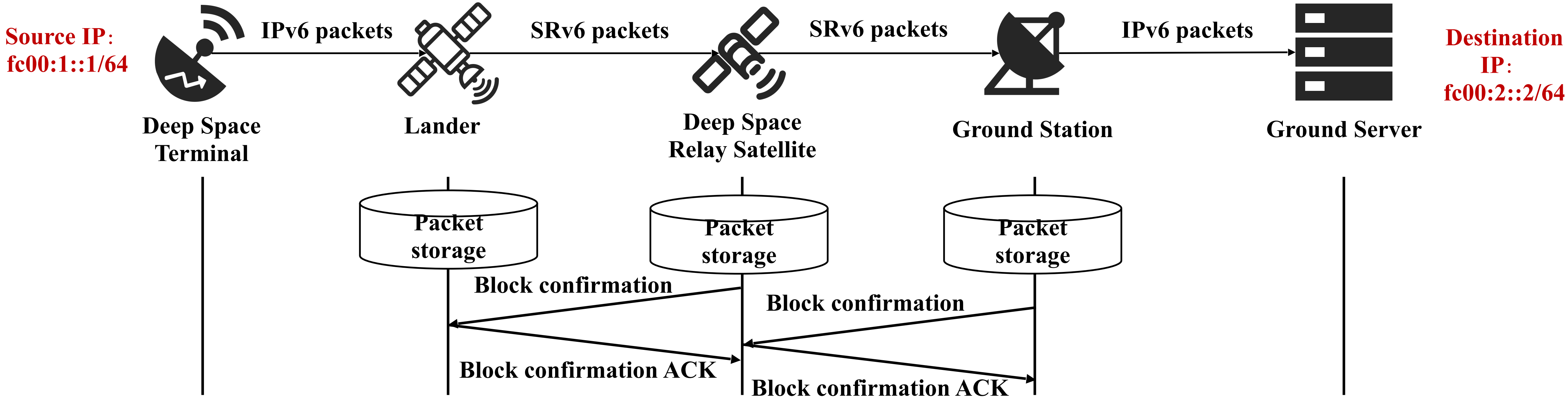


An SRv6 SID is an IPv6 Address

SID	Description
End SID	update the IPv6 DA and then search the IPv6 Forwarding Information Base (FIB) for packet forwarding
End.X SID	update the IPv6 DA and then forward packets through the outbound interface bound to the SID
End.DT6 SID	decapsulate packets and then search the routing table of the involved IPv6 VPN instance for packet forwarding
End.DX6 SID	decapsulate packets and then forward the resulting IPv6 packets through the Layer 3 interface bound to the SID
End.XS SID	**Our new SID**, store packets locally, update the IPv6 DA and then forward packets through the outbound interface bound to the SID

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

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.

