

IPv6 Protocol Architecture

New Functional Improvement

- Address Space
 - Increase from 32-bit to 128-bit address space
- Management
 - Stateless autoconfiguration (SLAAC) means no more need to configure IP addresses for end systems, even via DHCP
- Performance
 - Simplified header means efficient packet processing
 - No header checksum re-calculation at every hop (when TTL is decremented) => **left up to the lower and upper layers!**
- No hop-by-hop fragmentation - PMTUD

IPv4/IPv6 Header Comparison

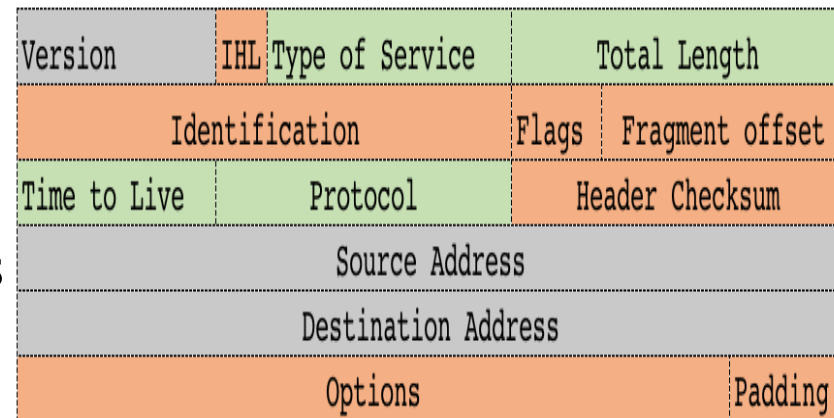
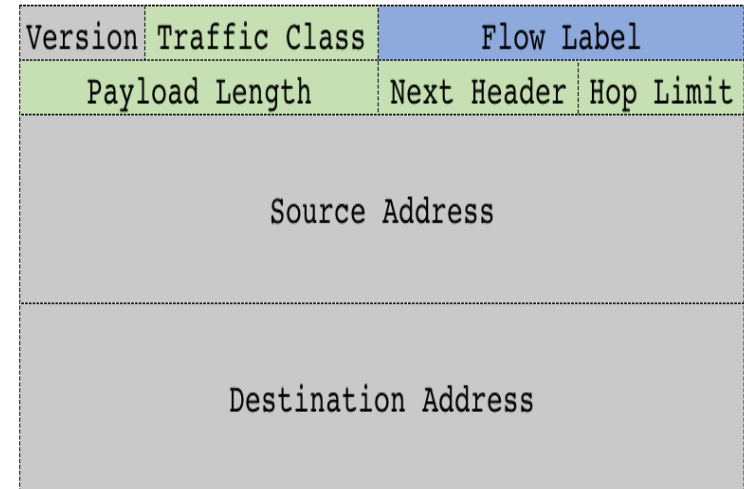
Version	IHL	Type of Service	Total Length	
Identification			Flags	Fragment offset
Time to Live	Protocol		Header Checksum	
Source Address				
Destination Address				
Options				Padding

- Not kept in IPv6
- Renamed in IPv6
- Same name and function
- New in IPv6

Version	Traffic Class	Flow Label	
Payload Length		Next Header	Hop Limit
Source Address			
Destination Address			

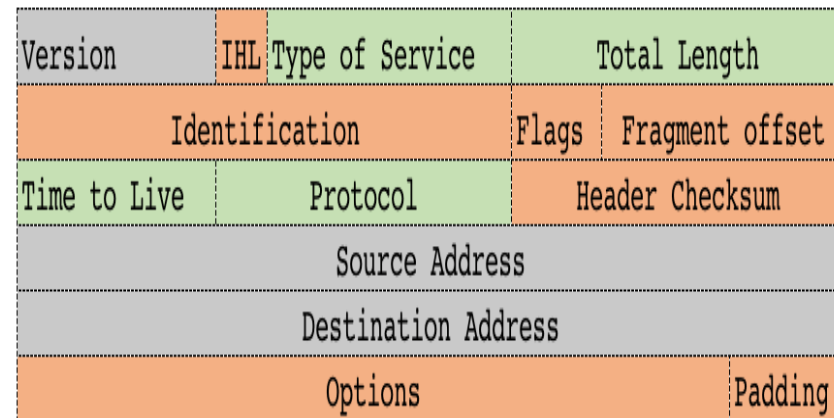
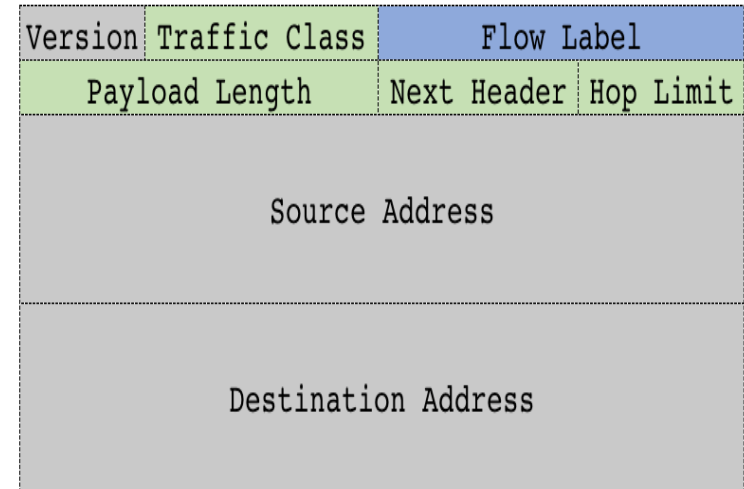
IPv6 Protocol Header Format

- Version (4-bit):
 - 4-bit IP version number (6)
- Traffic class (8-bit):
 - Similar to DiffServ in IPv4; define different classes or priorities.
- Flow label (20-bit):
 - allows IPv6 packets to be identified based on flows (multilayer switching techniques and faster packet-switching performance)



IPv6 Protocol Header Format

- Payload length (16-bit):
 - Defines the length of the IPv6 payload (including extension headers); Total Length in IPv4 includes the header.
- Next header (8-bit):
 - Identifies the type of information following IPv6 header. Could be upper layer (TCP/UDP), or an extension header (similar to Protocol field in IPv4).
- Hop limit (8-bit):
 - Similar to TTL in IPv4

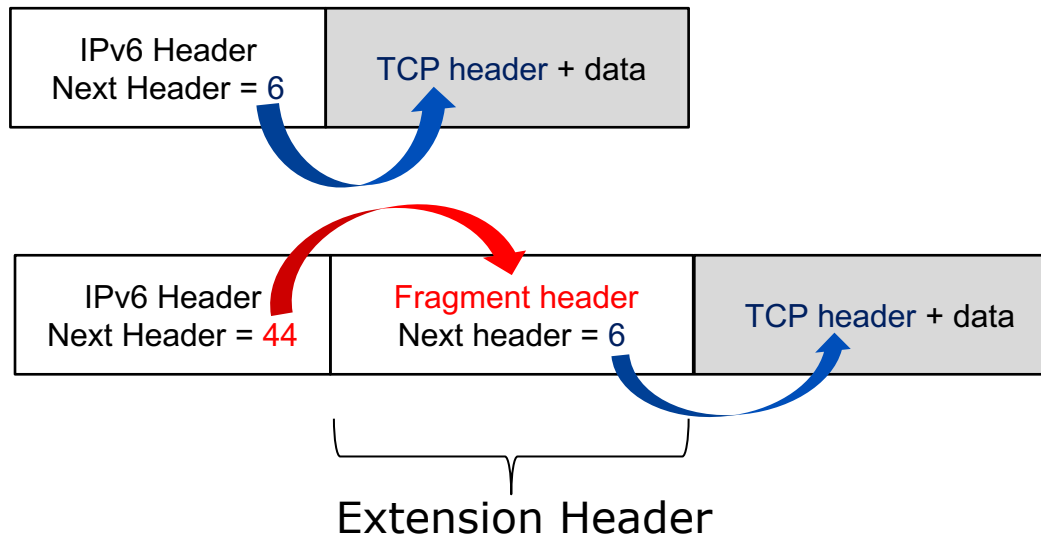


IPv6 & IPv4 Packet Example

- Example IPv6 packet on this link:
- <https://www.cloudshark.org/captures/84fd54ad03e0>
- Example IPv4 packet on this link:
- <https://www.cloudshark.org/captures/09f49cda5b80>

