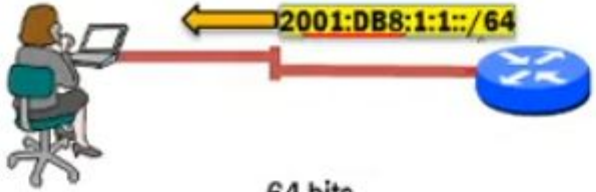
IPv6(Continued..)

EUI-64 (Extended Unique Identifier - 64)

How do routers automatically generate a unique interface ID

modified EUI-64 (Extended Unique Identifier)

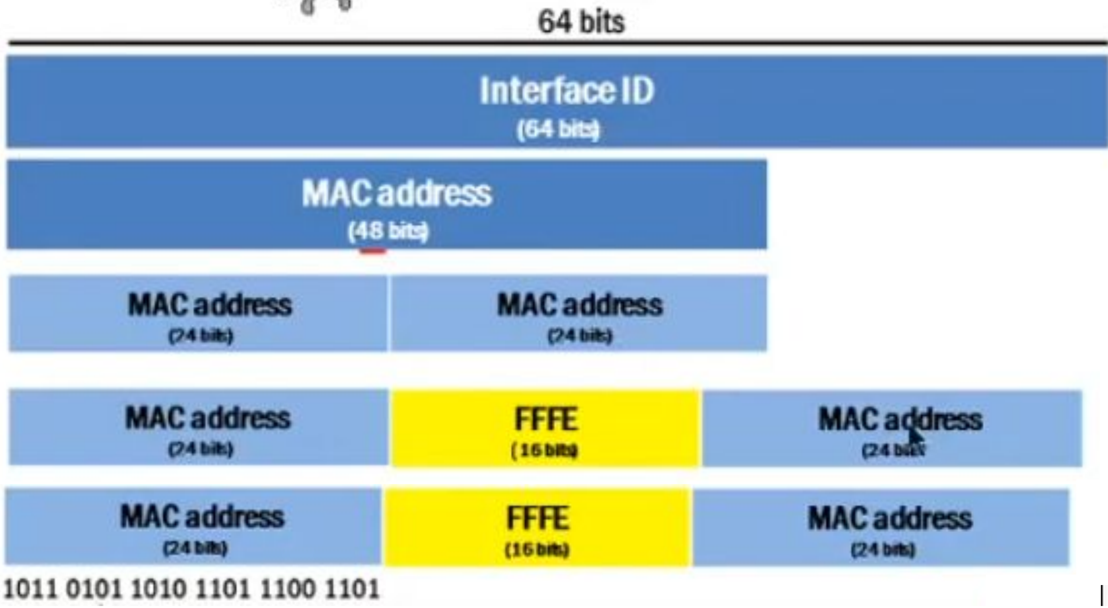


Take MAC address of the interface

Split MAC into two parts

Insert FFFE between MAC parts

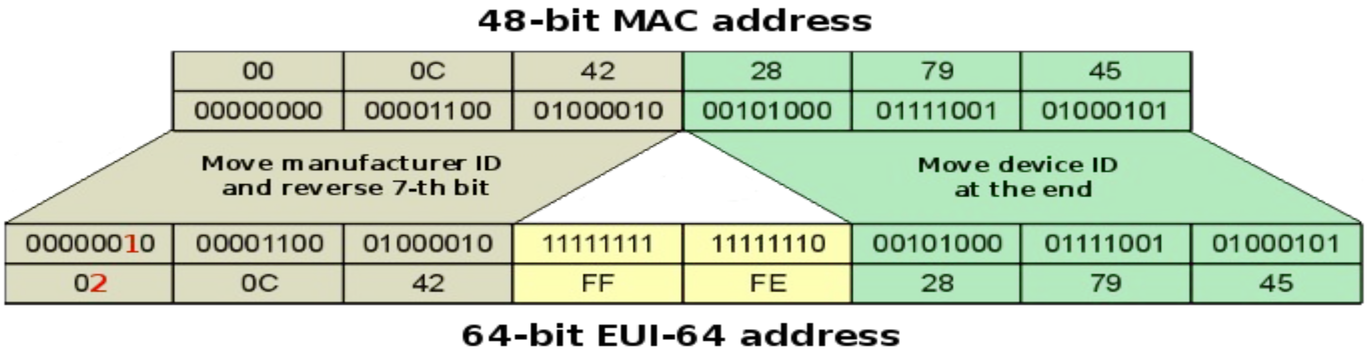
Invert the 7th bit of first MAC part



How to derive 64 bit Interface Id from 48 bit MAC Address?

```
|0              7 8             5 6             3|
+----+----+----+----+----+----+
|cccc|ccug|cccc|cccc|cccc|cccc|
+----+----+----+----+----+----+
```

U/L bit



When IANA assigns an Organizationally Unique Identifier (OUI) to a NIC card vendor, the 7th bit will be 0, indicating the OUI was universally assigned. **Should a user manually change their MAC address, this 7th bit would be set to 1, indicating the Ethernet address was locally administered**.

64-Bit Extended Unique Identifier (EUI-64)

Step #1: Split the 48-bit MAC address in the middle
Step #2: Insert FF.FE in the middle
Step #3: Change format to use a colon delimiter
Step #4: Convert the first eight bits to binary
Step #5: Flip the 7th bit
Step #6: Convert the first eight bits back into hexadecimal

0015.2BE4.9B60

0015.2BFF.FEE4.9B60

0015:2BFF:FEE4:9B60

00000010

0215:2BFF:FEE4:9B60

Problems:

1. Given a MAC Address : 00:1B:44:11:3A:B7

Find the Link Local IPv6 Address.

1. Assume that the network prefix is : 2001:DB8:0:1::/64 and MAC - 00:1B:44:11:3A:B7

   Find the IPv6 address. What type of IPv6 address is this?

IPv6 Multicast Address:

- Always has the prefix : FF00::/8

Flags	
R = 0 R = 1	No embedded RP Embedded RP
P = 0 P = 1	Not based on unicast Based on unicast
T = 0 T = 1	Permanent address (IANA assigned) Temporary address (local assigned)

Scope	
1	Node
2	Link
3	Subnet
4	Admin
5	Site
8	Organisation
E	Global

Note: Multicast addresses can only be destination addresses and not source addresses.

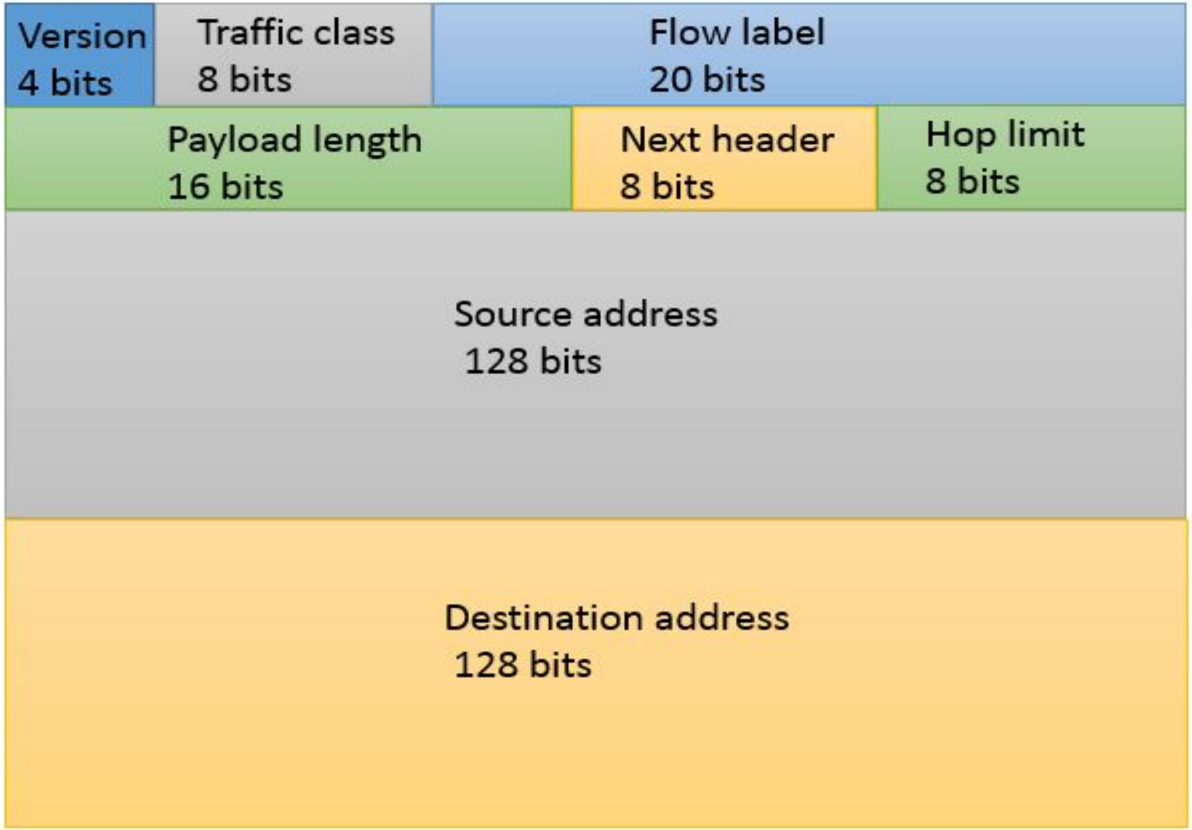
Few Multicast Addresses:

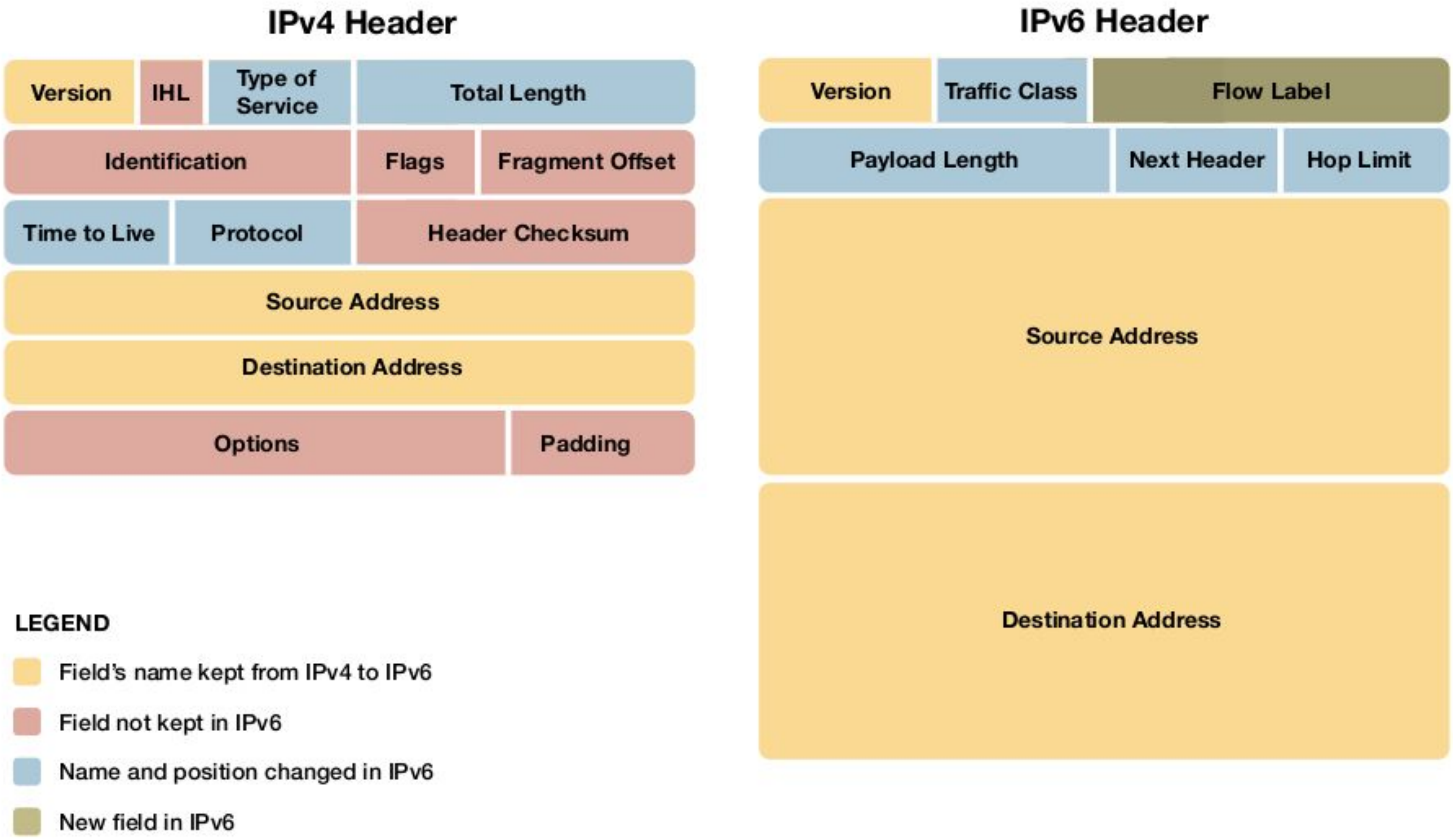
Prefix	Flag	Scope	Predefined Group ID	Compressed Format	Description (IPv6 assumed)
FF	0	2	0:0:0:0:0:0:1	FF02::1	All-devices
FF	0	2	0:0:0:0:0:0:2	FF02::2	All-routers
FF	0	2	0:0:0:0:0:0:5	FF02::5	OSPF routers
FF	0	2	0:0:0:0:0:0:6	FF02::6	OSPF DRs
FF	0	2	0:0:0:0:0:0:9	FF02::9	RIP routers
FF	0	2	0:0:0:0:0:0:A	FF02::A	EIGRP routers
FF	0	2	0:0:0:0:0:1:2	FF02::1:2	DHCP servers/relay agents

