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

<table>
<thead>
<tr>
<th>IPv4 Headers</th>
<th>IPv6 Headers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Version</td>
</tr>
<tr>
<td>IHL</td>
<td>Traffic Class</td>
</tr>
<tr>
<td>Type of Service</td>
<td>Flow Label</td>
</tr>
<tr>
<td>Total Length</td>
<td>Payload Length</td>
</tr>
<tr>
<td>Identification</td>
<td>Next Header</td>
</tr>
<tr>
<td>Flags</td>
<td>Hop Limit</td>
</tr>
<tr>
<td>Fragment offset</td>
<td>Source Address</td>
</tr>
<tr>
<td>Time to Live</td>
<td>Destination Address</td>
</tr>
<tr>
<td>Protocol</td>
<td>Options</td>
</tr>
<tr>
<td>Header Checksum</td>
<td>Padding</td>
</tr>
</tbody>
</table>

- **Not kept in IPv6**
- **Renamed in IPv6**
- **Same name and function**
- **New in IPv6**
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.
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:

- IPv6 header
- Hop-by-Hop Options header
- Destination Options header
- Routing header
- Fragment header
- Authentication header (RFC 4302)
- Encapsulating Security Payload header (RFC 4303)
- Destination Options header
- TCP header (Upper-Layer header)
Chaining Extension Headers

<table>
<thead>
<tr>
<th>Extension Header</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hop-by-Hop Options</td>
<td>0</td>
</tr>
<tr>
<td>Fragment</td>
<td>44</td>
</tr>
<tr>
<td>Destination Options</td>
<td>60</td>
</tr>
<tr>
<td>Routing</td>
<td>43</td>
</tr>
<tr>
<td>Authentication</td>
<td>51</td>
</tr>
<tr>
<td>Encapsulating Security Payload</td>
<td>50</td>
</tr>
</tbody>
</table>
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

IPv6 network

Host1
2001:db8:1::1

IPv6 Packet

IPv6 network

Host2
2001:db8:2::2
On the Source Node (Host1)

IPv6 Packet

IPv6 Header

TCP Segment

Host1

2001:db8:1::

1
On the Source Node (Host1)

IPv6 Packet

IPv6 Header

TCP Segment

2001:db8:1::1

Packet size > Path MTU, how to encapsulate the packet?
On the Source Node (Host1)

IPv6 Packet

IPv6 Header

TCP Segment

Host1

2001:db8:1::1

Divide the packet into fragments.
On the Source Node (Host1)

IPv6 Packet
IPv6 Header
TCP Segment

Host1
2001:db8:1::1

IPv6 Packet 1
IPv6 Packet 2
IPv6 Packet 3

Divide the packet into fragments.
On the Source Node (Host1)

IPv6 Packet

IPv6 Header

TCP Header

Fragment 1

Fragment 2

Fragment 3

IPv6 Packet

IPv6 Header

Fragment Header

IPv6 Packet

IPv6 Header

Fragment Header

IPv6 Packet

IPv6 Header

Fragment Header

Divide the packet into fragments.
On the Source Node (Host1)

IPv6 Packet

IPv6 Header

TCP Header

Fragment 1

Fragment 2

Fragment 3

IPv6 Header

Fragment Header

IPv6 Packet 1

IPv6 Header

Fragment Header

IPv6 Packet 2

IPv6 Header

Fragment Header

IPv6 Packet 3

Divide the packet into fragments.
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)

IPv6 Packet 1
- IPv6 Header
- Fragment Header
- TCP Header
- Fragment 1

IPv6 Packet 2
- IPv6 Header
- Fragment Header
- Fragment 2

IPv6 Packet 3
- IPv6 Header
- Fragment Header
- Fragment 3

Host2
2001:db8:2::2
On the Destination Node (Host2)

IPv6 Packet 1
- IPv6 Header
- Fragment Header
- TCP Header
- Fragment 1

IPv6 Packet 2
- IPv6 Header
- Fragment Header
- Fragment 2

IPv6 Packet 3
- IPv6 Header
- Fragment Header
- Fragment 3

Reassemble the fragments to be the original packet.
On the Destination Node (Host2)

IPv6 Packet 1
- IPv6 Header
- Fragment Header
- TCP Header
- Fragment 1

IPv6 Packet 2
- IPv6 Header
- Fragment Header
- Fragment 2

IPv6 Packet 3
- IPv6 Header
- Fragment Header
- Fragment 3

IPv6 Packet
- IPv6 Header

Reassemble the fragments to be the original packet.
On the Destination Node (Host2)

Reassemble the fragments to be the original packet.
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
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.

Packet 1
size=1500 bytes
Path MTU Discovery

Host1: MTU cache=1500

Link MTU values are marked on each link.

Captured packets are available:
https://www.cloudshark.org/captures/7dd0b50eb768
Path MTU Discovery

Link MTU values are marked on each link.

Packet 1
size=1500bytes
Path MTU Discovery

Link MTU values are marked on each link.

Packet 1
size=1500 bytes

Because 1300 < 1500, the packet 1 cannot be transmitted.
Path MTU Discovery

Link MTU values are marked on each link.

Packet 1
size=1500 bytes
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.

Packet 1
size=1500 bytes
Drop!

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

ICMP Error:
Packet size too big!
MTU = 1300

Host1 Update:
MTU cache=1300
Path MTU Discovery

Frame 3: 1294 bytes on wire (10352 bits), 1294 bytes captured (10352 bits)
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 Control Message Protocol v6, 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

Host1: MTU cache=1300

Link MTU values are marked on each link.

Packet 2
size=1300 bytes

2001:db8:1::1
Router 1
Router 2
Router 3
2001:db8:2::2
Path MTU Discovery

Link MTU values are marked on each link.

Host1: MTU cache=1300

Packet 2
size=1300 bytes

---

Frame 4: 1310 bytes on wire (10480 bits), 1310 bytes captured (10480 bits)


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

Link MTU values are marked on each link.

Host1: MTU cache=1300

Packet 2
size=1300 bytes
Path MTU Discovery

Host1: MTU cache=1300

Link MTU values are marked on each link.

Packet 2 size=1300bytes
Link MTU values are marked on each link.

Host1: MTU cache=1300

Packet 2
size=1300bytes
Path MTU Discovery

Link MTU values are marked on each link.

Host1: MTU cache=1300

Packet 2 size=1300bytes

Path MTU = 1300
Questions