IPv6 Extension Header

- IPv6 allows an optional *Extension Header* in between the IPv6 header and upper layer header
 - Allows adding new features to IPv6 protocol without major re-engineering



Next Header values:

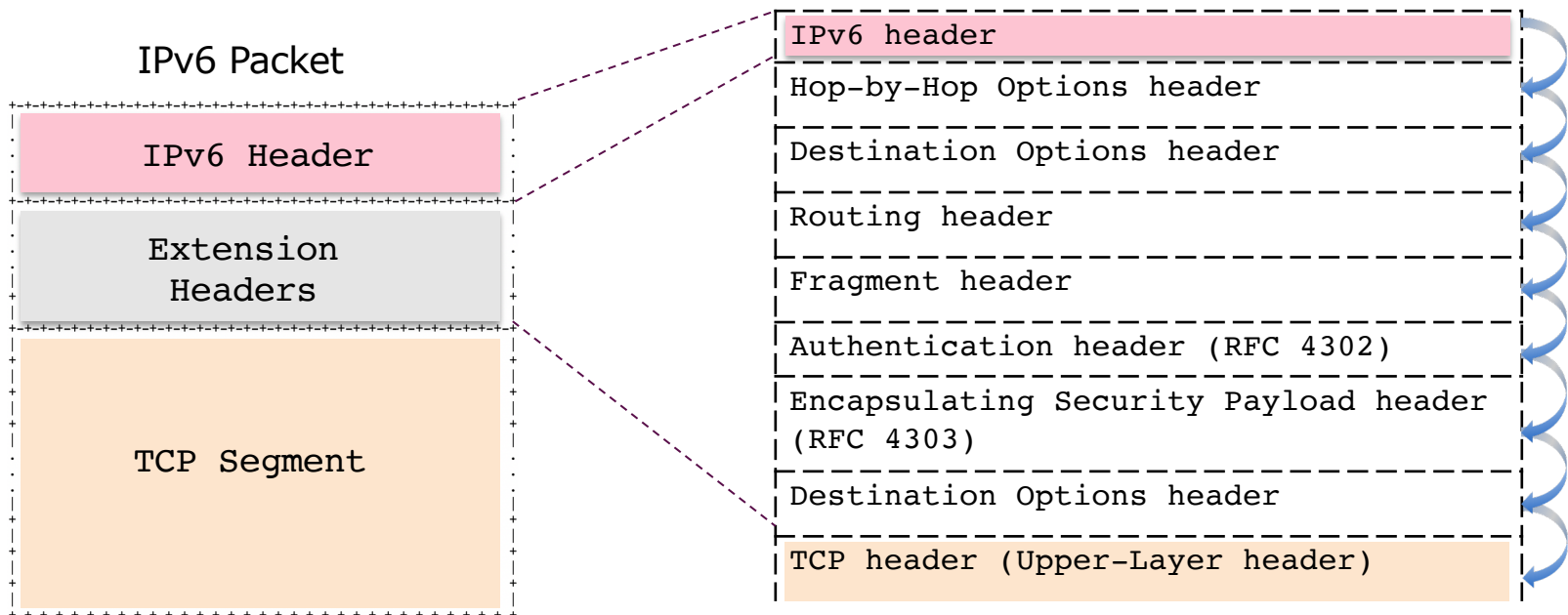
0 Hop-by-hop option
6 TCP
17 UDP
43 Source routing (RFC5095)
44 Fragmentation
50 Encrypted security payload
51 Authentication
58 ICMPv6
59 Null (No next header)
60 Destination option

IPv6 Extension Header (contd)

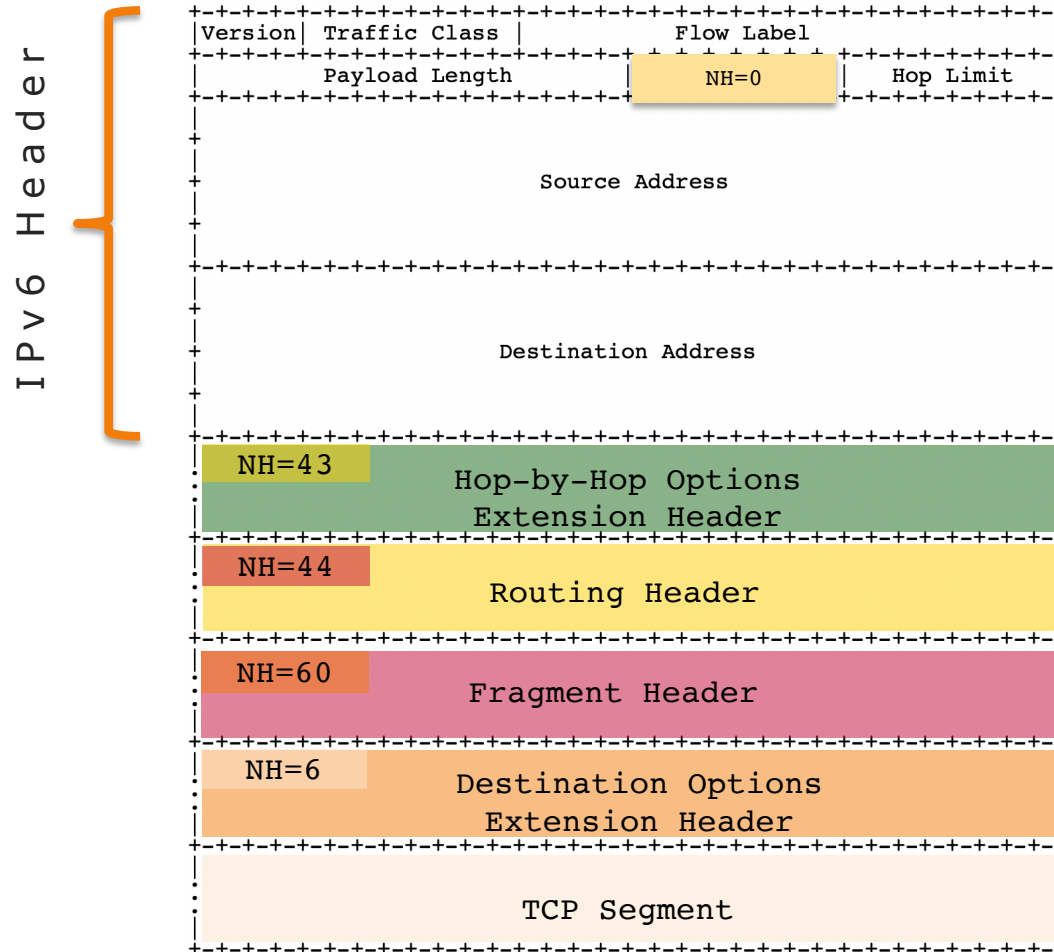
- An IPv6 packet may carry none or many extension headers
 - A next header value of 6 or 17 (TCP/UDP) indicates there is no extension header
 - the next header field points to TCP/UDP header, which is the payload
- Unless the next header value is 0 (*Hop-by-Hop option*), extension headers are processed only by the destination node, specified by the destination address.