IPv4 Addressing Concepts and their IPv6 equivalents

IPv4 Address	IPv6 Address
Address Length – 32 bits	128 bits
Address Representation - decimal	hexadecimal
Internet address classes	Not applicable in IPv6
Multicast addresses (224.0.0.0/4)	IPv6 multicast addresses (FF00::/8)
Broadcast addresses	Not applicable in IPv6
Unspecified address is 0.0.0.0	Unspecified address is ::
Loopback address is 127.0.0.1	Loopback address is ::1
Public IP addresses	Global unicast addresses
Private IP addresses (10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16)	Site-local addresses (FEC0::/10)
Autoconfigured addresses (169.254.0.0/16)	Link-local addresses (FE80::/64)

IPv6 Header Format




IPv4 Header
Version
IHL
Type of Service
Total Length
Identification
Flags
Fragment Offset
Time to Live
Protocol
Header Checksum
Source Address
Destination Address
Options
Padding
IPv6 Header
Version
Traffic Class
Flow Label
Payload Length
Next Header
Hop Limit
Source Address
Destination Address
LEGEND
Field's name kept from IPv4 to IPv6
Field not kept in IPv6
Name and position changed in IPv6
New field in IPv6

80% of the Header is IPv6 Addresses (Source and Destination)

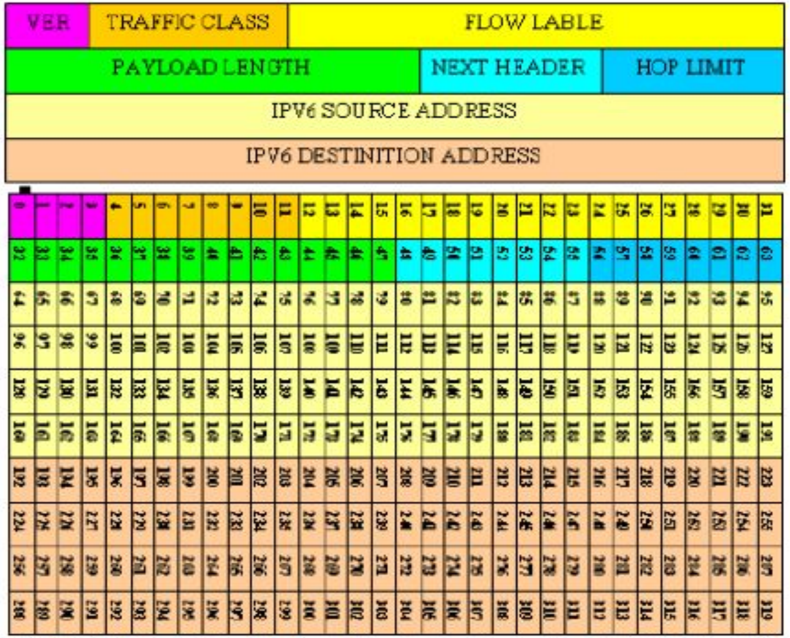
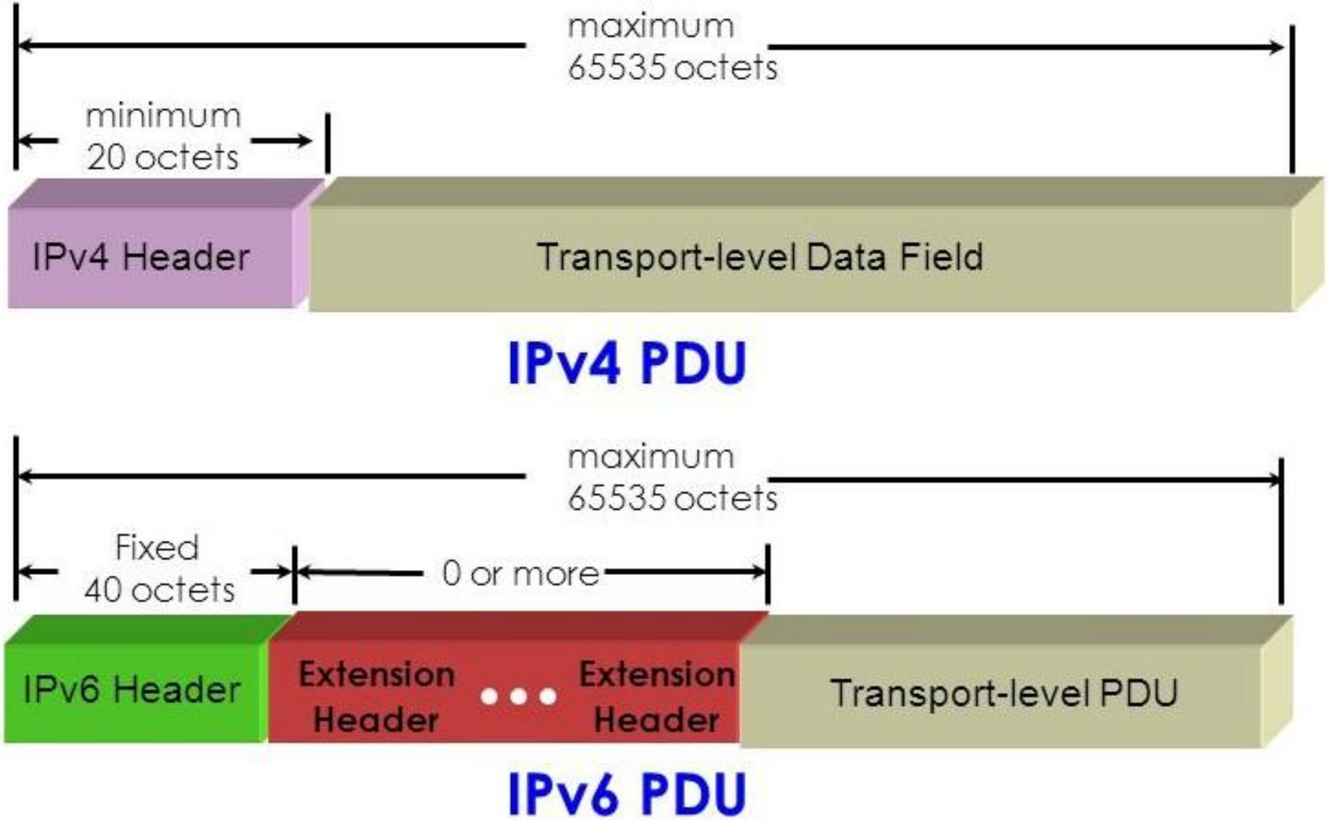


Figure2: 320-bits IPv6 basic header: each color represents part of the header so the first part of the figure shows the names of IPv6 parts while the second part shows bits numbers for each part, (VER ➔ 0 -3, TRAFFIC CLASS➔4-11, FLOW LABLE➔12-31, PAYLOAD LENGTH➔32-47, NEXT HEADER➔48-55, HOP LIMIT➔56-63, IPV6 SOURCE ADDRESS➔64➔191, IPV6 DESTINITION ADDRESS➔192➔319)

Paper: Batiha, Khaldoun. (2013). Improving IPV6 Addressing Types and Size. International journal of Computer Networks & Communications. 5. 41-51. 10.5121/ijcnc.2013.5404.

Ipv6 Fixed Header is followed by Extension Headers



IPv6 Header Fields

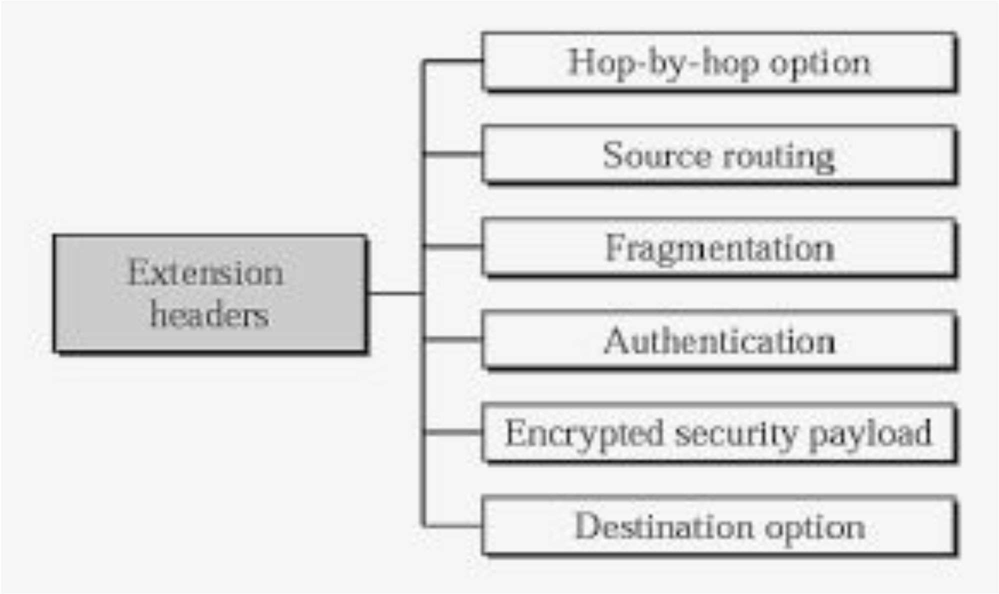
- Version : 4-bit field defines the version number of IP.
- Traffic class: This 8-bit field is used to distinguish different payloads with different delivery requirements.
- Flow label: 20-bit field that is designed to provide special handling for a particular flow of data.
- Payload length: 2-byte field defines the length of the IP datagram excluding the base header.
- Next header: 8-bit field defines the header that follows the base header in the datagram.
  - Either one of the optional extension headers used by IP or the header of an encapsulated packet such as UDP or TCP.
  - Each extension header also contains this field.
  - Protocol field in IPv4.

IPv6 Header Fields

- Hop limit: 8-bit field same as TTL.
- Source address: 128-bit
- Destination address: 128-bit. If Source routing is used, this field contains the address of the next router.

Extension Headers

- Goal is to give more functionality to the IP datagram.
- Six types of extension headers have been defined.