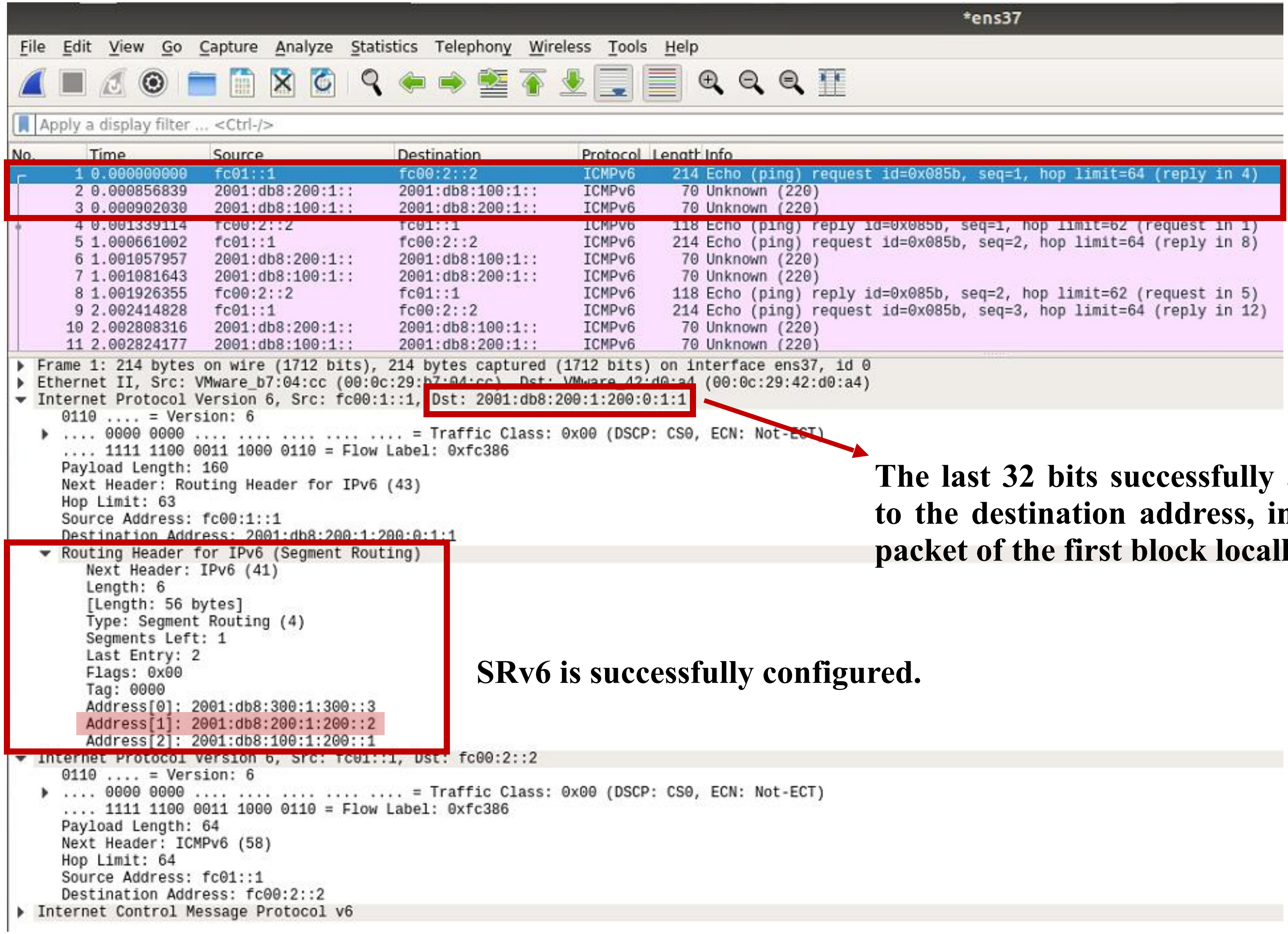
```
[ 1283.135172] Print skb info module loaded
[ 1283.135177] Block Number: 1, Packets: 4, Address: 2001:db8:300:1:300::3
[ 1283.135179] Packet Number: 1
[ 1283.135180] Packet Number: 2
[ 1283.135180] Packet Number: 3
[ 1283.135181] Packet Number: 4
```

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.

Prototype Validation



Data packets, block confirmation messages, and block confirmation ACK messages are all successfully transmitted.

The last 32 bits successfully add the storage location information to the destination address, indicating that it is stored in the first packet of the first block locally.

SRv6 is successfully configured.

DTN 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.

Next Work

1. **Comparative test between IP and DTN** from Application view

- Take CFDP/BP/LTP vs FTP/QUIC/UDP/IP as test cases
- Test the time it takes to transfer multiple files in parallel

2. To propose **a more detailed SRv6 Deep Space IP Architecture**

- control plane architecture design, including routing, mobility management, DNS and security
- performance improvement in data plane, including three packet buffer stages model, porting Linux kernel codes to FPGA