Extension Header Order

- When more than one extension header is used in the same packet, it is **recommended** that those headers appear in the following order in RFC 8200:



Chaining Extension Headers



Extension Header	Type
Hop-by-Hop Options	0
Fragment	44
Destination Options	60
Routing	43
Authentication	51
Encapsulating Security Payload	50

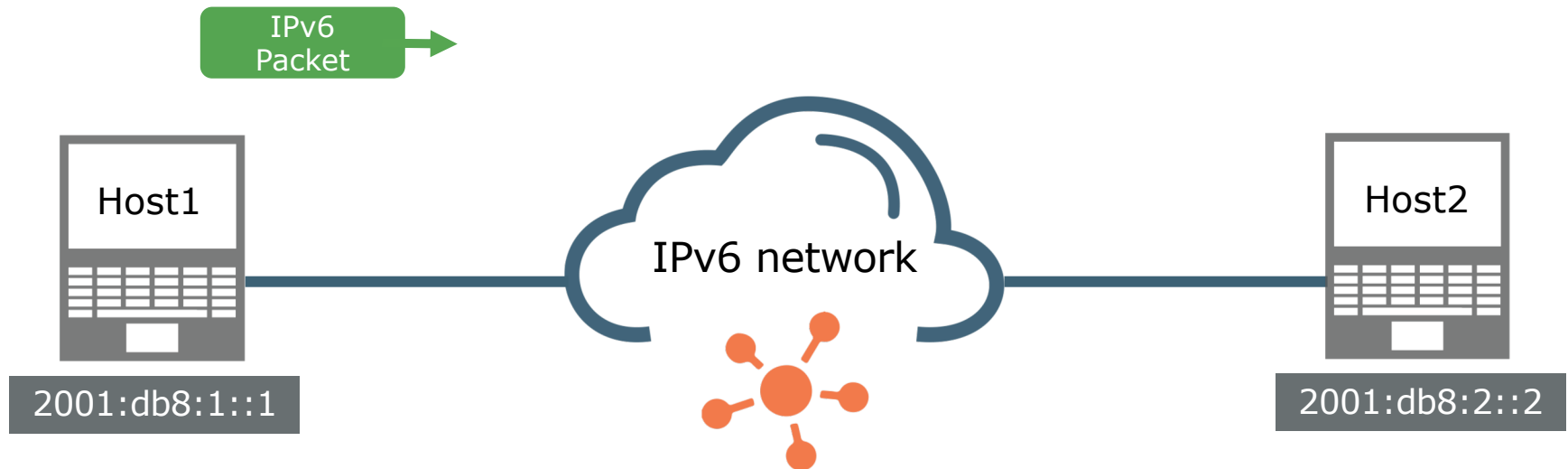
Extension Header Example

- Example IPv6 packet with an Extension Header on this link:
- <https://www.cloudshark.org/captures/7dd0b50eb768>

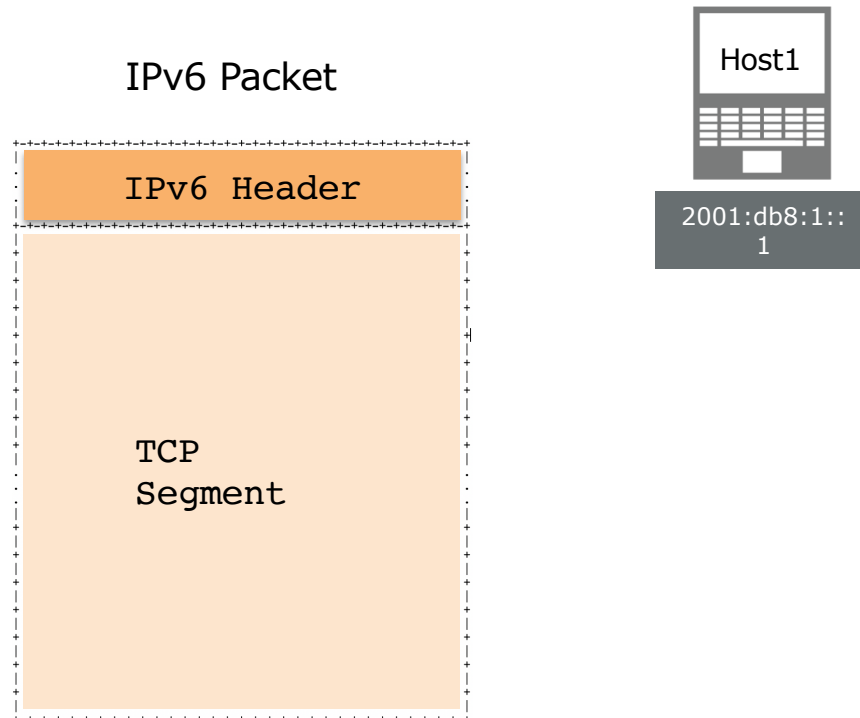
Fragmentation Handling In IPv6

- In IPv6, fragmentation is only performed by the host/source nodes, and not the routers along the path (unlike IPv4)
- Each source device tracks the MTU size for each session
- When a IPv6 host has large amount of data to be sent, it will be send in a series of IPv6 packets (fragmented)
 - IPv6 hosts use Path MTU Discovery (PMTUD) to determine the most optimum MTU size along the path

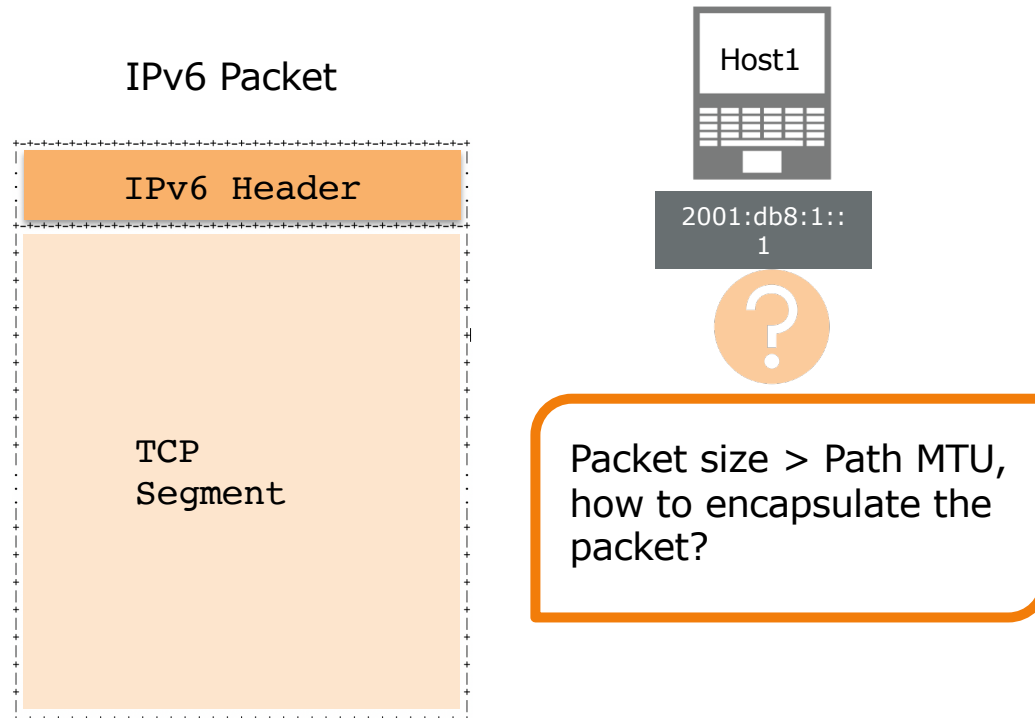
Example of Fragment Header



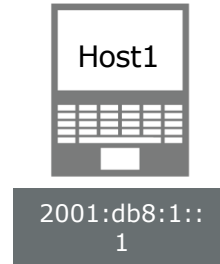
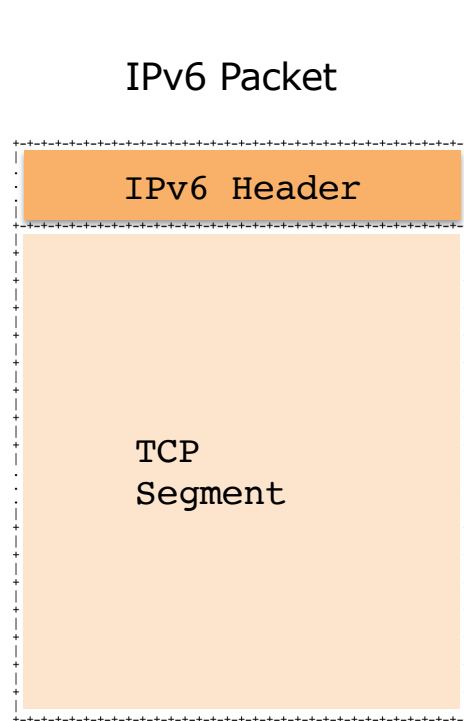
On the Source Node (Host1)



On the Source Node (Host1)

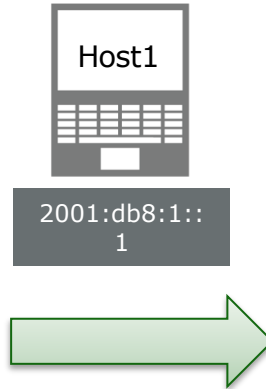
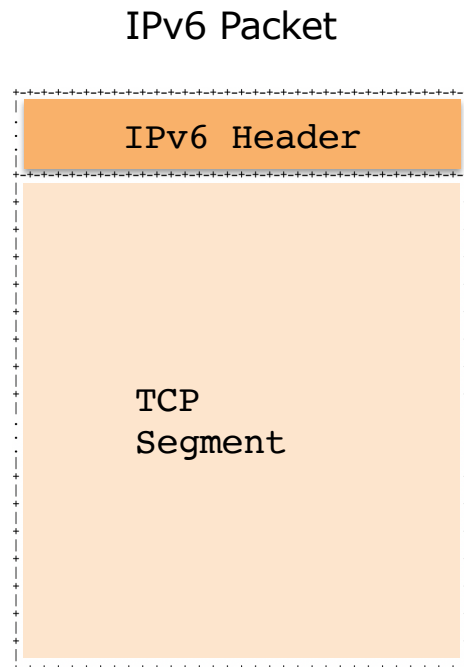


On the Source Node (Host1)



Divide the packet
into fragments.