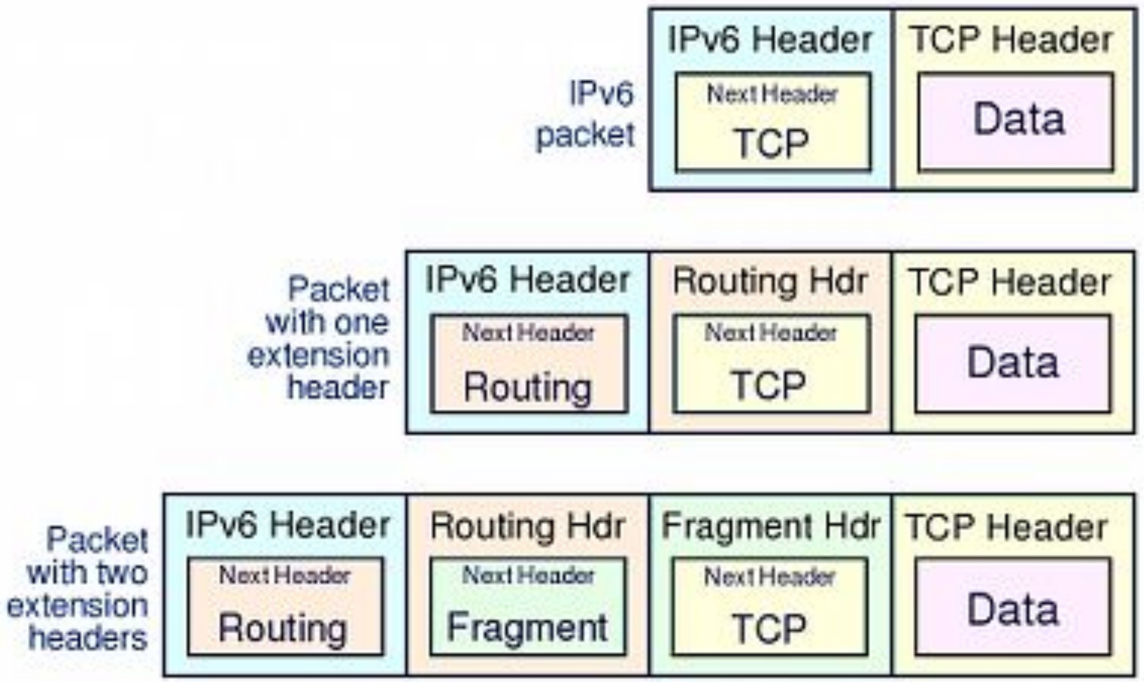
Next Headers:

Code	*Next Header*
0	Hop-by-hop option
2	ICMP
6	TCP
17	UDP
43	Source routing
44	Fragmentation
50	Encrypted security payload
51	Authentication
59	Null (No next header)
60	Destination option

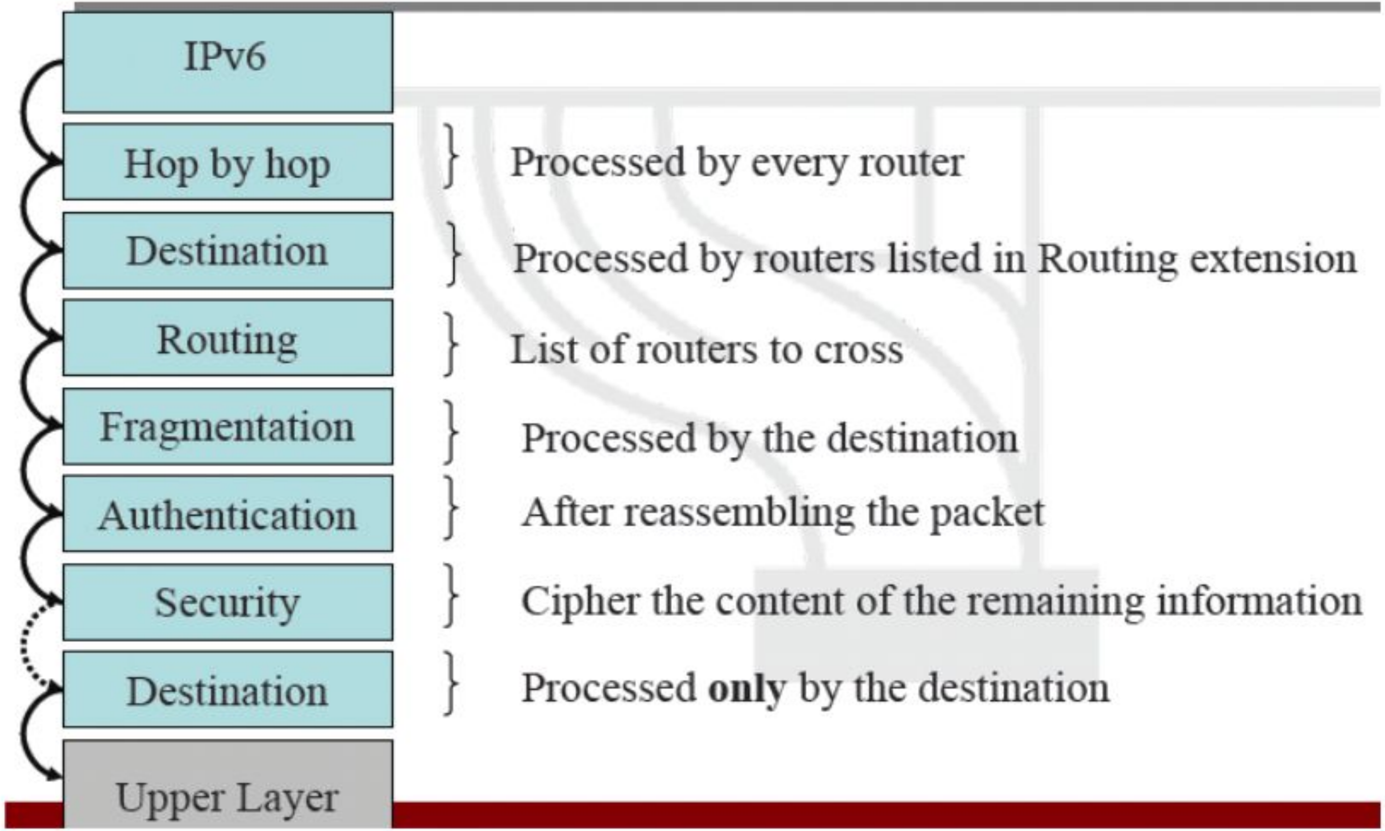
Routing Header

Next Header Value (Decimal)	Extension Header Name	Extension Header Length (Bytes)	Variable-Length Options (TLV) Used?	Extension Header Description
0	Hop-by-Hop Options	Variable	Yes	Used to carry optional information, which must be examined by every router along the path of the packet.
43	Routing	Variable	No	Allows the source of the packet to specify the path to the destination.
44	Fragment	8	No	Used to fragment IPv6 packets.
50	Encapsulating Security Payload (ESP)	Variable	No	Used to provide authentication, integrity, and encryption.
51	Authentication Header (AH)	Variable	No	Used to provide authentication and integrity.
60	Destination Options	Variable	Yes	Used to carry optional information that only needs to be examined by a packet's destination node(s).

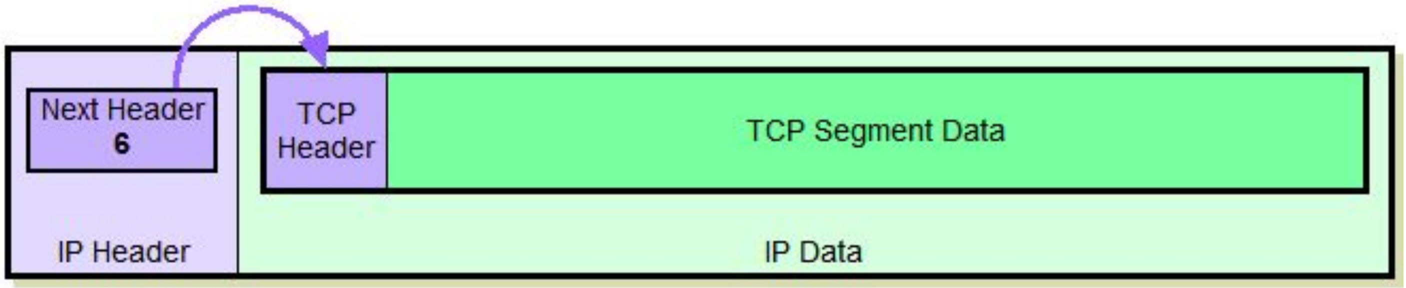
Extension Headers



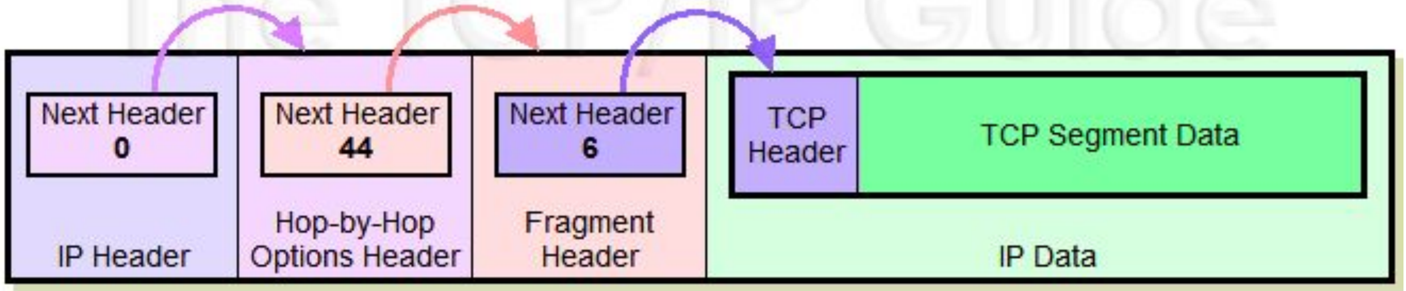
Order of Extension Headers:



Example:



IPv6 Datagram With No Extension Headers Carrying TCP Segment



IPv6 Datagram With Two Extension Headers Carrying TCP Segment

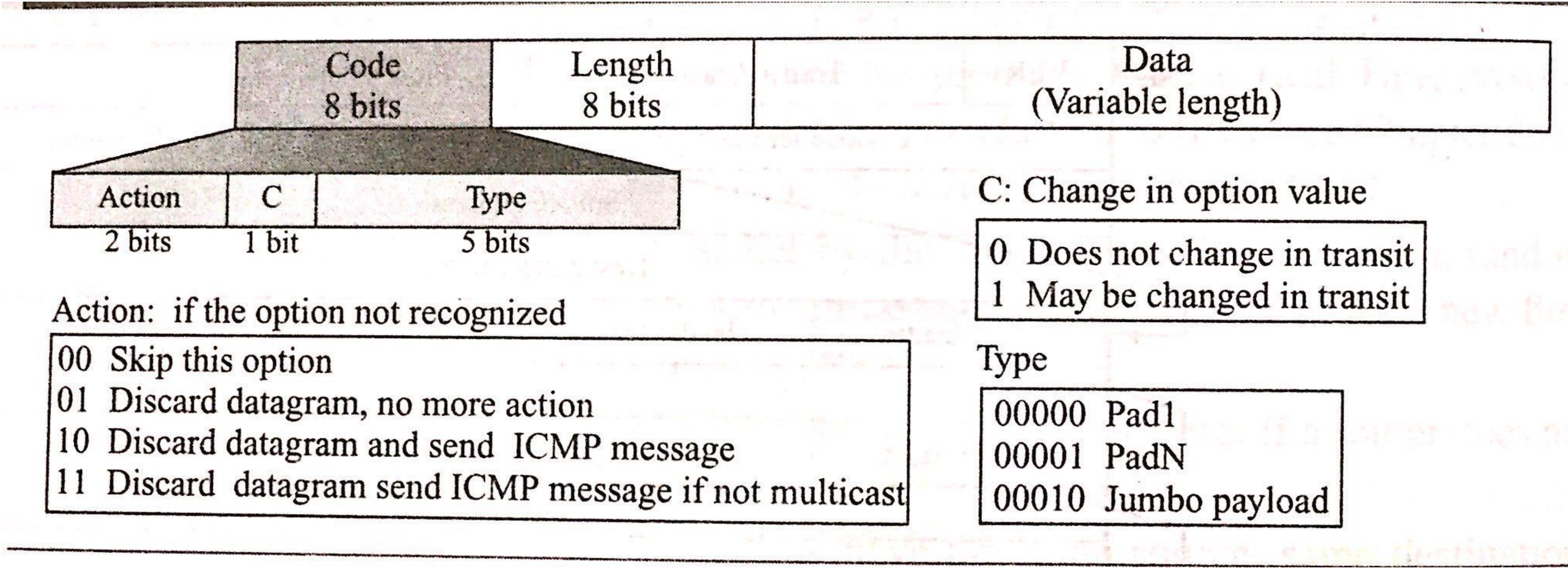
Hop-by-Hop option:

