CS 43: Computer Networks
IP

Kevin Webb
Swarthmore College

November 7, 2017

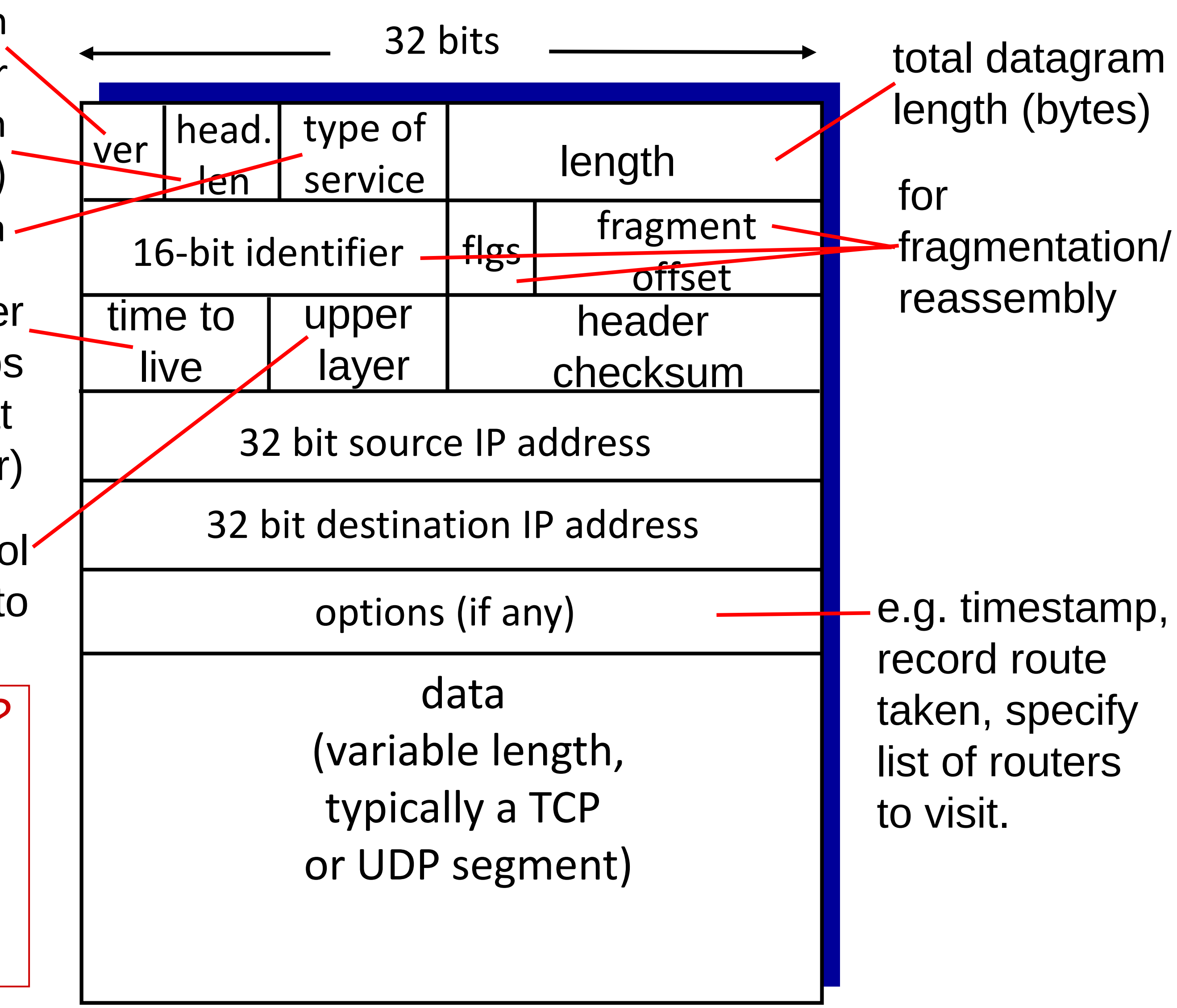
Outline

- IP header format
- Subnets and IP addressing
  - CIDR
  - Route aggregation
- DHCP: Assigning an IP address to an interface
- Fragmentation

Outline

- IP header format
- Subnets and IP addressing
  - CIDR
  - Route aggregation
- DHCP: Assigning an IP address to an interface
- Fragmentation