On the Source Node (Host1)



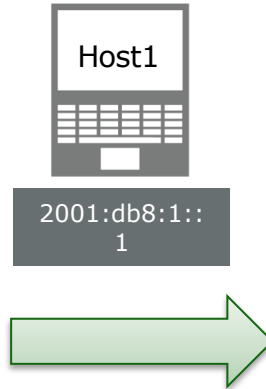
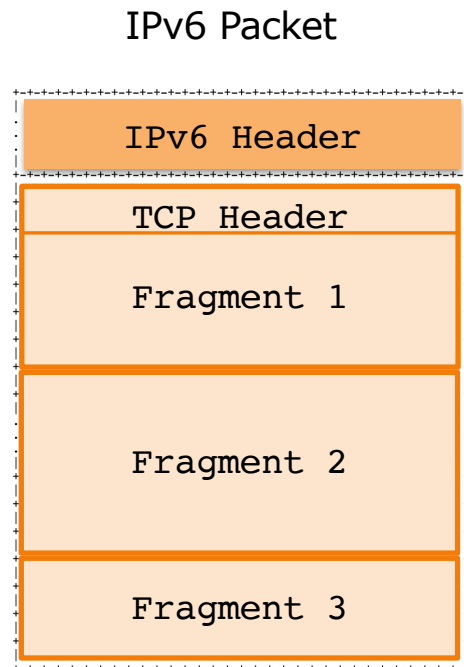
Divide the packet
into fragments.

IPv6 Packet 1

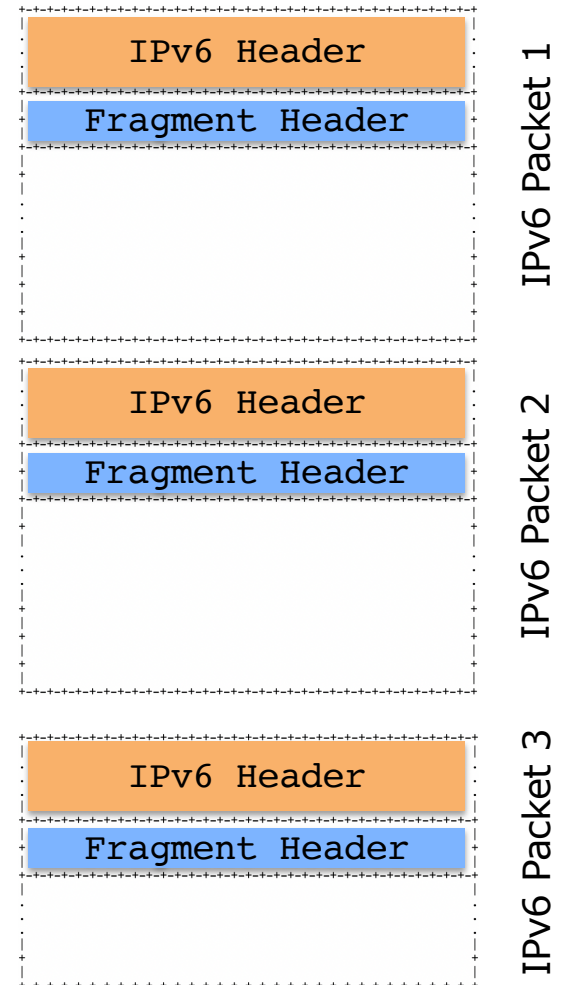
IPv6 Packet 2

IPv6 Packet 3

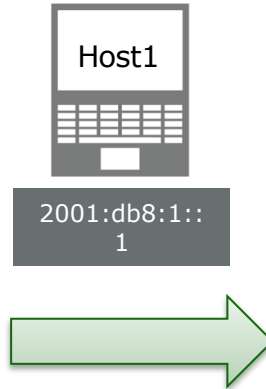
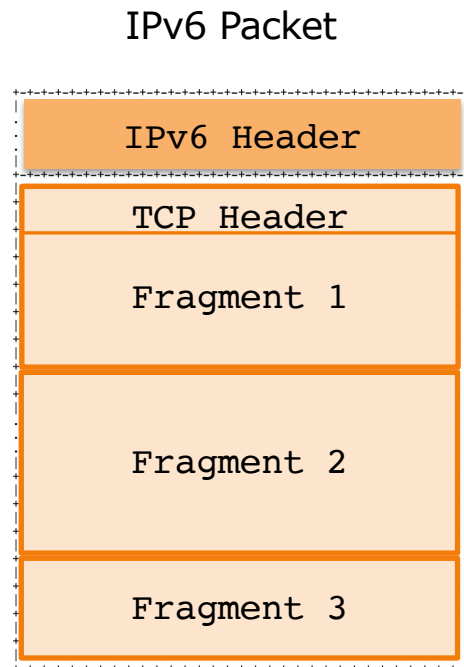
On the Source Node (Host1)



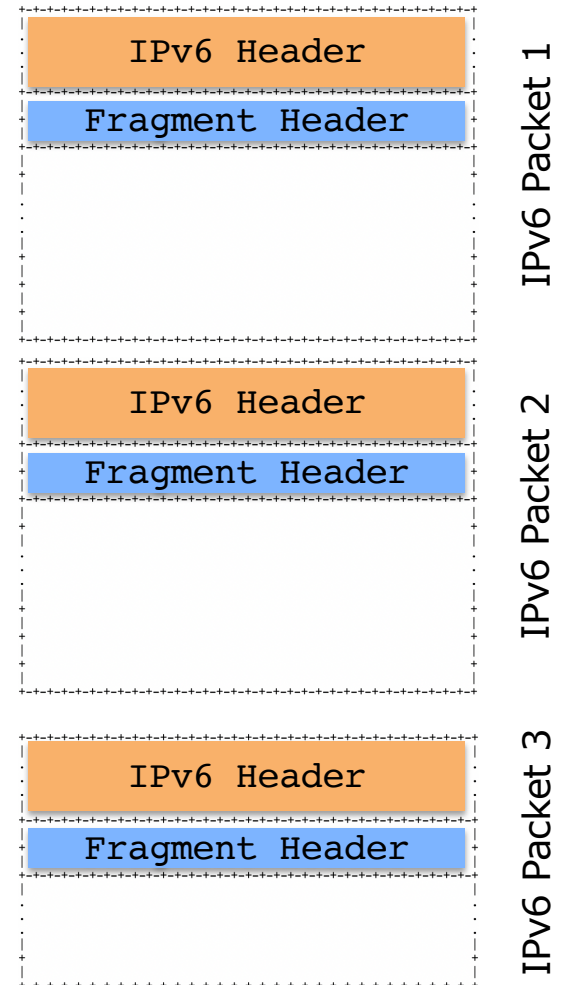
Divide the packet
into fragments.



