ICS 451: Today's plan

- IP
- addresses
- netmasks
- headers
- fragmentation

Internet Protocol

- IP version 4 or version 6
  - overall operation the same, details different
  - IPv4: 32-bit addresses, IPv6: 128-bit addresses
- best-effort datagram communication

IPv4 addresses

- 32 bits or 4 bytes
- each byte written in decimal: 128.171.224.100
- 127.0.0.1 is 0x7F000001 or 01111111000000000000000000000001
- One address for each interface
  - and for for the loopback interface (127.0.0.1)

IPv4 addresses and routing

- 32 bits means 4Gi=4,294,967,296 addresses
- if assigned at random, routers would need to have 4Gi routing table entries
  - and routing information for 4Gi routes would have to be distributed to each router
- to summarize information, adjacent addresses are grouped into *networks*
  - so routers only need a route to each network
- The network part of the adress is the first *n* bits
  - *n* varies

Netmasks

- each packet carries an IP destination address
- routing protocols exchange IP destination addresses together with a *netmask* that indicates the value of *n* for that destination
- two representations for the netmask:
  - actual bits, such as 255.255.0.0 or 255.255.192.0
    - the first *n* bits are 1, the remainder are 0
  - number of bits, such as /16 or /18
- first representation used in the computer

Network Sizes

- with netmasks, all networks have sizes that are powers of 2
  - but two addresses are reserved:
    - all 0's is the network number
    - all 1's is the network broadcast address
- every network with at least two addresses can be split into smaller subnetworks
  - each will have a size a power of two

Routing Table Lookup

- each packet has an IP destination address A
- given an entry with destination network D and network mask M, the packet matches the entry if

  D & M == A & M

- that is, if the first *n* bits of the destination matches the first *n* bits of the address

Multiple Matches

- there may be more than one routing table entry that matches a destination IP address
- if so, the one with the longest mask is used
- this lets us have generic routes, with shorter masks, and specific routes, with longer masks
- a route with a 0-bit netmask is a default route
  - used only if nothing else matches
- a route with a 32-bit (128-bit) netmask is a *host route*, always used if it matches

Address Classes

- Once upon a time, there were many more IP addresses than hosts
- back then, netmasks were only used internally
- addresses beginning with a 0 bit (first byte 0..127) were in class A and had 8-bit netmasks
- addresses with 10... (first byte 128..191) were in class B, and had 16-bit netmasks
- addresses with 110... (192..223) were in class C, and had 24-bit netmasks
- 1110... (224..239): class D multicast addresses

Reasons for using CIDR

- Classless InterDomain Routing
  - the alternative to address classes
  - requires address masks
    - changes to routing protocols, but not IP itself
- fixed address classes are too inflexible
  - an organization might need part of a class B
  - or multiple class C addresses
- so everyone requested class B addresses (e.g. UH, 128.171.0.0/16)
  - 1980's: class B addresses were running out

Reasons for using IPv6

- 2010's: IPv4 addresses were running out
  - e.g. UH, 2607:f278:4101:3d::/64
- 128 bits or 16 bytes
- each two bytes written in hex: 2001:1900:3001:11::2c
- Any single sequence of zeros replaced by ::
- ::1 is 0x00000000 00000000 00000000 00000001, and is the loopback address
- Still one address for each interface
  - must match network to which attached