- When source needs to pass information to all routers visited by the datagram.
- Ex: debugging,control functions, if the length of the datagram is more than 65535 bytes.
- Hop by hop header format:

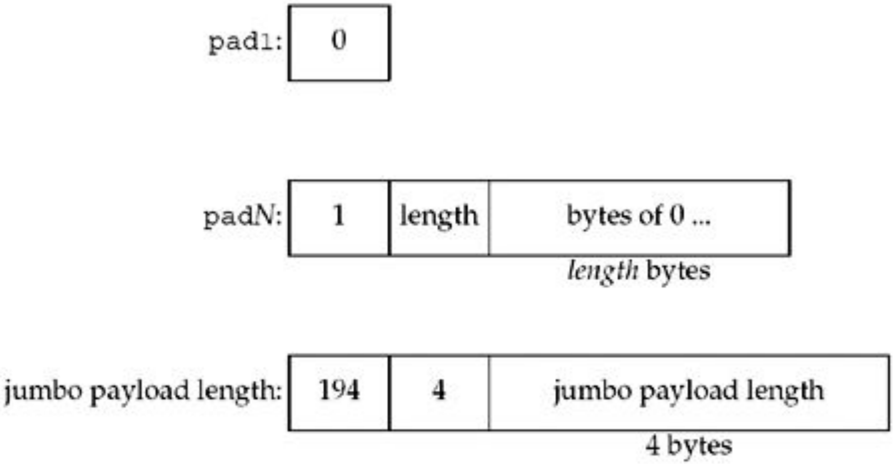
```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |  Hdr Ext Len  |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
|                                                               |
.                                                               .
.                            Options                            .
.                                                               .
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **OPTIONS:**
  - Type-Length-Value format.
  - The Option Type field both identifies the option and determines the way it is handled by the processing node.
  - The Option Length field indicates the number of bytes in the option
  - The option data is the specific data associated with the option.

The Code or Type field



Different types



Example:

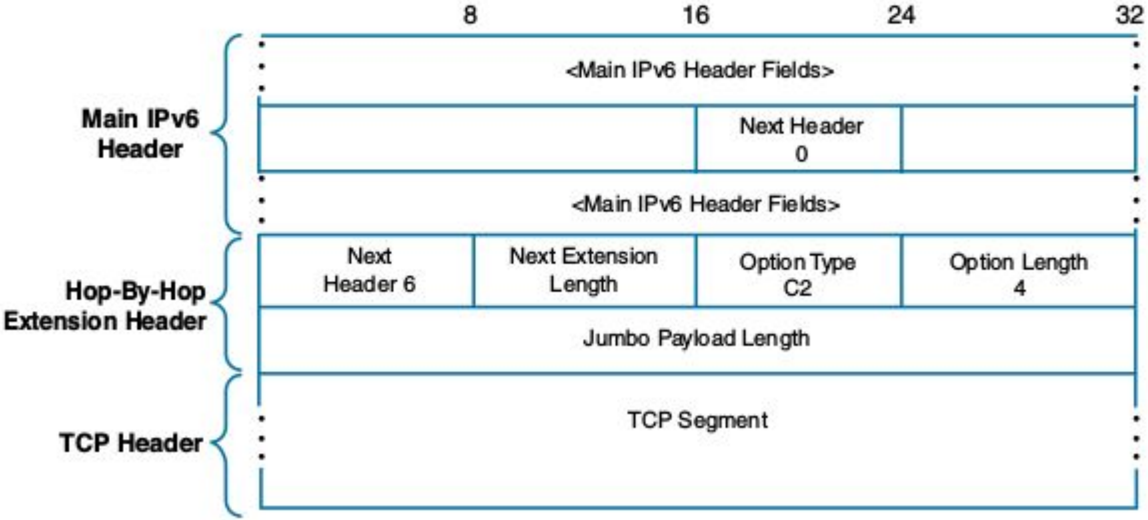


Figure 3-22 *Hop-by-Hop Extension Header with a Jumbo Payload Option*

- **Routing Header:**
  - Combines the concept of strict source route and loose source route options of IPv4.

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |  Hdr Ext Len  |  Routing Type | Segments Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
.                                                               .
.                       type-specific data                      .
.                                                               .
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Type 0 Routing header has the following format:

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |  Hdr Ext Len  | Routing Type=0| Segments Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Reserved                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                           Address[1]                          +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                           Address[2]                          +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                               .                               .
.                               .                               .
.                               .                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                           Address[n]                          +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |   Reserved    |      Fragment Offset    |Res|M|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Identification                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **Fragment Header:**

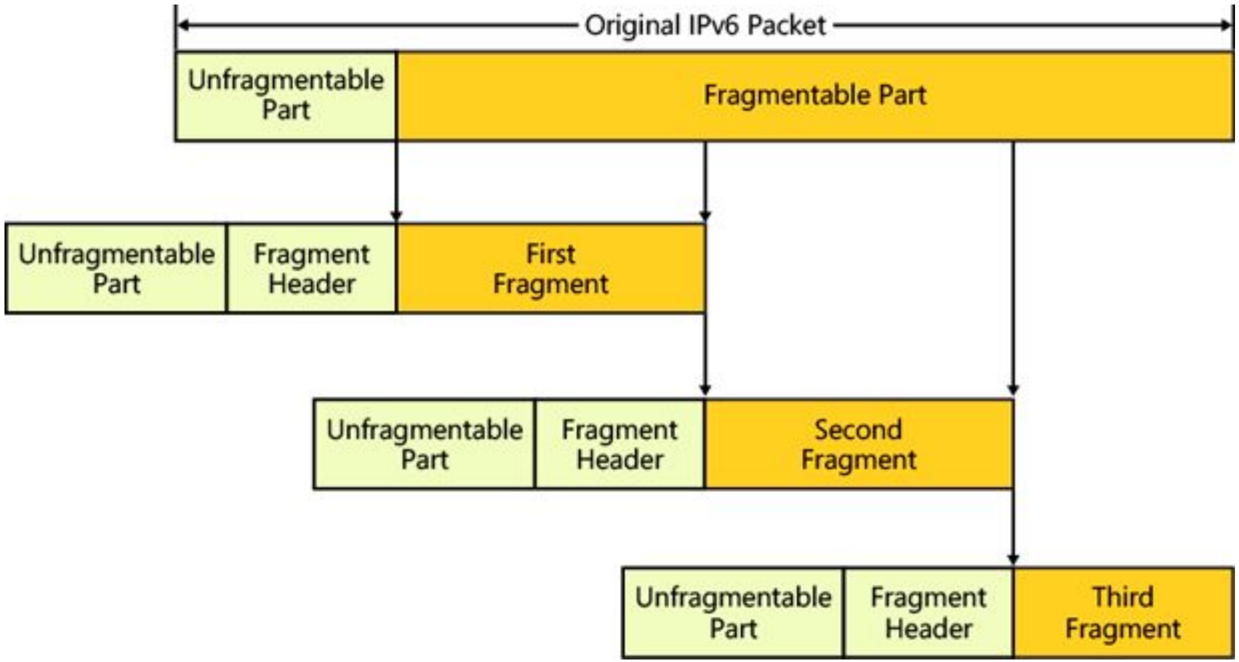
```
Fragment Offset     13-bit unsigned integer.  The offset, in
                    8-octet units, of the data following this
                    header, relative to the start of the
                    Fragmentable Part of the original packet.

Res                 2-bit reserved field.  Initialized to zero for
                    transmission; ignored on reception.

M flag              1 = more fragments; 0 = last fragment.

Identification      32 bits.
```

```
original packet:

+-----------------+--------------------//-----------------------+
|  Unfragmentable |                 Fragmentable                |
|       Part      |                     Part                    |
+-----------------+--------------------//-----------------------+
```

```
+------------------+-----------------+--------+--------+-//-+--------+
|  Per-Fragment    |Ext & Upper-Layer|  first | second |    |  last  |
|    Headers       |    Headers      |fragment|fragment|....|fragment|
+------------------+-----------------+--------+--------+-//-+--------+
```

fragment packets:

```
+------------------+---------+-------------------+----------+
|  Per-Fragment    |Fragment | Ext & Upper-Layer |  first   |
|    Headers       | Header  |   Headers         | fragment |
+------------------+---------+-------------------+----------+