IP datagram format

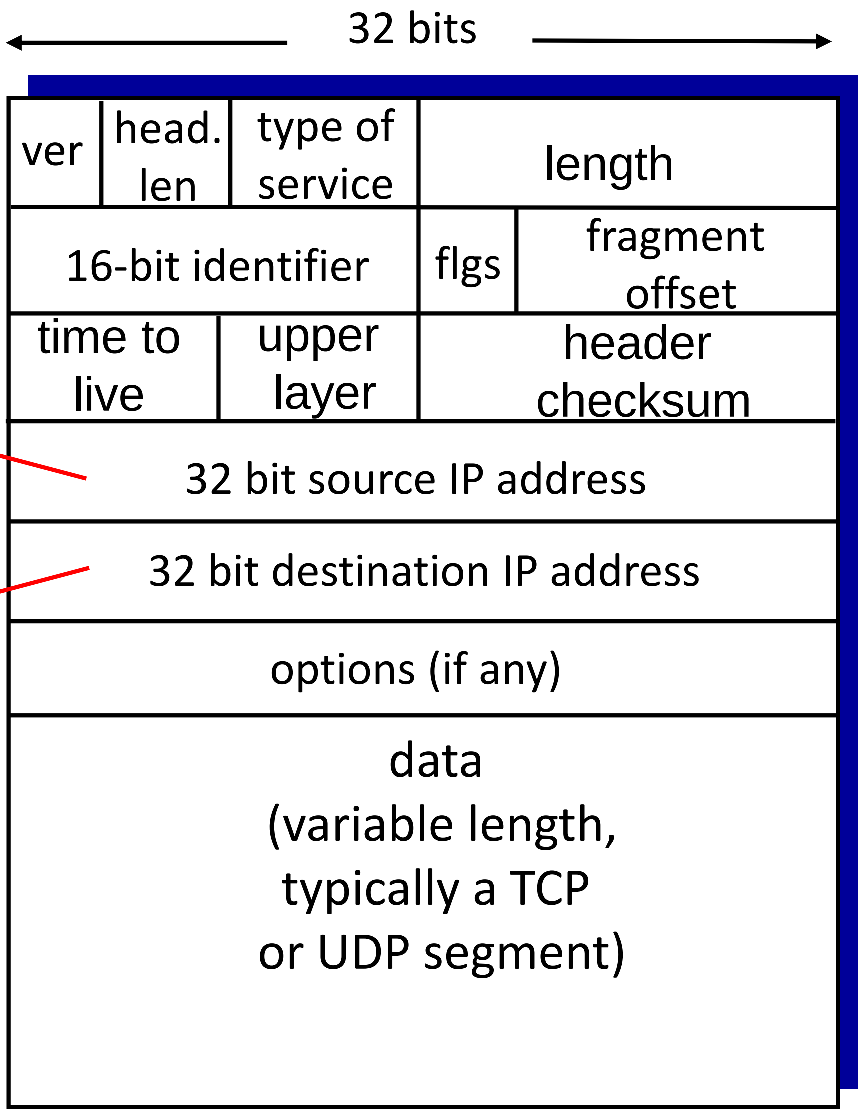


IP protocol version number
header length (in 32-bit words)
"type" of data
max number remaining hops (decremented at each router)
upper layer protocol to deliver payload to
32 bits
ver
head. len
type of service
length
16-bit identifier
flgs
fragment offset
time to live
upper layer
header checksum
32 bit source IP address
32 bit destination IP address
options (if any)
data (variable length, typically a TCP or UDP segment)
total datagram length (bytes)
for fragmentation/ reassembly
e.g. timestamp, record route taken, specify list of routers to visit.

how much overhead?

- 20 bytes of TCP
- 20 bytes of IP
- = 40 bytes + app layer overhead

IP datagram format



Source endpoint.

Final destination endpoint.

Addresses must be unique on the network!

Outline

- IP header format
- Subnets and IP addressing
  - CIDR
  - Route aggregation
- DHCP: Assigning an IP address to an interface
- Fragmentation

IP Addresses

- 32-bit (4-byte) unsigned integer value.
  - Usually written in “dotted decimal” or “dotted quad”
  - E.g., 130.58.68.9 => 10000010 . 00111010 …