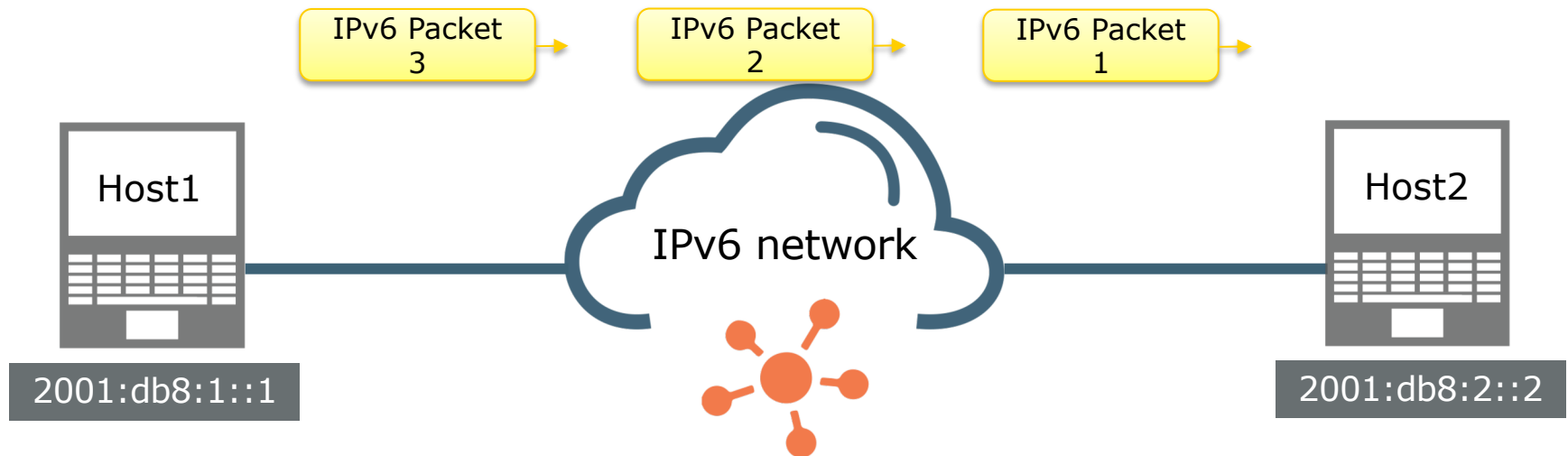
On the Source Node (Host1)



Divide the packet
into fragments.

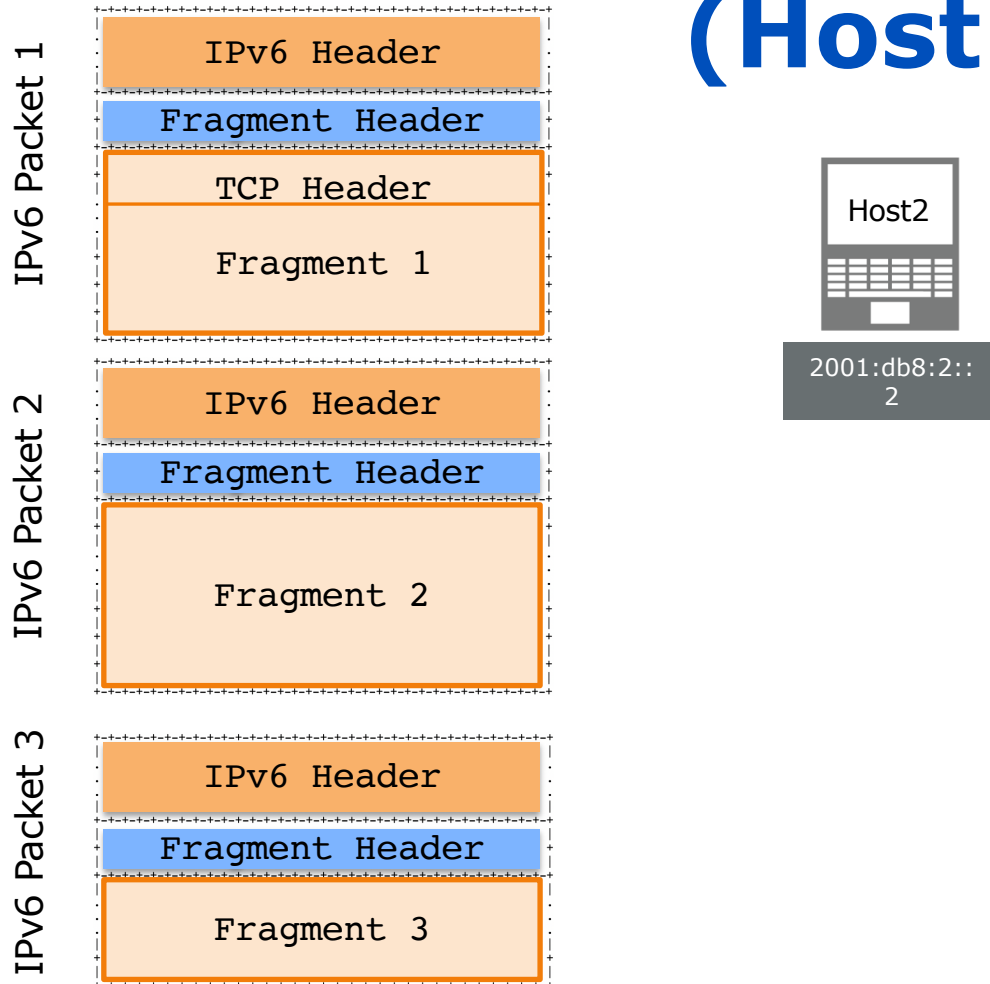


Example of Fragment Header

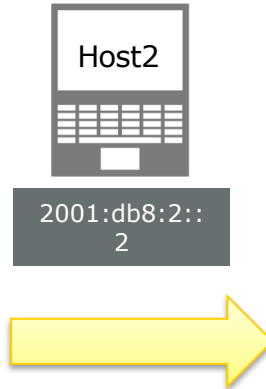
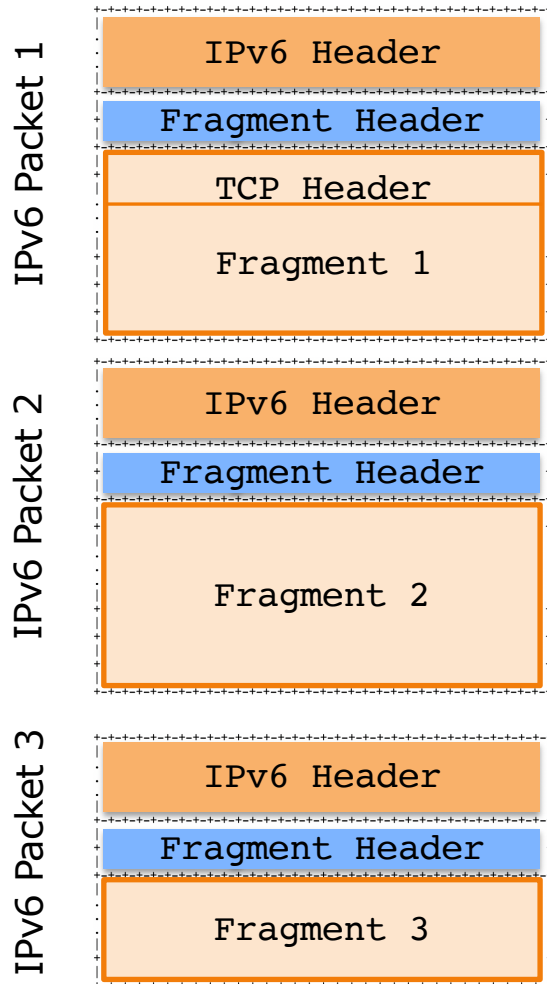


The 3 fragmented packets are transmitted on the path, reach the destination Host2, without any other fragmentation on the path.