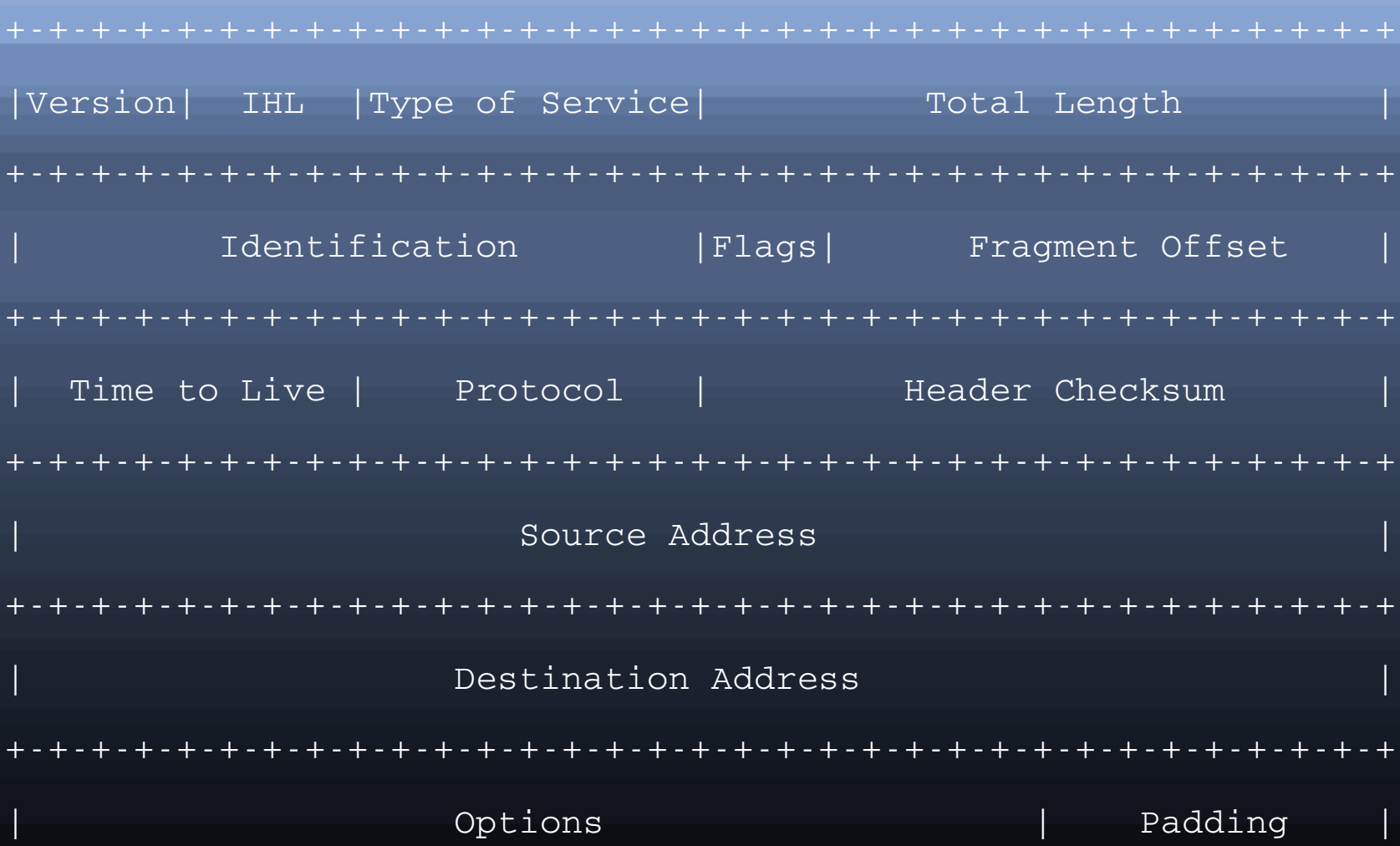
IP headers

- version number (4 or 6)
- source and destination addresses
- Time To Live (IPv4) or Hop Limit (IPv6)
  - number of times the packet may be forwarded
- Protocol (IPv4) or Next Header (IPv6)
  - 1 for ICMP, 6 for TCP, 17 for UDP
- Total Lenght (IPv4) or Payload Length (IPv6)
- Type of Service (IPv4) or Traffic Class (IPv6)

IPv4-only header fields

- Header Length, in 32-bit words
  - 5..15 words, so 20-60 Bytes
- Identification, Flags, and Fragment Offset
  - used by routers to split packets into fragments
  - and by hosts to reassemble fragments into packets
- Header checksum
  - must be recomputed by each router, because the TTL changes

