

Network Layer (II)

Network Layer General Functions

- Addressing: IP address
 - Error control: rerouting, ICMP (Internet Control Message Protocol)
 - QoS: TOS (Type of Service) field and Differentiated Services
-

ICMP

- Internet Control Message Protocol

- used by hosts & routers to communicate network-level information
- typically used for diagnostic/control purpose/generated in response to errors in IP
- Runs on top of IP but still within the network layer
- **Examples:**
 - ping (**Echo Request/Reply**)
 - uses ICMP to send requests to the destination and replies to the source
 - traceroute (**Time-to-live Exceeded, Destination Unreachable**)
 - every node on the path sends three responses to measure the RTT using ICMP or to tell the destination is unreachable.
 - “TTL is 0” (**Packet Drop**)
 - a router uses ICMP packet to indicate the source of a packet that the lifetime exceeded and the packet is dropped.
- ICMP messages use its **type** and **code** fields to specify its purposes. These values are well defined in the protocol. The list below shows a subset of the values as an example.

<u>Type</u>	<u>Code</u>	<u>description</u>
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery

11	0	TTL expired
12	0	bad IP header

Ping

- ping 137.146.213.6 and open Wireshark filter on ip.src == 137.146.103.160 && icmp

```

mbp-190250:~ yingli$ ping 137.146.213.6
PING 137.146.213.6 (137.146.213.6): 56 data bytes
64 bytes from 137.146.213.6: icmp_seq=0 ttl=62 time=81.929 ms
64 bytes from 137.146.213.6: icmp_seq=1 ttl=62 time=67.054 ms
64 bytes from 137.146.213.6: icmp_seq=2 ttl=62 time=79.090 ms
64 bytes from 137.146.213.6: icmp_seq=3 ttl=62 time=63.663 ms
64 bytes from 137.146.213.6: icmp_seq=4 ttl=62 time=60.488 ms
64 bytes from 137.146.213.6: icmp_seq=5 ttl=62 time=152.268 ms
64 bytes from 137.146.213.6: icmp_seq=6 ttl=62 time=64.357 ms
64 bytes from 137.146.213.6: icmp_seq=7 ttl=62 time=71.504 ms
64 bytes from 137.146.213.6: icmp_seq=8 ttl=62 time=77.126 ms
^C
--- 137.146.213.6 ping statistics ---
9 packets transmitted, 9 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 60.488/79.720/152.268/26.598 ms
mbp-190250:~ yingli$
  
```

- ping is a network tool to check the status of the network
- It uses ICMP and the client-server model
- The client sends an ICMP request (type 8 code 0), and the server sends an ICMP response (type 0 code 0) back to the client after receiving the request.
- The ping program calculates the RRT for each ping packet based on the request and response messages
- After ctrl-c, the ping program calculates the statistics based on the RRT of all ping packets
- Then, we can use the statistics to tell the status of the network. (packet loss and RRT).

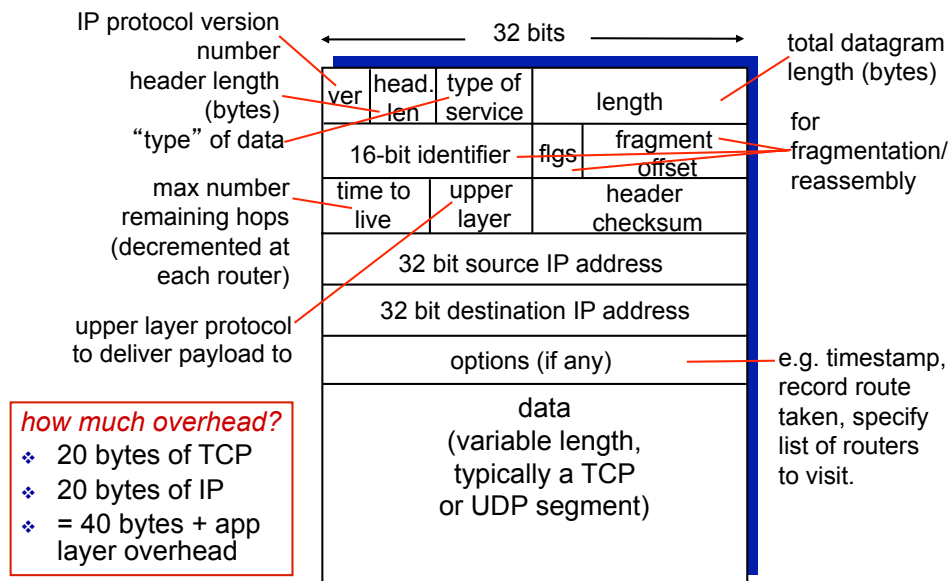
IP Design Goals

- Should have fields for source and destination IP addresses
- Needs means for **error control**:
 - detection of packet **header corruption** (L2 does the heavy lifting)
 - **limiting the lifespan** of a packet (TTL not in TCP)
- **Fragmentation**
 - carrying transport layer messages that are longer than what L2 can support

IPv4 Header

- **version**: v4 or v6; v6 has a different format, so decide **how to parse** the binary sequence
- **TOS**: distinguish different types of IP datagrams.
 - e.g. it might be useful to **distinguish real-time datagrams** (such as those used by an **IP telephony application**) **from non-real-time traffic** (e.g. **FTP**). The specific level of service to be provided is a **policy issue determined** and configured by the network administrator for that router.