On the Destination Node (Host2)

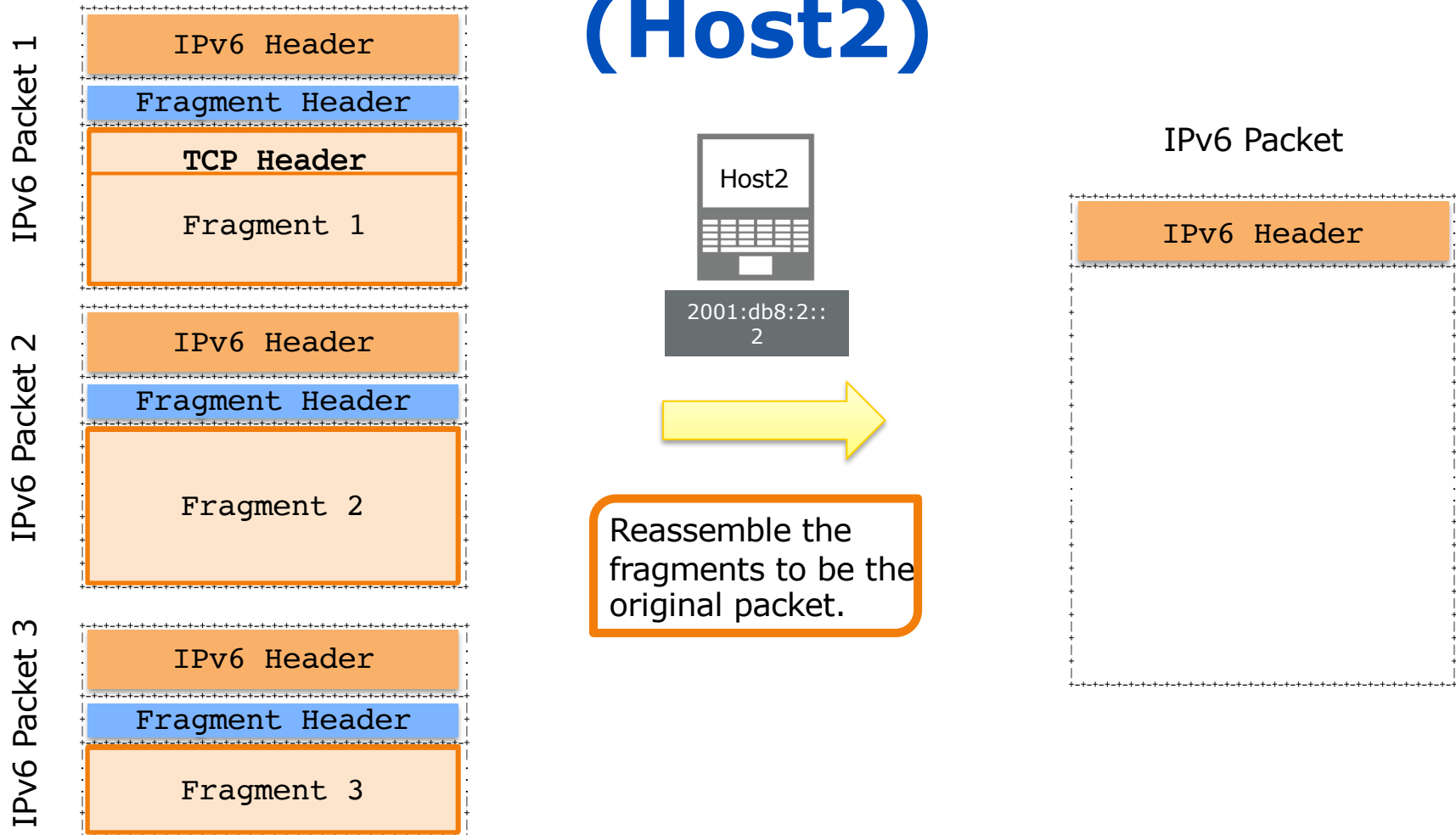


On the Destination Node (Host2)

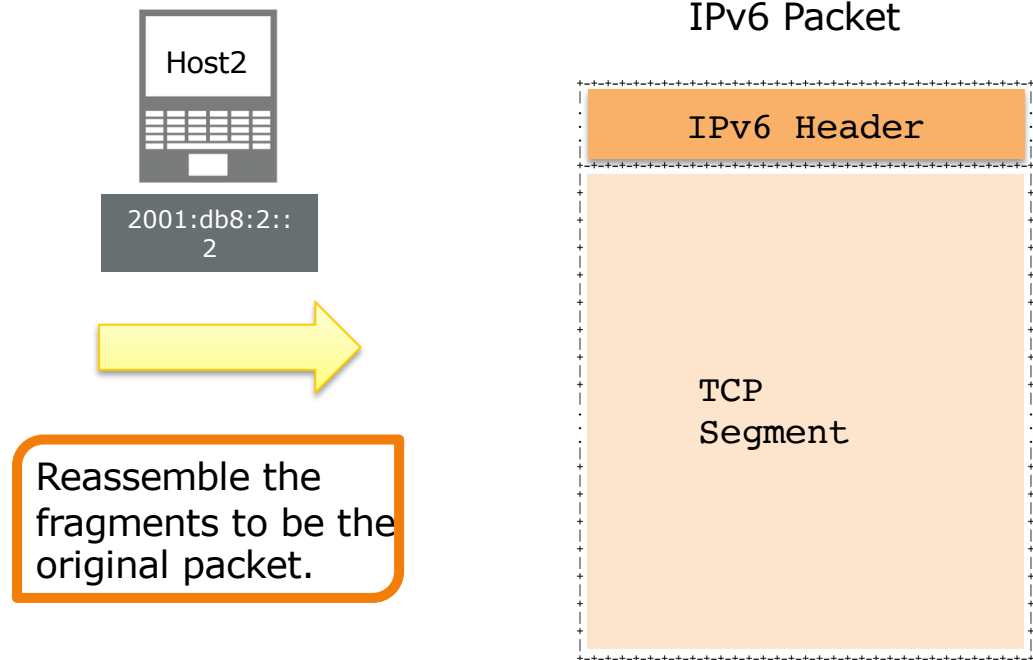


Reassemble the fragments to be the original packet.

On the Destination Node (Host2)



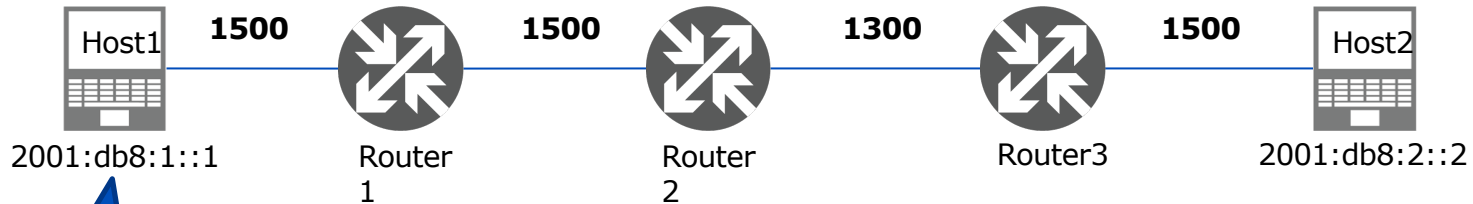
On the Destination Node (Host2)



Path MTU Discovery

- With PMTUD, the source IPv6 device assumes the initial PMTU is the MTU of the first hop in the path
 - upper layers (Transport/Application) send packets based on the first hop MTU
 - If the device receives an “*ICMPv6 packet too big (Type 2)*” message, it informs the upper layer to reduce its packet size, based on the actual MTU size (contained in the message) of the node that dropped the packet

Path MTU Discovery

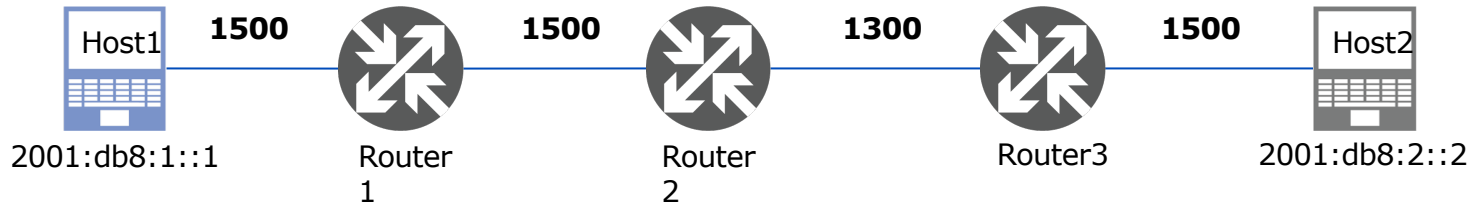


Link MTU values are marked on each link.

I have a packet with size 2000 bytes to send to Host2. It is larger than MTU, I have to fragment it.

Host1 :
MTU
cache=**1500**

Path MTU Discovery

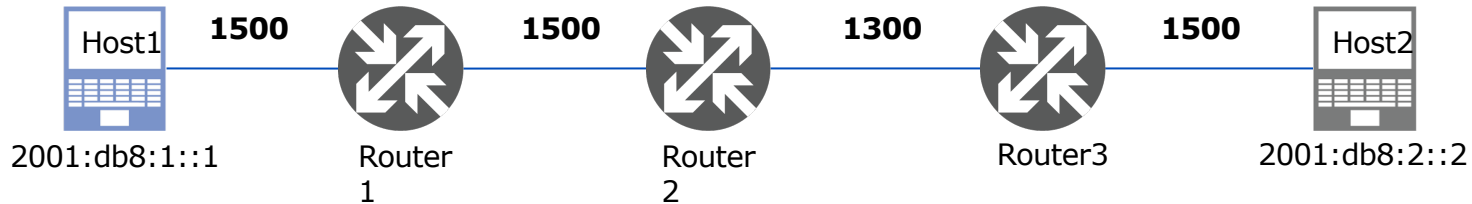


Link MTU values are marked on each link.