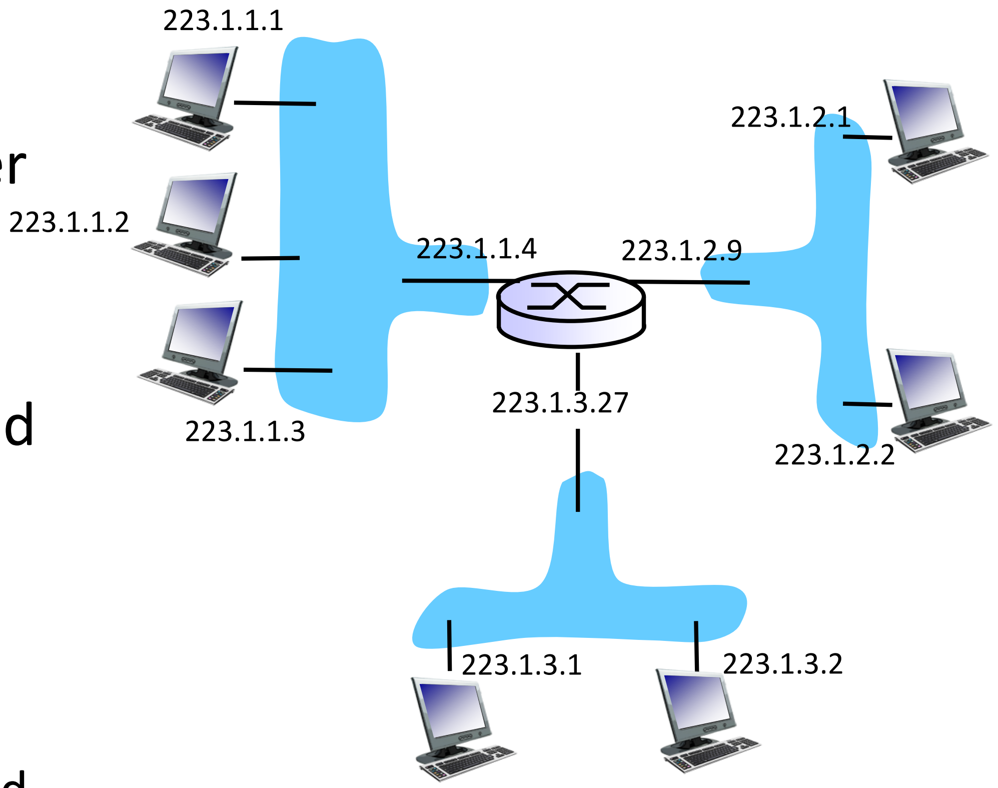
- 2^{32} => 4,294,967,296 possible addresses.

- In the early 80’s, that’s a lot!
  - Population was ~4.5 billion.

- Now…not so much.
  - Population > 7 billion.

Network Interfaces

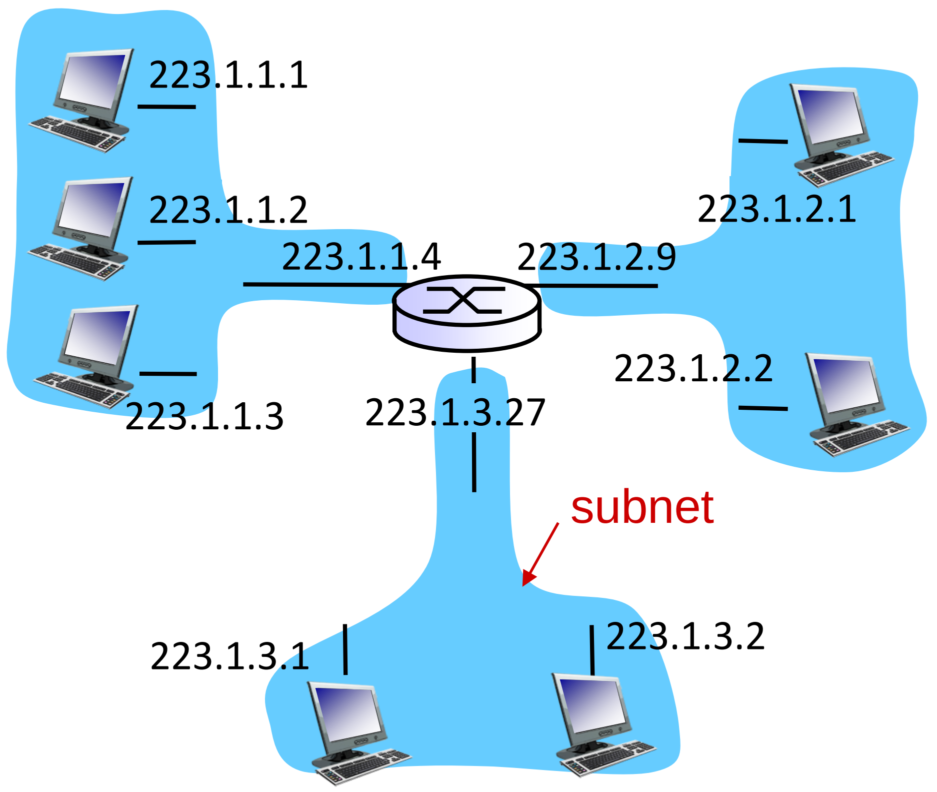
- *IP address:* 32-bit identifier for host, router *interface*
- *interface:* connection between host/router and physical link
  - router's typically have multiple interfaces
  - host typically has one or two interfaces (e.g., wired Ethernet, wireless 802.11)
- *IP addresses associated with each interface*