+------------------+--------+-------------------------------+
|  Per-Fragment    |Fragment|    second                     |
|    Headers       | Header |   fragment                    |
+------------------+--------+-------------------------------+
                      o
                      o
                      o
+------------------+--------+----------+
|  Per-Fragment    |Fragment|    last  |
|    Headers       | Header | fragment |
+------------------+--------+----------+
```


Original IPv6 Packet
Unfragmentable Part
Fragmentable Part
Unfragmentable Part
Fragment Header
First Fragment
Unfragmentable Part
Fragment Header
Second Fragment
Unfragmentable Part
Fragment Header
Third Fragment

- **NO NEXT HEADER:**
  - Value 59 indicates there is nothing following the header.

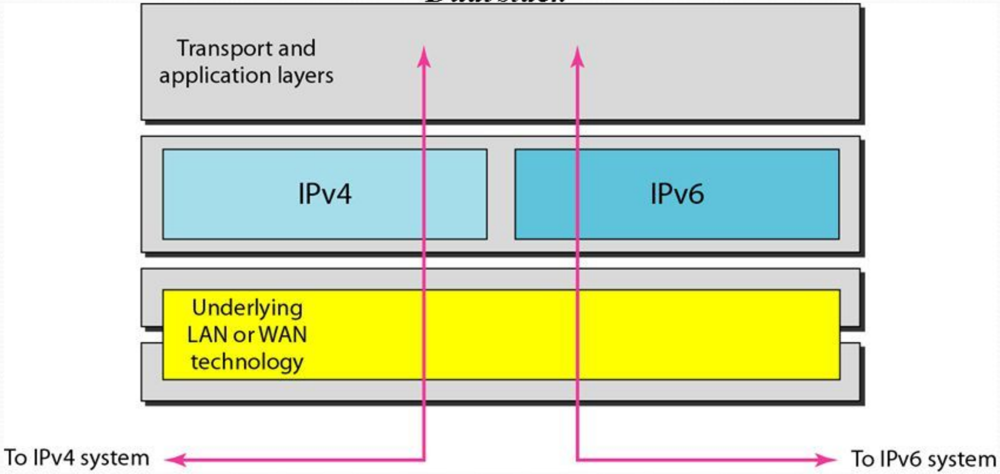
TRANSITION FROM IPv4 to IPv6

- Transition from IPv4 to IPv6 cannot happen suddenly, because of huge number of systems on the Internet
- Transition must be smooth.
- Three strategies devised by IETF:
  - Dual Stack
  - Tunneling
  - Header Translation

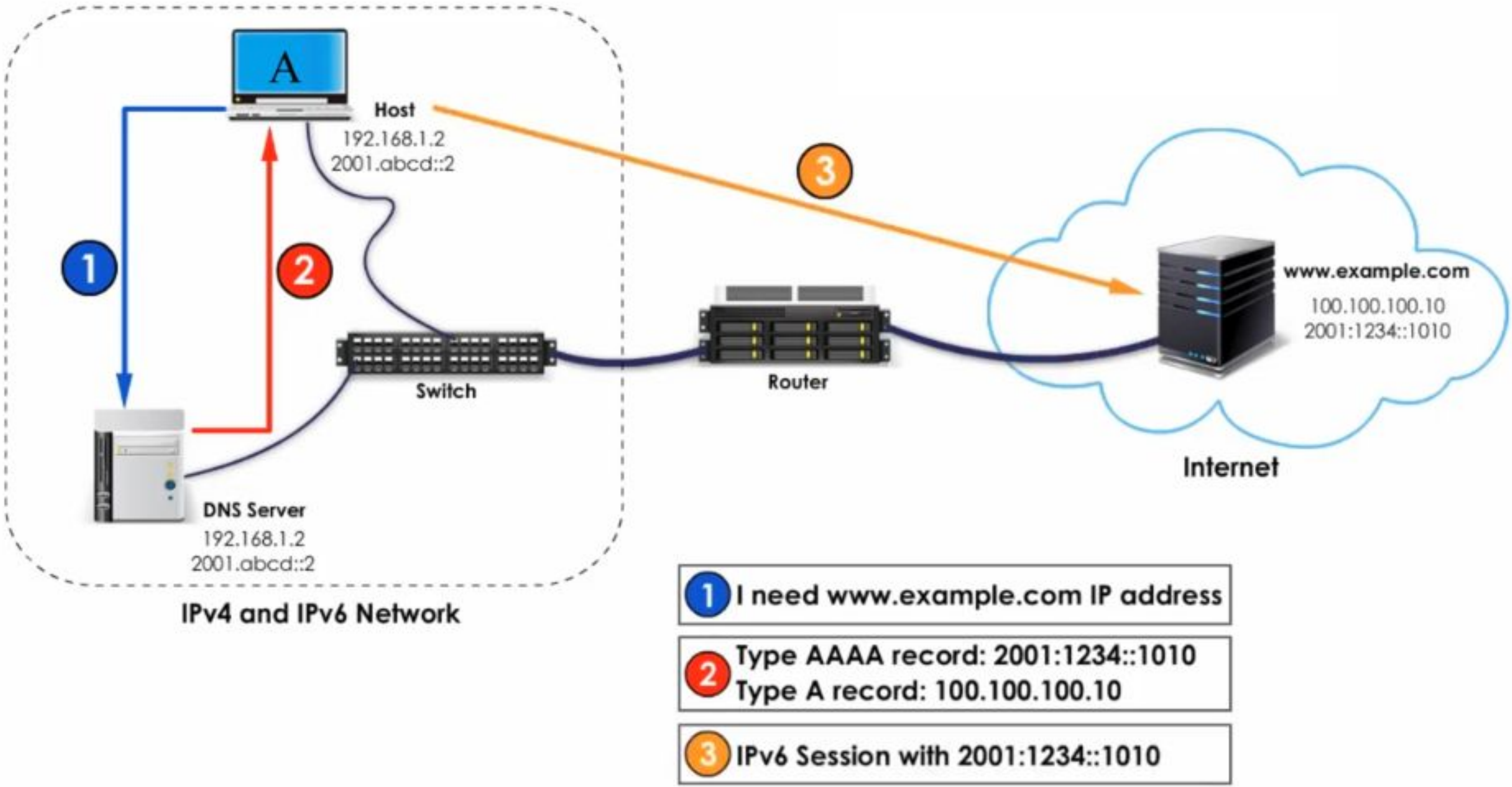
Transition: Dual Stack

TRANSITION FROM IPv4 TO IPv6

Dual stack

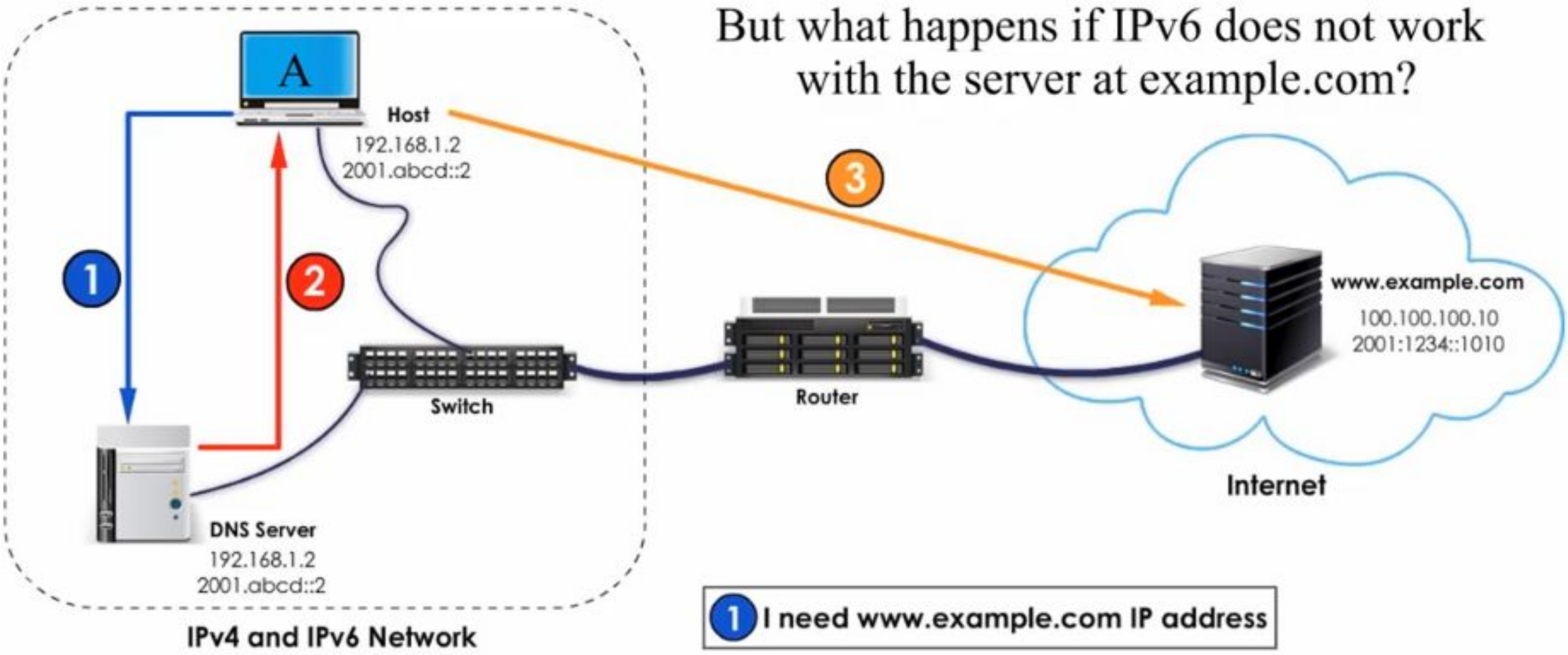


- **Each station must run IPv4 and IPv6 simultaneously until all the Internet uses IPv6.**