Host1 :
MTU cache=**1500**

Packet 1
size=1500bytes

Path MTU Discovery



Host1 :
MTU cache=**1500**

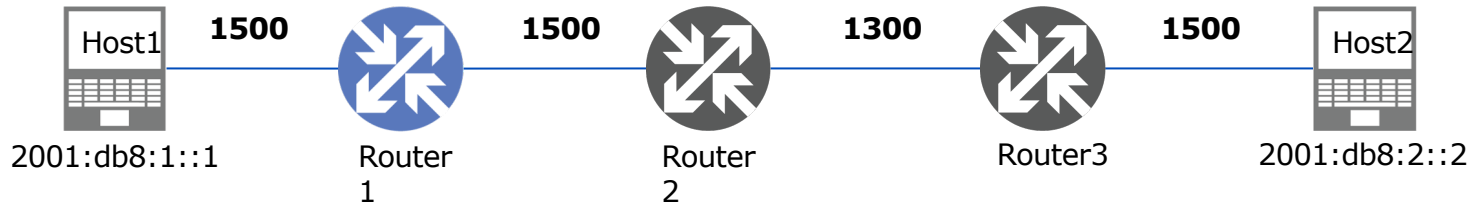
Link MTU values are marked on each link.

Packet 1
size=1500bytes

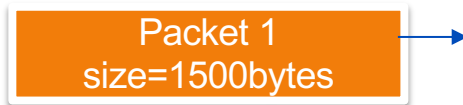
```
▶ Frame 1: 1510 bytes on wire (12080 bits), 1510 bytes captured (12080 bits) on 0
▶ Ethernet II, Src: Vmware_23:16:87 (00:0c:29:23:16:87), Dst: Vmware 25:cf:a1 (00:0c:29:25:cf:a1)
▼ Internet Protocol Version 6, Src: 2001:db8:1::1, Dst: 2001:db8:2::2
    0110 .... = Version: 6
    ▶ .... 0000 0000 .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
    .... 1010 1000 1001 1111 1000 = Flow Label: 0xa89f8
    Payload Length: 1456
    Next Header: Fragment Header for IPv6 (44)
    Hop Limit: 64
    Source: 2001:db8:1::1
    Destination: 2001:db8:2::2
    ▶ Fragment Header for IPv6
▶ [data]
```

Captured packets are available:
<https://www.cloudshark.org/captures/7dd0b50eb768>

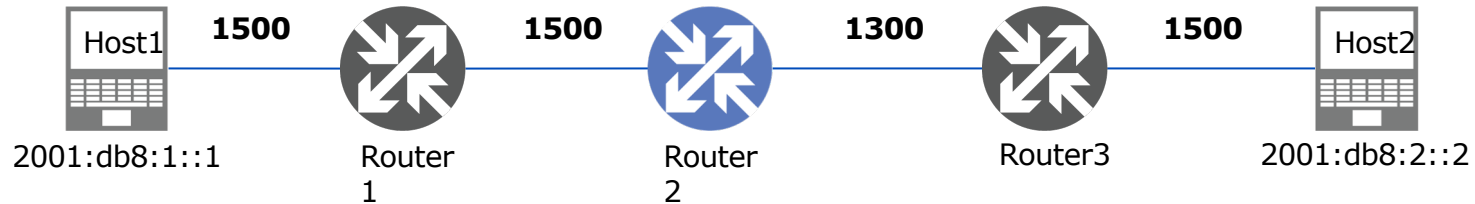
Path MTU Discovery



Link MTU values are marked on each link.



Path MTU Discovery

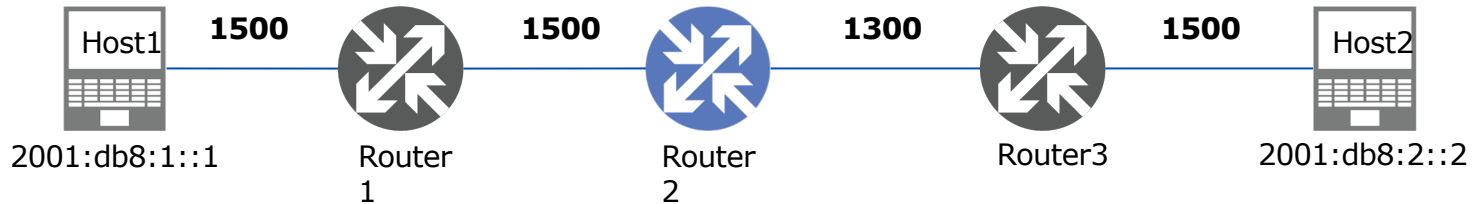


Link MTU values are marked on each link.

Packet 1
size=1500bytes

Because $1300 < 1500$, the packet 1 cannot be transmitted.

Path MTU Discovery



Link MTU values are marked on each link.

Packet 1
size=1500bytes

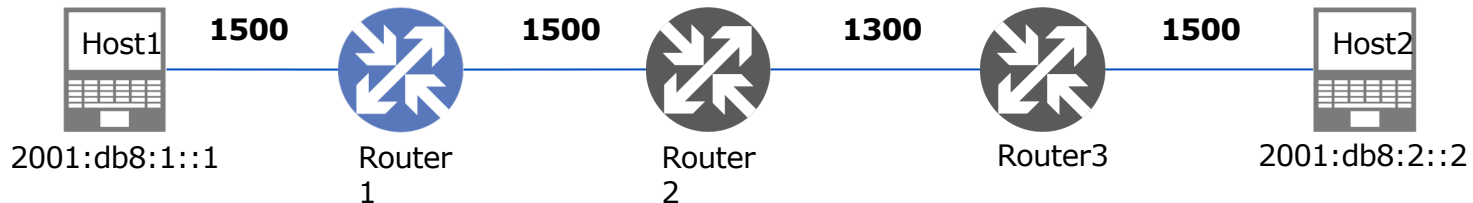


Drop!

Because $1300 < 1500$, the
Packet 1 cannot be transmitted.

← ICMPv6 Error:
Packet size too big!
MTU = 1300

Path MTU Discovery



Link MTU values are marked on each link.

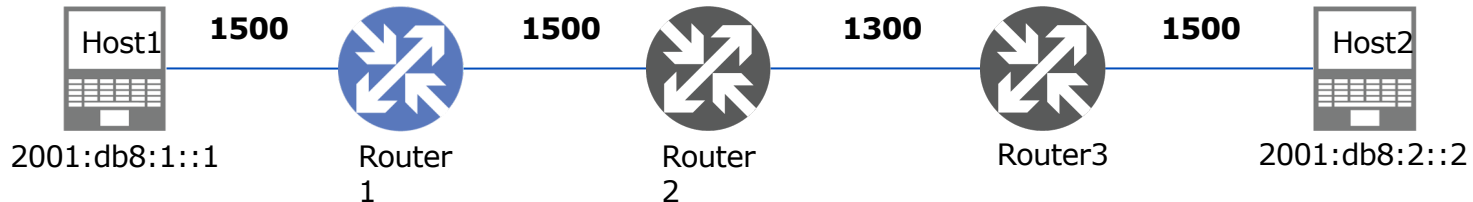


Because $1300 < 1500$, the packet 1 cannot be transmitted.