$$223.1.1.1 = \underbrace{11011111}_{223}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}$$

Subnets

- IP address:
  - subnet part - high order bits
  - host part - low order bits
- *what's a subnet ?*
  - device interfaces with same subnet part of IP address
  - can physically reach each other *without intervening router*
  - On the same link layer

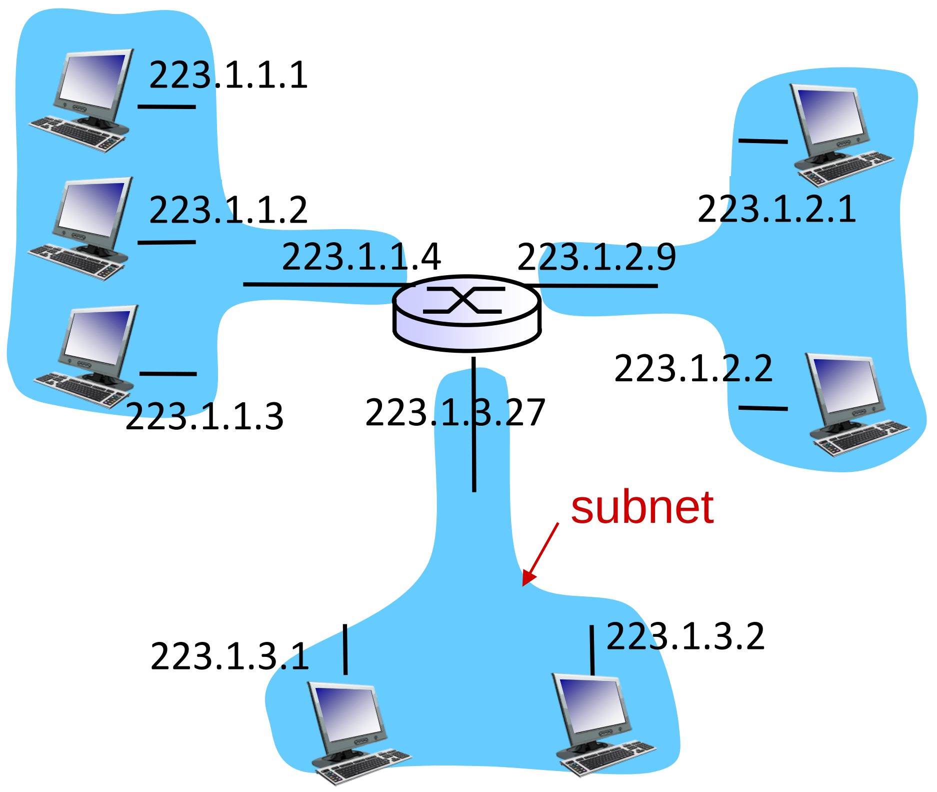


network consisting of 3 subnets

Subnets

Book recipe

- To determine the subnets, detach each interface from its host or router, creating islands of isolated networks
- Each isolated network is called a *subnet*



Assigning Addresses

- IANA – Internet Assigned Numbers Authority
  - (Run by Jon Postel until 1988)
  - Now a part of ICANN

- ICANN: Internet Corporation for Assigned Names and Numbers
  - Manages IP addresses, DNS, resolves disputes

Who gets an address? How many?

- Back in the old days, you called up Jon Postel
  - "How many addresses do you need?"
  - "Here you go! I may have rounded a bit."

- Classful Addressing
  - Class A: 8-bit prefix, 24 bits for hosts (16,777,216)
  - Class B: 16-bit prefix, 16 bits for hosts (65,536)
  - Class C: 24-bit prefix, 8 bits for hosts (256)

CIDR

- Classless Interdomain Routing
  - Prefix (subnet) length is no longer fixed
  - (Can be division of bits rather than just 8/24, 16/16, and 24/8)

Why do we give out addresses in CIDR blocks? How many of these statements are true? (Which ones?)