- **The source host queries the DNS. If the DNS returns an IPv4 address, the source host sends an IPv4 packet. If the DNS returns an IPv6 address, the source host sends an IPv6 packet.**



A
Host
192.168.1.2
2001.abcd::2
1
2
3
Switch
Router
www.example.com
100.100.100.10
2001:1234::1010
Internet
DNS Server
192.168.1.2
2001.abcd::2
IPv4 and IPv6 Network
1 I need www.example.com IP address
2 Type AAAA record: 2001:1234::1010
Type A record: 100.100.100.10
3 IPv6 Session with 2001:1234::1010

But what happens if IPv6 does not work with the server at example.com?

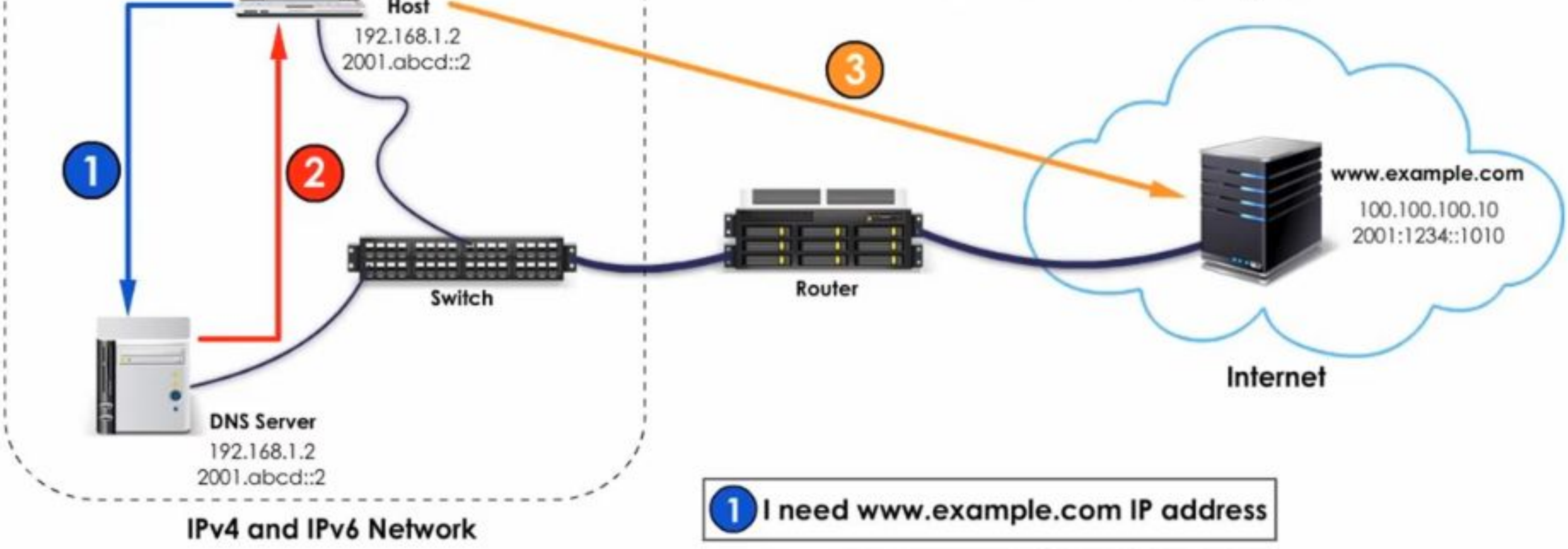


1 I need www.example.com IP address

2 Type AAAA record: 2001:1234::1010
Type A record: 100.100.100.10

3 IPv6 Session with 2001:1234::1010

The client would experience some delay.
The reason is simple.

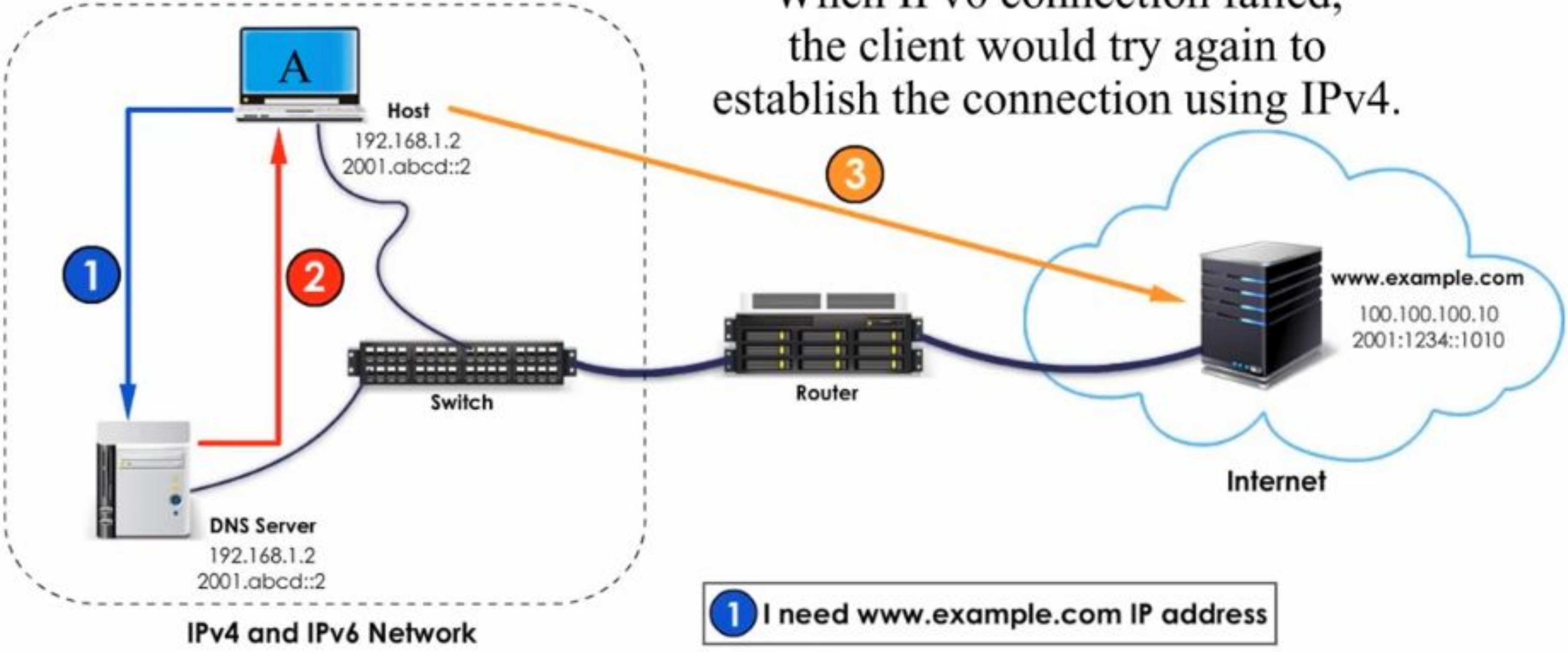


1 I need www.example.com IP address

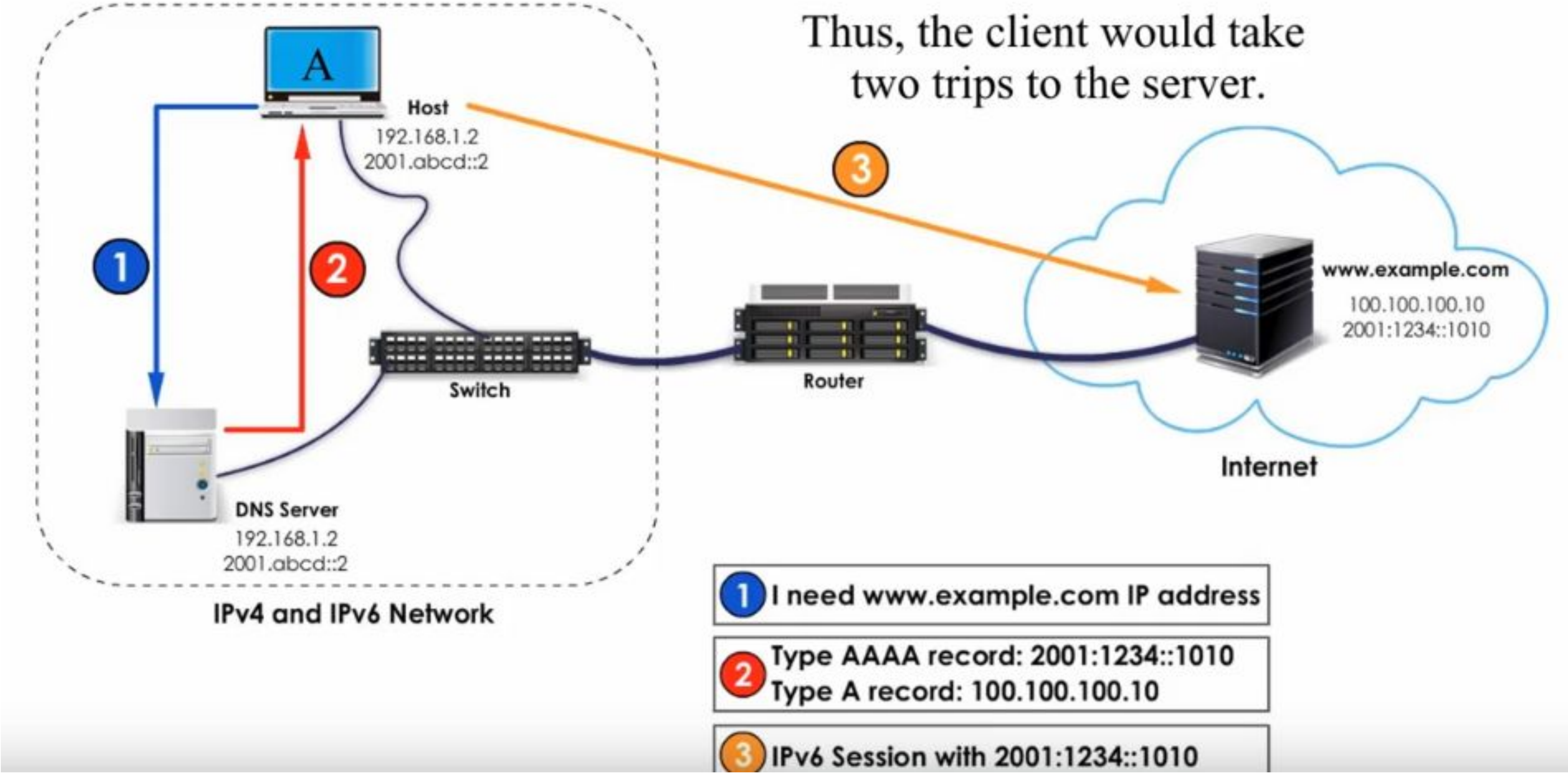
2 Type AAAA record: 2001:1234::1010
Type A record: 100.100.100.10

3 IPv6 Session with 2001:1234::1010

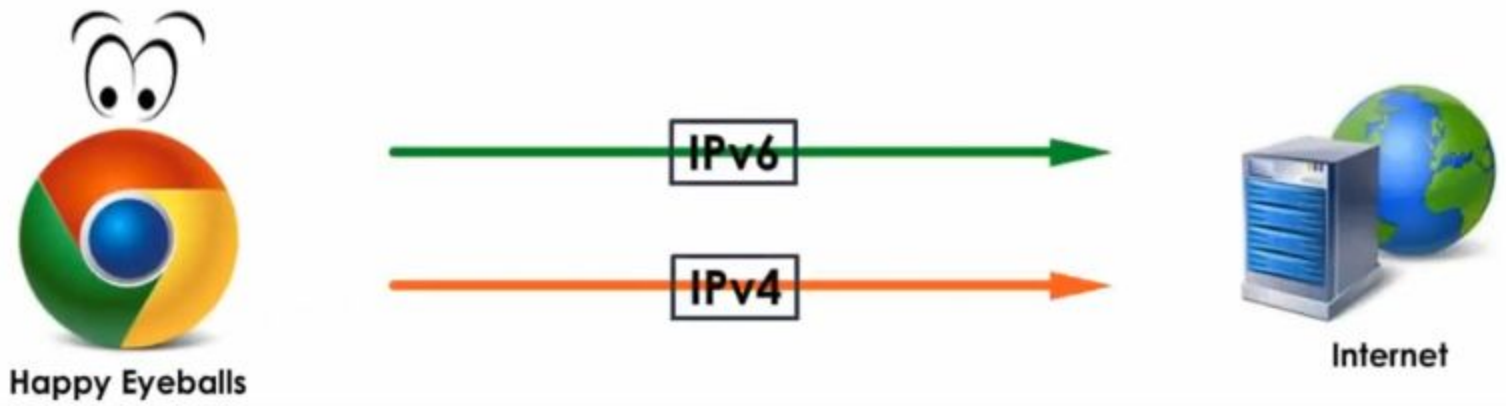
When IPv6 connection failed, the client would try again to establish the connection using IPv4.

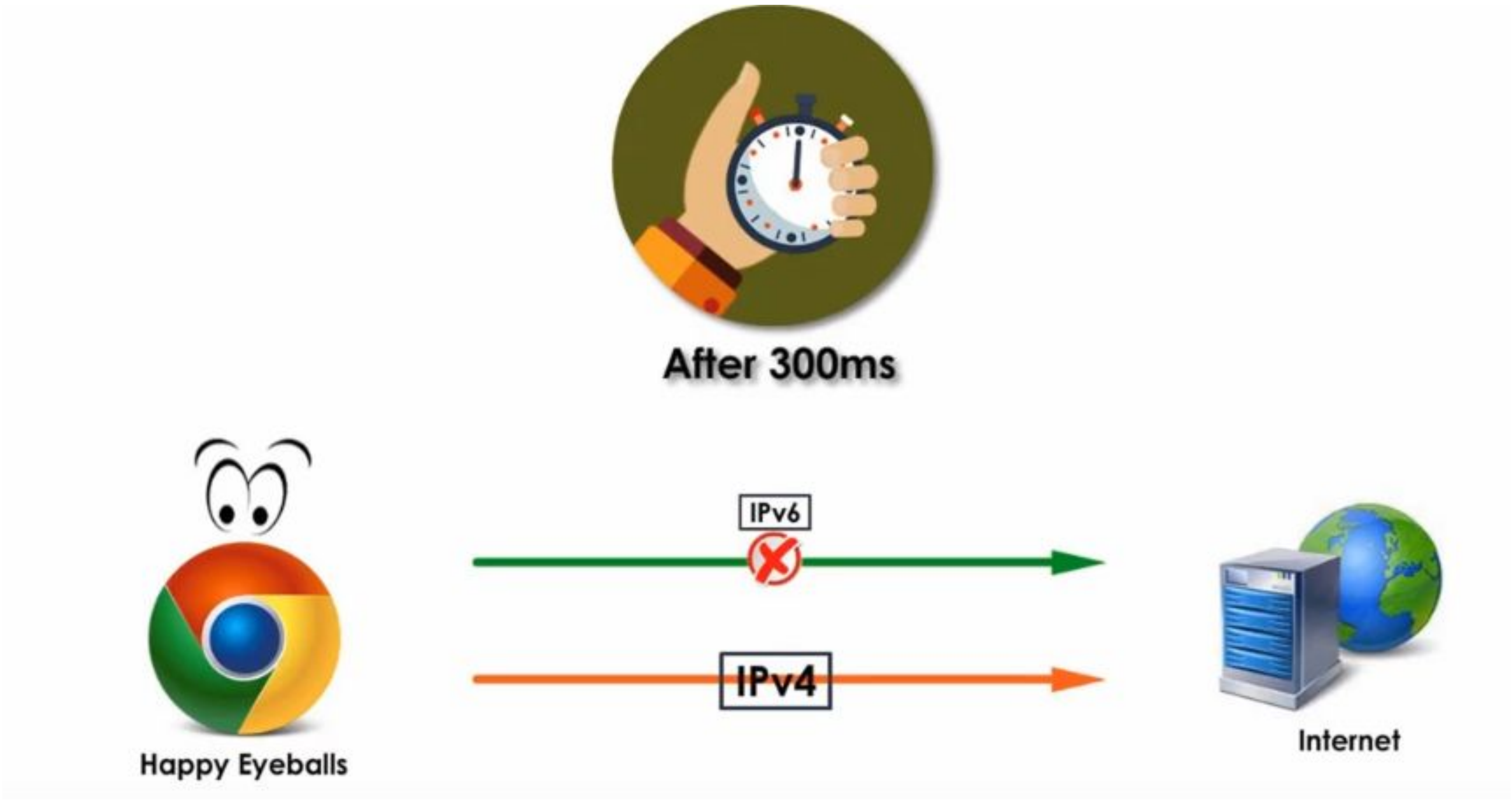


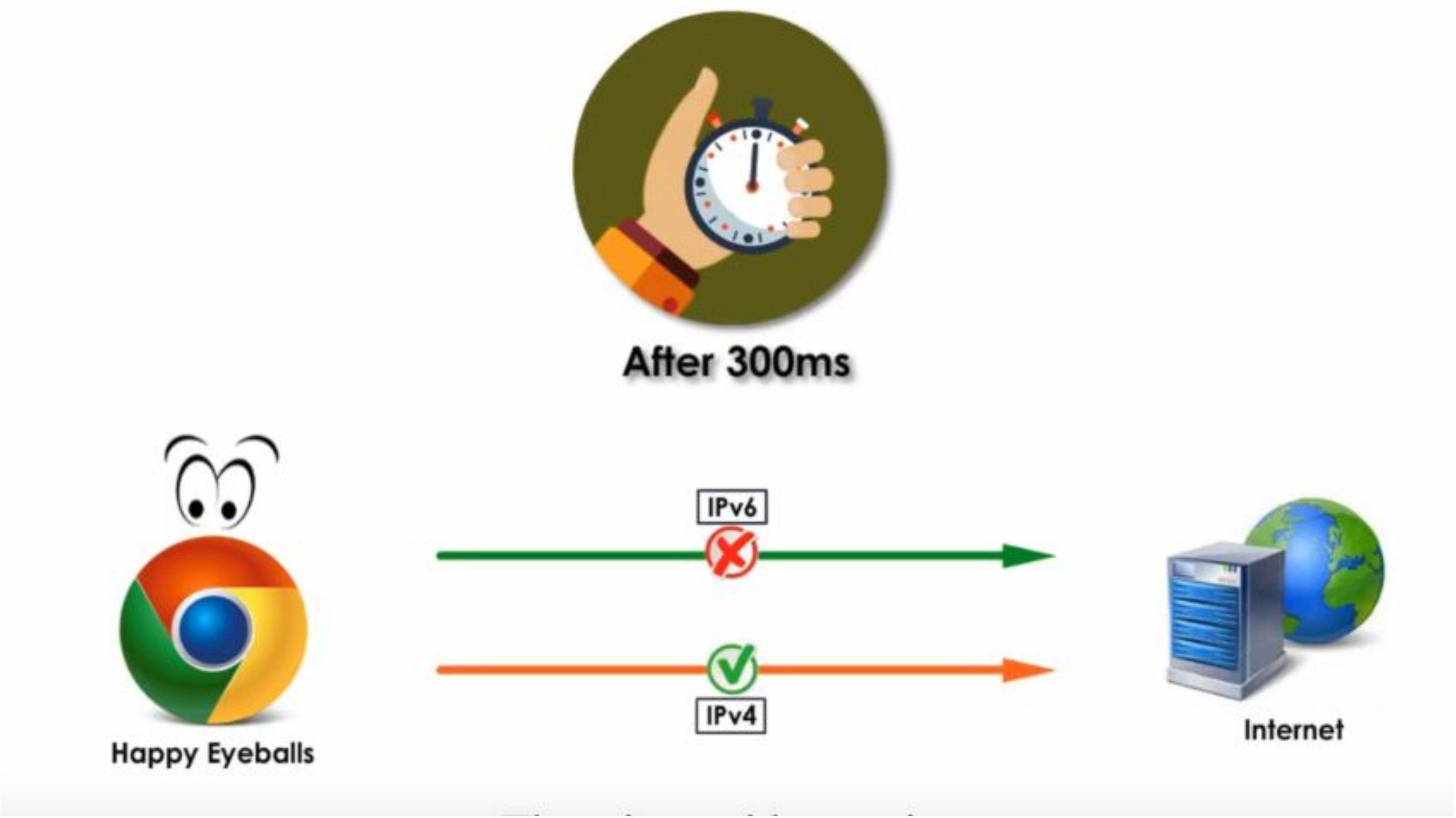