Host1 Update :
MTU cache=**1300**

Path MTU Discovery



```

> Frame 3: 1294 bytes on wire (10352 bits), 1294 bytes captured (10352 bits)
> Ethernet II, Src: Vmware_25:cf:a1 (00:0c:29:25:cf:a1), Dst: Vmware_23:16:87 (00:0c:29:23:16:87)
> Internet Protocol Version 6, Src: 2001:db8:12::2, Dst: 2001:db8:1::1
> Internet Control Message Protocol v6
  Type: Packet Too Big (2)
  Code: 0
  Checksum: 0x2e57 [correct]
  Checksum Status: Good
  MTU: 1300
  > Internet Protocol Version 6, Src: 2001:db8:1::1, Dst: 2001:db8:2::2
  > data

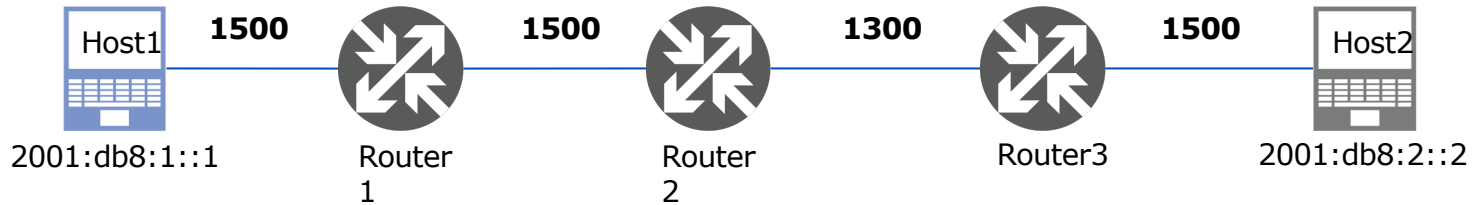
```

ICMPv6 Error:
Packet size too big!
MTU = 1300

Captured packets are available:

<https://www.cloudshark.org/captures/7dd0b50eb768>

Path MTU Discovery

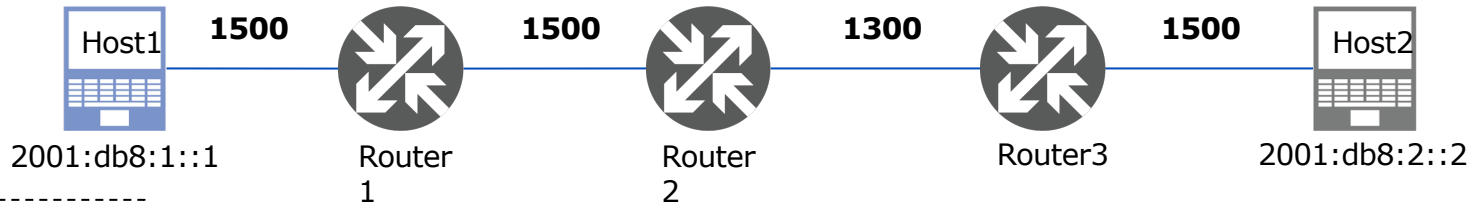


Link MTU values are marked on each link.

Host1 :
MTU cache=**1300**

Packet 2
size=1300bytes

Path MTU Discovery



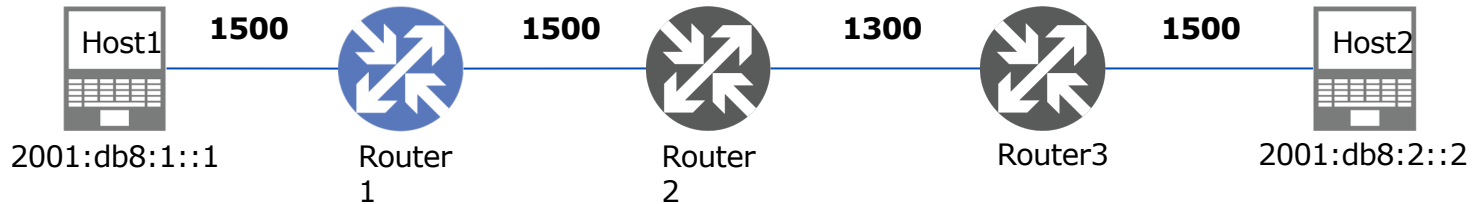
Host1 :
MTU cache=**1300**

Link MTU values are marked on each link.

Packet 2
size=1300bytes

```
▶ Frame 4: 1310 bytes on wire (10480 bits), 1310 bytes captured (10480 bits) on 0
▶ Ethernet II, Src: Vmware_23:16:87 (00:0c:29:23:16:87), Dst: Vmware 25:cf:a1 (00:0c:29:25:cf:a1)
▼ Internet Protocol Version 6, Src: 2001:db8:1::1, Dst: 2001:db8:2::2
    0110 .... = Version: 6
    ▶ .... 0000 0000 .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
    .... 1010 1000 1001 1111 1000 = Flow Label: 0xa89f8
    Payload Length: 1256
    Next Header: Fragment Header for IPv6 (44)
    Hop Limit: 64
    Source: 2001:db8:1::1
    Destination: 2001:db8:2::2
    ▶ Fragment Header for IPv6
▶ [data]
```

Path MTU Discovery

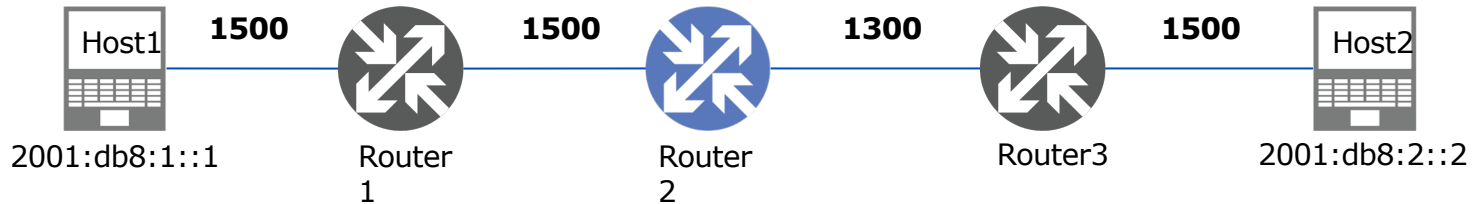


Host1 :
MTU cache=1300

Link MTU values are marked on each link.

Packet 2
size=1300bytes

Path MTU Discovery

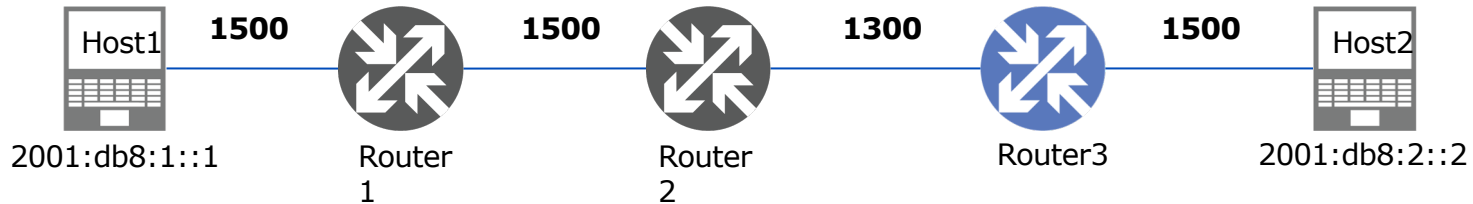


Host1 :
MTU cache=1300

Link MTU values are marked on each link.

Packet 2
size=1300bytes

Path MTU Discovery

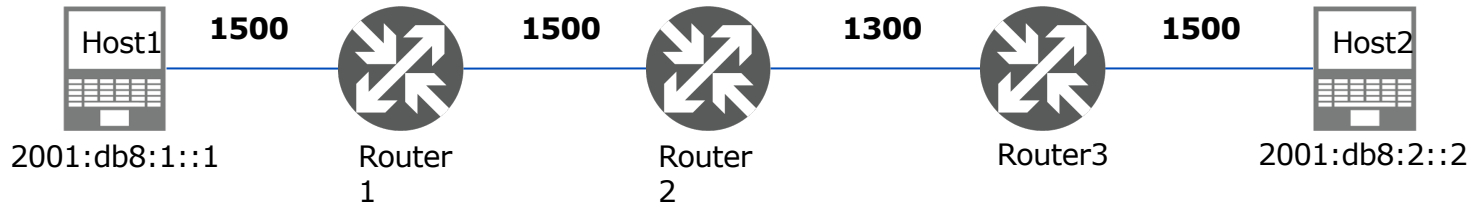


Host1 :
MTU cache=1300

Link MTU values are marked on each link.

Packet 2
size=1300bytes

Path MTU Discovery



Host1 :
MTU cache=1300

Link MTU values are marked on each link.

Packet 2
size=1300bytes

Path MTU = 1300



Questions