- It requires fewer resources at routers.
- It requires fewer resources at end hosts.
- It reduces the number of block allocations that need to be managed.
- It better utilizes the IP address space.

A – 0, B – 1, C – 2, D – 3, E – 4

CIDR

- Classless Interdomain Routing
  - Prefix (subnet) length is no longer fixed
  - Address blocks come with a **subnet mask**

- Subnet mask written in two ways:
  - Dotted decimal: 255.255.240.0
  - /20
  - Both mean:
    11111111   11111111   11110000   00000000

CIDR

- Addresses divided into two pieces:
  - Prefix portion (network address)
  - Host portion

- Given an IP address and mask,
  we can determine:
  - The prefix (network address) by ANDing
  - The broadcast address by ORing inverted mask

Network Address (Subnet Address)

- E.g., 230.8.1.3/18

11100110	**00001000**	**00000001**	**00000011**
11111111	**11111111**	**11000000**	**00000000**

Network Address (Subnet Address)

- E.g., 230.8.1.3/18

11100110	**00001000**	**00000001**	**00000011**
11111111	**11111111**	**11000000**	**00000000**
11100110	**00001000**	**00000000**	**00000000**

Network address: 230.8.0.0

Broadcast Address

- E.g., 230.8.1.3/18

11100110	**00001000**	**00000001**	**00000011**
~~11111111~~	**~~11111111~~**	**~~11000000~~**	**~~00000000~~**
00000000	**00000000**	**00111111**	**11111111**

Broadcast Address

- E.g., 230.8.1.3/18

11100110	**00001000**	**00000001**	**00000011**
00000000	**00000000**	**00111111**	**11111111**