1	I need www.example.com IP address
2	Type AAAA record: 2001:1234::1010 Type A record: 100.100.100.10
3	IPv6 Session with 2001:1234::1010



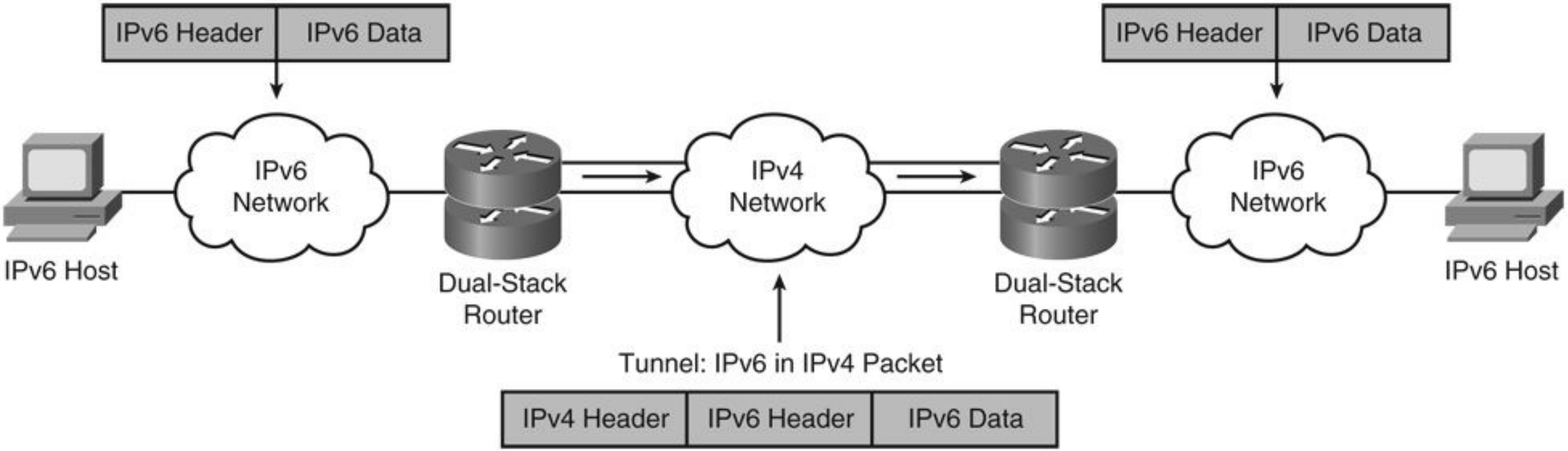
Thus, the client would take two trips to the server.
A
Host
192.168.1.2
2001.abcd::2
1
2
3
Switch
Router
DNS Server
192.168.1.2
2001.abcd::2
IPv4 and IPv6 Network
www.example.com
100.100.100.10
2001:1234::1010
Internet
1 I need www.example.com IP address
2 Type AAAA record: 2001:1234::1010
Type A record: 100.100.100.10
3 IPv6 Session with 2001:1234::1010

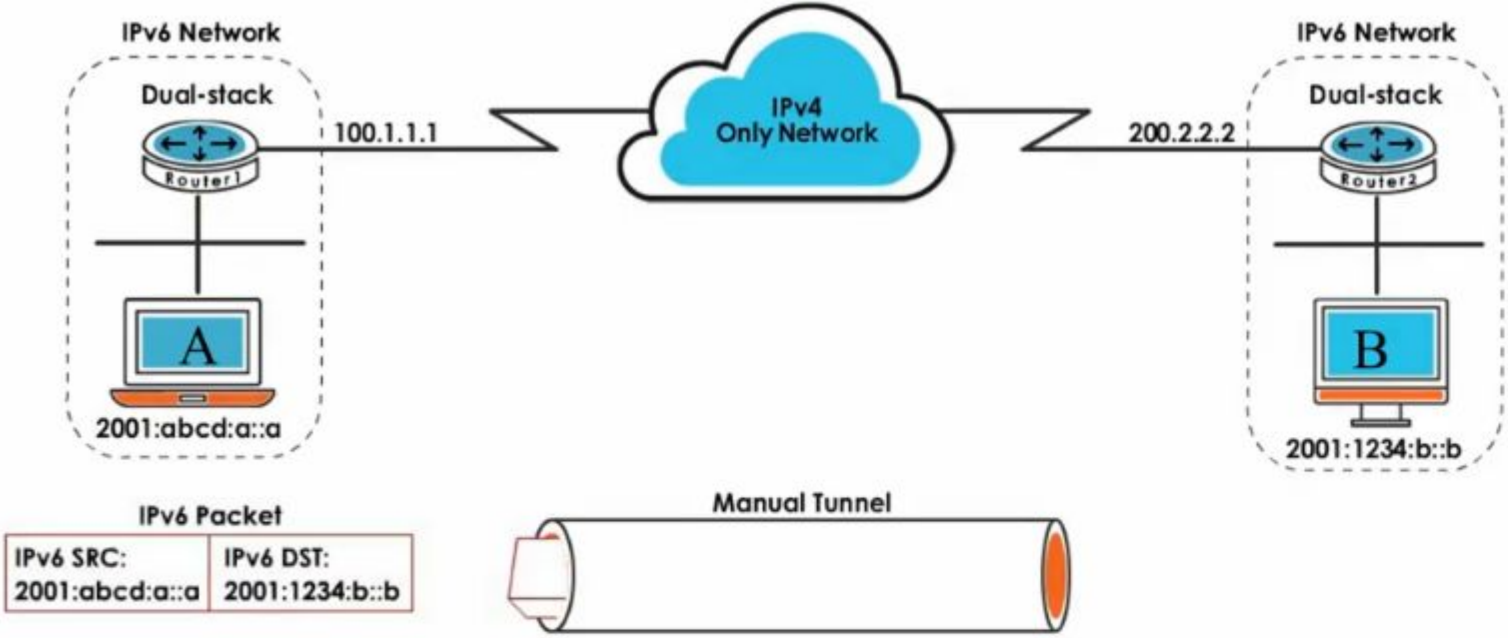

IPv6
IPv4
Happy Eyeballs
Internet

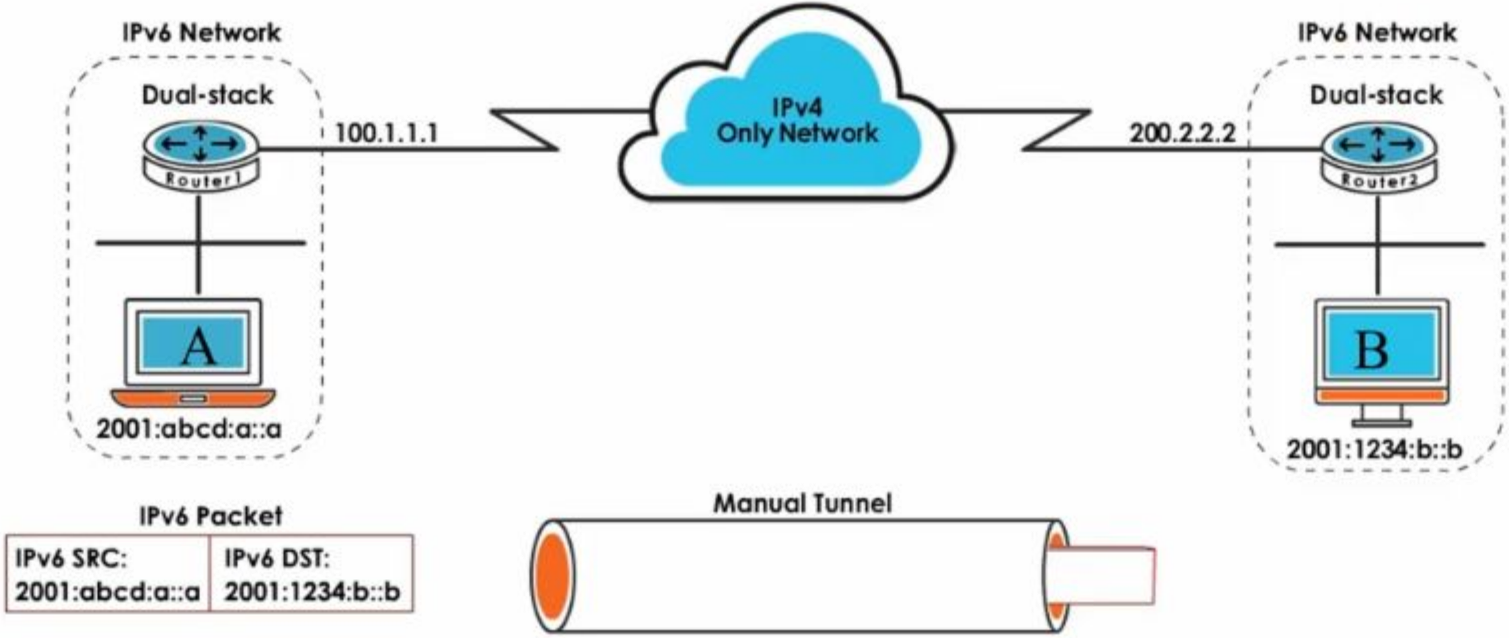

After 300ms
IPv6
IPv4
Happy Eyeballs
Internet

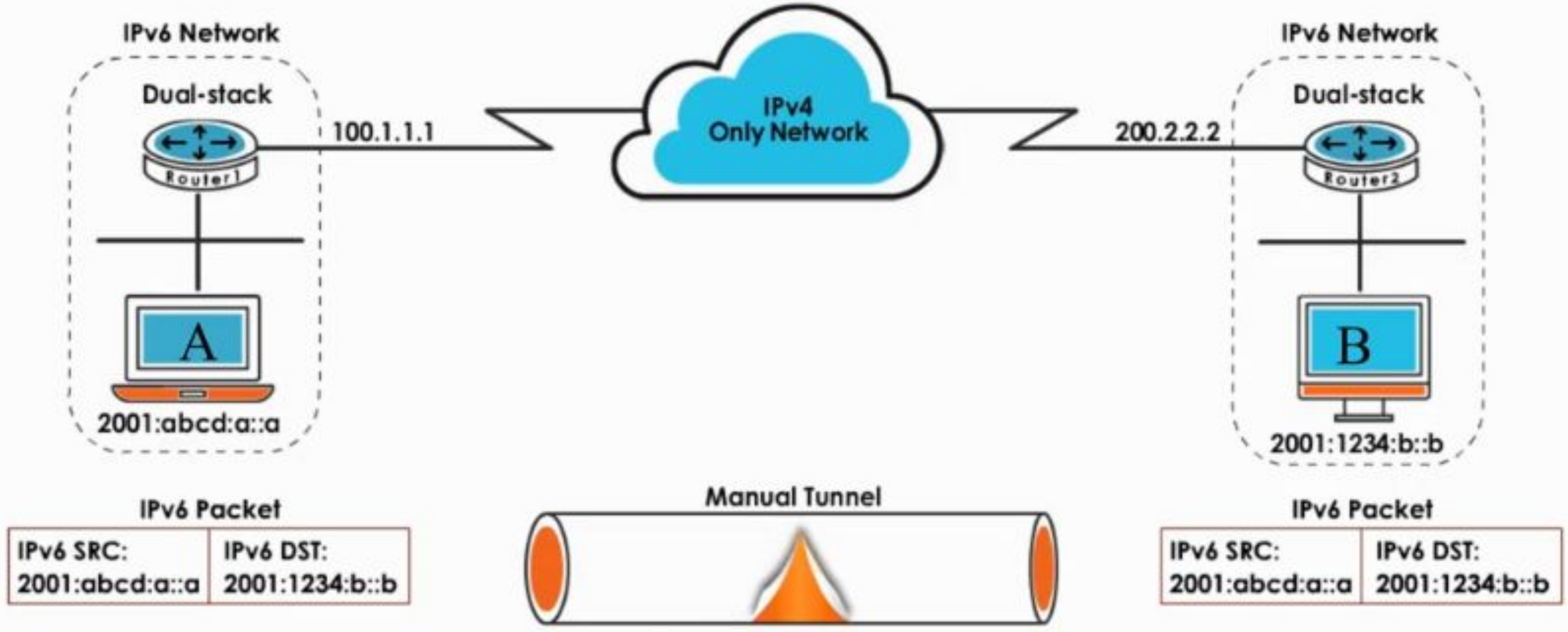

After 300ms
IPv6
IPv4
Happy Eyeballs
Internet

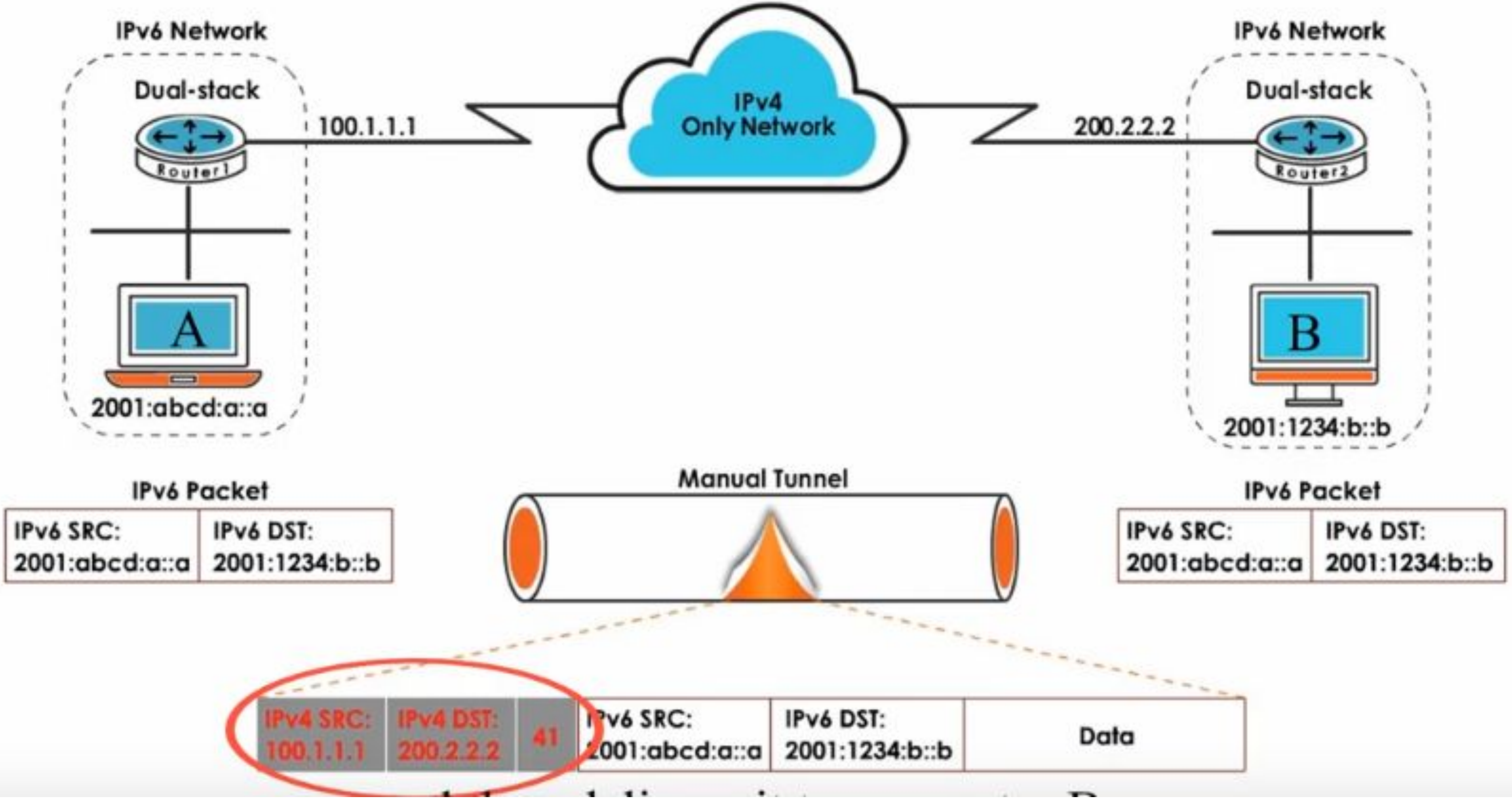
Transition: Tunneling



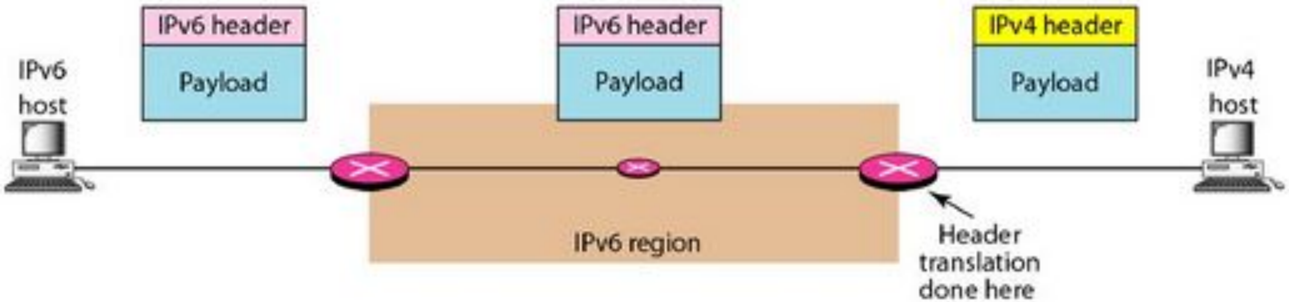

IPv6 Network
Dual-stack
Router1
100.1.1.1
A
2001:abcd:a::a
IPv4
Only Network
IPv6 Network
Dual-stack
200.2.2.2
Router2
B
2001:1234:b::b
IPv6 Packet
IPv6 SRC:
2001:abcd:a::a
IPv6 DST:
2001:1234:b::b
Manual Tunnel


IPv6 Network
Dual-stack
Router1
2001:abcd:a::a
100.1.1.1
IPv4
Only Network
200.2.2.2
IPv6 Network
Dual-stack
Router2
2001:1234:b::b
IPv6 Packet
IPv6 SRC:
2001:abcd:a::a
IPv6 DST:
2001:1234:b::b
Manual Tunnel


IPv6 Network
Dual-stack
Router1
A
2001:abcd:a::a
100.1.1.1
IPv4
Only Network
200.2.2.2
IPv6 Network
Dual-stack
Router2
B
2001:1234:b::b
IPv6 Packet
IPv6 SRC: 2001:abcd:a::a
IPv6 DST: 2001:1234:b::b
Manual Tunnel
IPv6 Packet
IPv6 SRC: 2001:abcd:a::a
IPv6 DST: 2001:1234:b::b


IPv6 Network
Dual-stack
Router1
100.1.1.1
IPv4
Only Network
200.2.2.2
IPv6 Network
Dual-stack
Router2
A
2001:abcd:a::a
B
2001:1234:b::b
IPv6 Packet
IPv6 SRC:
2001:abcd:a::a
IPv6 DST:
2001:1234:b::b
Manual Tunnel
IPv6 Packet
IPv6 SRC:
2001:abcd:a::a
IPv6 DST:
2001:1234:b::b
IPv4 SRC:
100.1.1.1
IPv4 DST:
200.2.2.2
41
IPv6 SRC:
2001:abcd:a::a
IPv6 DST:
2001:1234:b::b
Data

Transition: Header Translation



RULES FOR HEADER TRANSLATION:

- IPv6 address is changed to IPv4 address by extracting rightmost 32 bits.
- The value of IPv6 priority field is discarded
- The type of service field in IPv4 is set to zero
- The checksum is calculated for IPv4 and placed in corresponding field
- IPv6 flow label is ignored
- Compatible extension headers are converted to options and inserted in the IPv4 header. Some may have to be dropped
- Length of IPv4 header is calculated and inserted into the corresponding field.
- Total length of IPv4 packet is calculated and inserted in the corresponding field