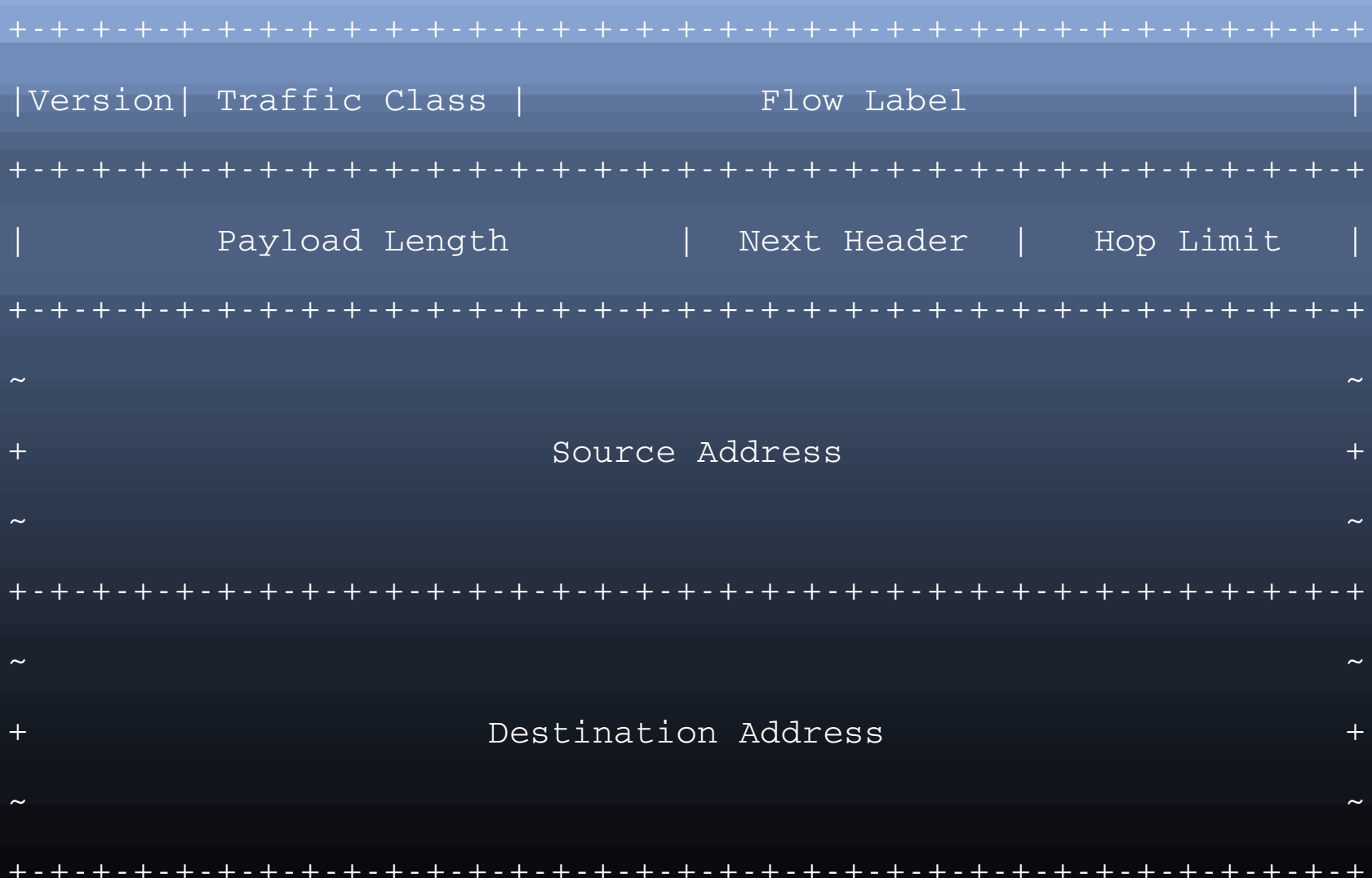
IPv4 header


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

IPv6-only header fields

- Flow Label
  - 20 bits, usage not well defined
- Next Header
  - combines IPv4 “protocol” field and IPv4 “header length” field
  - identifies the next header, whether TCP, UDP, or an IP extension header
- extension headers include hop-by-hop, routing, fragmentation, and destination headers
  - each extension header has a Next Header field

IPv6 header



- or see wikipedia, IPv6, under packet format

IP fragmentation

- when a datagram is larger than the MTU of the outgoing interface, IP fragments it
  - in IPv6, only senders fragment, routers don't
  - routers can only fragment if the Don't Fragment flag is not set
- fragments are IP packets with IP headers
- each fragment carries the same ID
  - hopefully different from IDs of other datagrams that have the same source and destination

IP fragment offset

- the destination needs to reassemble incoming fragments into a whole datagram
- each fragment carries a fragment offset
  - the place where the fragment's payload fits in the reassembled payload
- e.g. fragment offset 144: this payload is to be placed beginning at byte offset 144 in the reassembled payload
- offsets must be a multiple of 8 bytes
  - and the 3 least significant bits are not sent

IP reassembly

- see if already reassembling for this (source,destination,packetID)
  - if not, create an empty reassembled payload
- save the payload at the appropriate offset in the reassembled payload
  - need to keep track of where we have received data for the payload
- if the packet is complete, give to TCP/UDP
  - size given by last fragment
  - given by MoreFragments flag = 0

IPv6 fragmentation

- only done by senders, never by routers
- done in an extension header
- fragment extension header carries 13-bit fragment offset, 32-bit packet ID, and 1-bit More Fragments flag