Broadcast Address

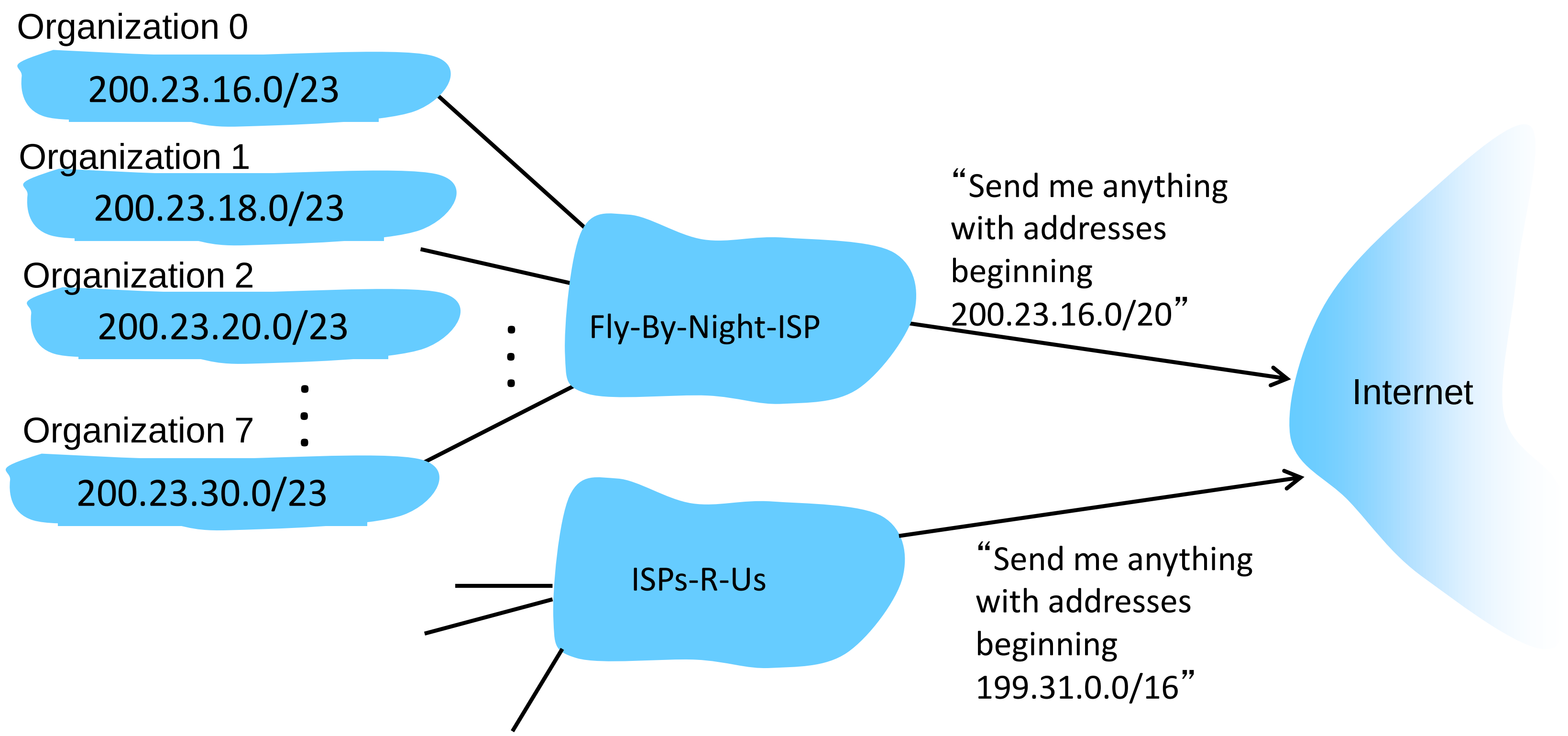
- E.g., 230.8.1.3/18

11100110	**00001000**	**00000001**	**00000011**
00000000	**00000000**	**00111111**	**11111111**
11100110	**00001000**	**00111111**	**11111111**