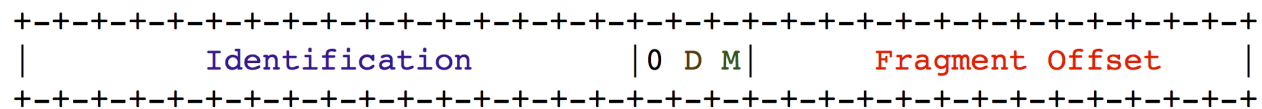
- **upper layer**: used only when an IP datagram reaches its final destination, indicating the transport layer protocol, 6 - TCP, 17 - UDP, 1 - ICMP, so that the destination will know how to parse the data.



Fragmentation

- Problem:
 - network layer needs to deliver a **PDU** (protocol data unit) that is **longer than what the link layer permits**
- Solutions:
 - drop and inform (default behavior of IPv6)
 - **break** the PDU to **smaller units** (fragments) and **reassemble** them **at the destination** (default behavior of IPv4)

IPv4 Fragmentation



- **Identification** (16 bits)
 - identifies the original fragmented packet
 - sender generates it and increments the number for each packet it sends
- **Fragment Offset** (13 bits)
 - specifies the location (in 8-byte blocks) of the fragment in the packet
- **M** (1 bit) - More Fragments
- **D** (1 bit) - Do not Fragments

Fragmentation Example

- Example from Wikipedia: http://en.wikipedia.org/wiki/IPv4#Fragmentation_and_reassembly
- Transport layer Packet length 4,500 bytes, MTU 2,500 bytes:

Fragment	Total bytes	Header bytes	Data bytes	"More fragments" flag	Fragment offset (bytes)
1	2500	20	2480	1	0
2	2040	20	2020	0	310

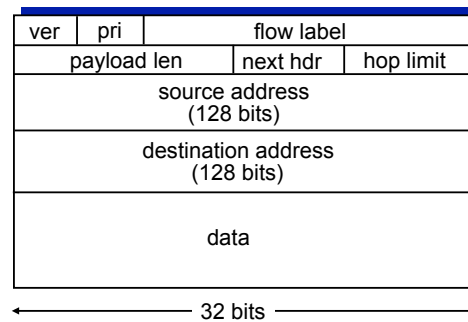
- 0.
- $0 + 2480/8 = 310$
- Please note that the MTU is not static across the entire network.
- It is possible that the above two fragments reach another router connecting a link with smaller MTU. Then, the above two fragments will be further fragmented at that router.
- Suppose the above two fragments reach a router connecting a link with MTU 1,500 bytes. These two fragments will be further fragmented:

Fragment	Total bytes	Header bytes	Data bytes	"More fragments" flag	Fragment offset (bytes)
1	1500	20	1480	1	0
2	1020	20	1000	1	185
3	1500	20	1480	1	310
4	560	20	540	0	495

- 0.
- $0 + 1480/8 = 185$
- $185 + 1000/8 = 310$
- $310 + 1480/8 = 495$
- Keep the good stuff ...
 - unreliable datagram service
 - TTL, TOS (for compatibility)
- Eliminate the unnecessary ...
 - no fragmentation (only as an option)
 - no header checksums
- Address the issues ...
 - longer addresses and more

IPv6 - Protocol Design

IPv6 Header



- version: 4-bit, 6 indicate v6
- **pri**: 8-bit traffic class field, like the TOS field in v4, used to **give priority to certain datagram**
- **flow label**: 20-bit, identify **datagrams in same “flow”**
 - concept of “flow” not well defined: audio and video are flow, or file transfer and email are not; it is possible that traffic carried by a high-priority user (someone paying for better service for their traffic) might be also treated as flow.
- payload len: 16-bit integer, number of bytes in the data part (not including the 40-byte header)
- next header: identify upper layer protocol for data (TCP, UDP)
- Discussions
 - **Fragmentation**: **handle by the source and destination**, if the router receives a too large packet to be forwarded, it **drops** the packet, and **send a “Packet Too Big” ICMPv6 message**.
 - Header checksum: TCP/UDP and link layer protocols have checksum

IPv6 Address Representation

- An IPv6 address is represented by **8 groups** of **16-bit hexadecimal** values separated by colons (:
- Can be abbreviated:
 - **omit leading zeroes** in a 16-bit value
 - **replace one group of consecutive zeros** by a double colon
- Example:
 - 2606:4100:38c0:9::5 vs
 - 2606:4100:38c0:**000**9:**0000**:**0000**:**0000**:**000**5
- ping IPv6: `gloin:~> ping6 fe80::da9e:f3ff:fe41:f985%eno1`

Disclaimer: Notes adapted from the textbook and online resources.