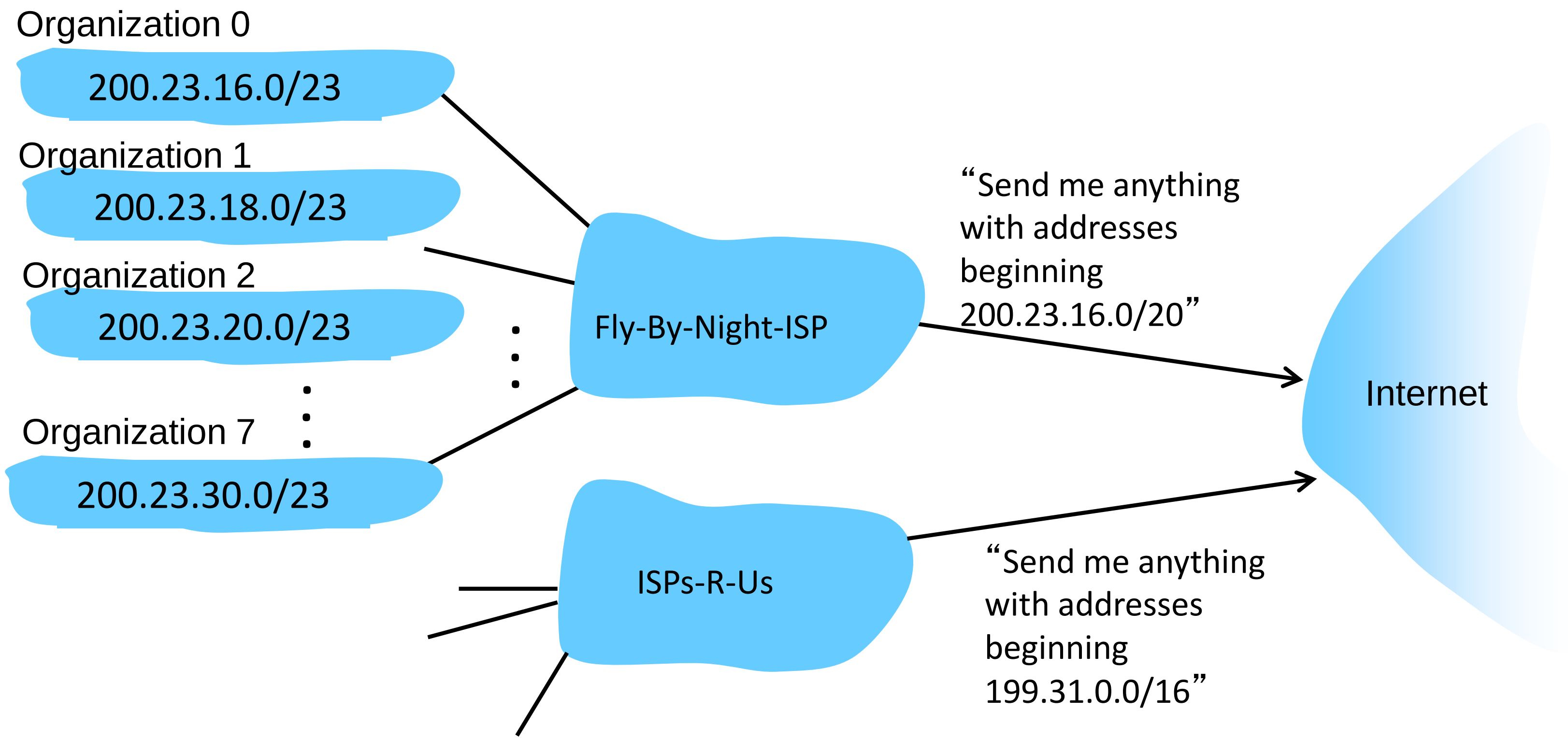
Broadcast address: 230.8.63.255

Hierarchical Addressing: Route Aggregation

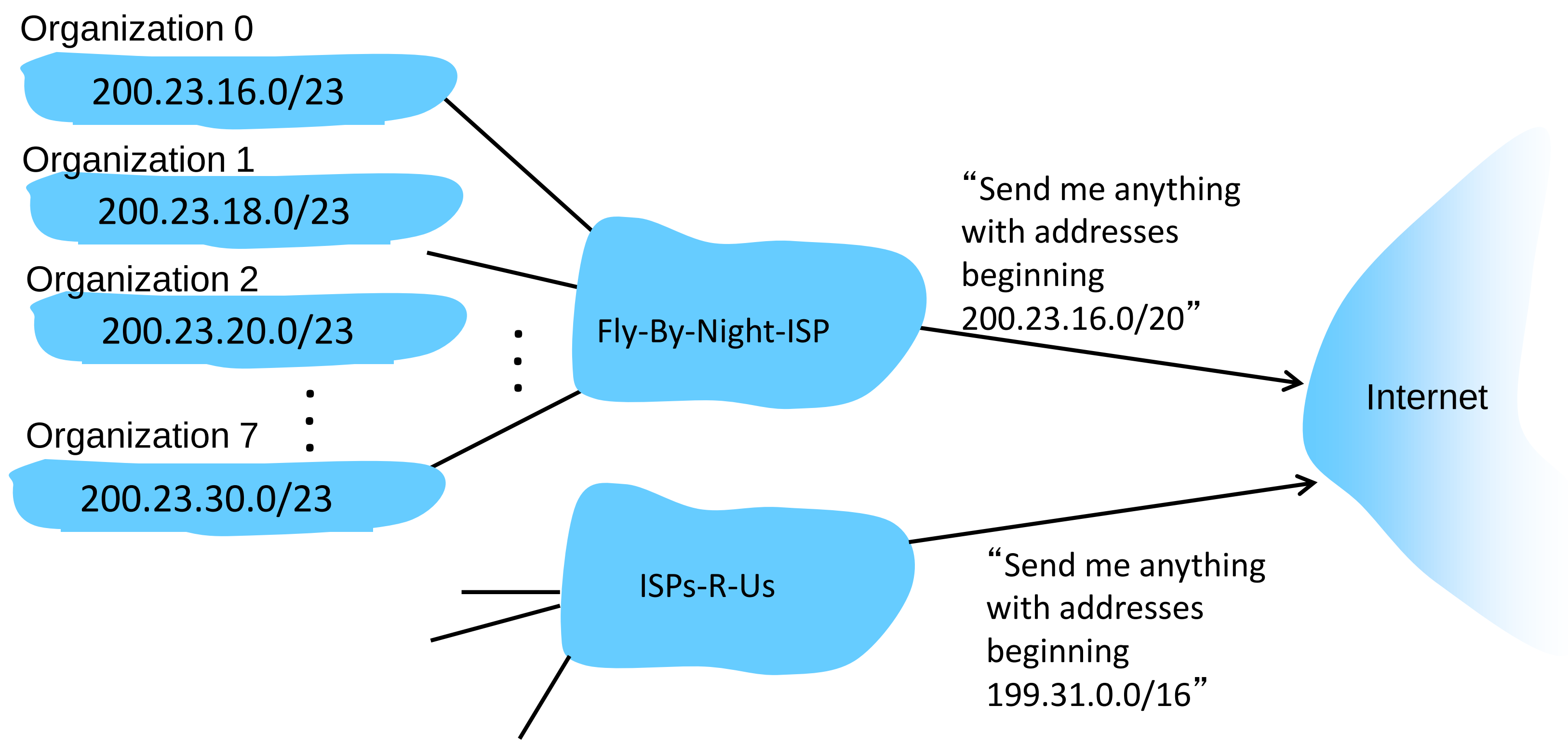
Hierarchical addressing allows efficient advertisement of routing information:



What should we do if organization 1 decides to switch to ISPs-R-Us?



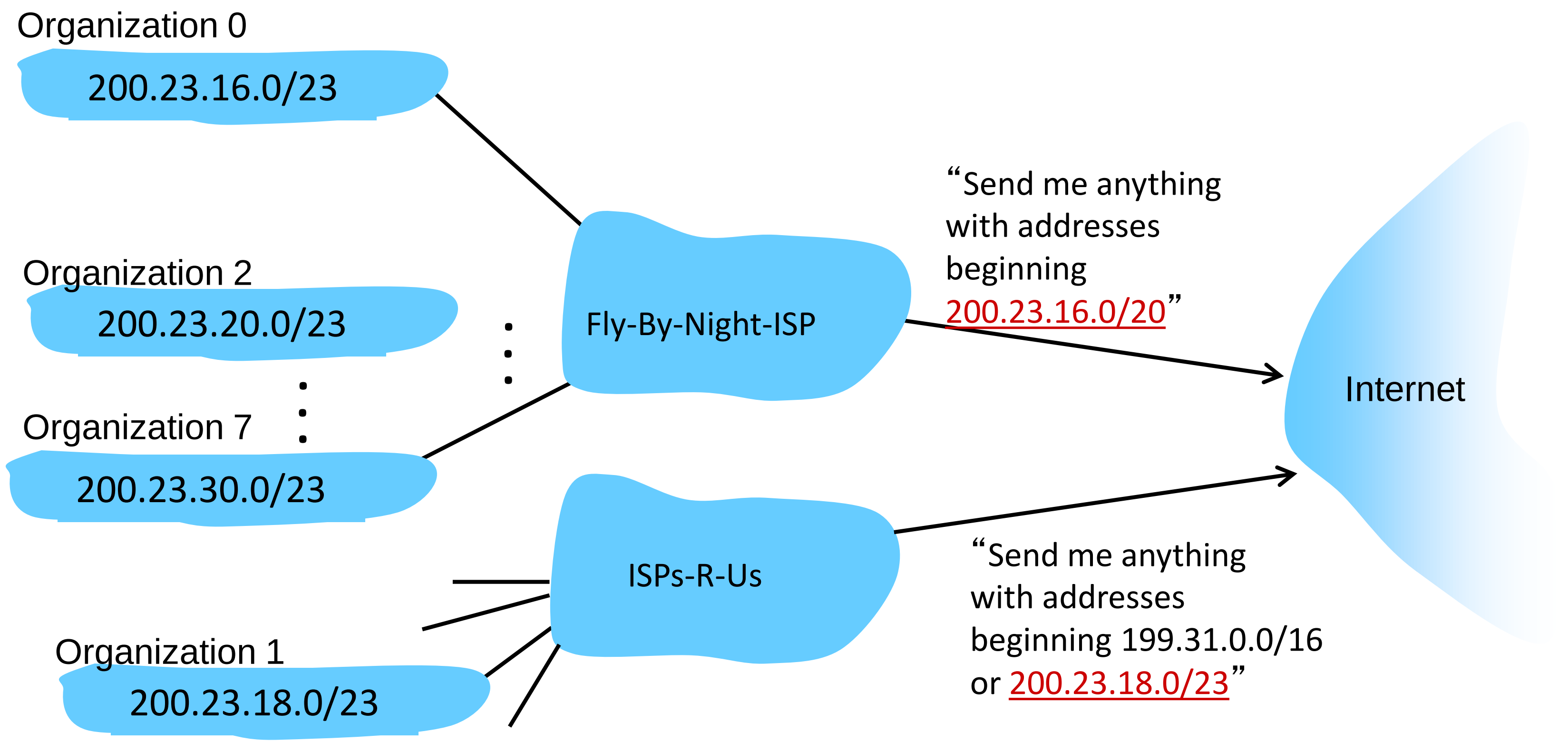
What should we do if organization 1 decides to switch to ISPs-R-Us?



A. Move 200.23.18.0/23 to ISPs-R-Us (and break up Fly-By-Night's /20 block).
B. Give new addresses to Organization 1 (and force them to change all their addresses).
C. Some other solution.

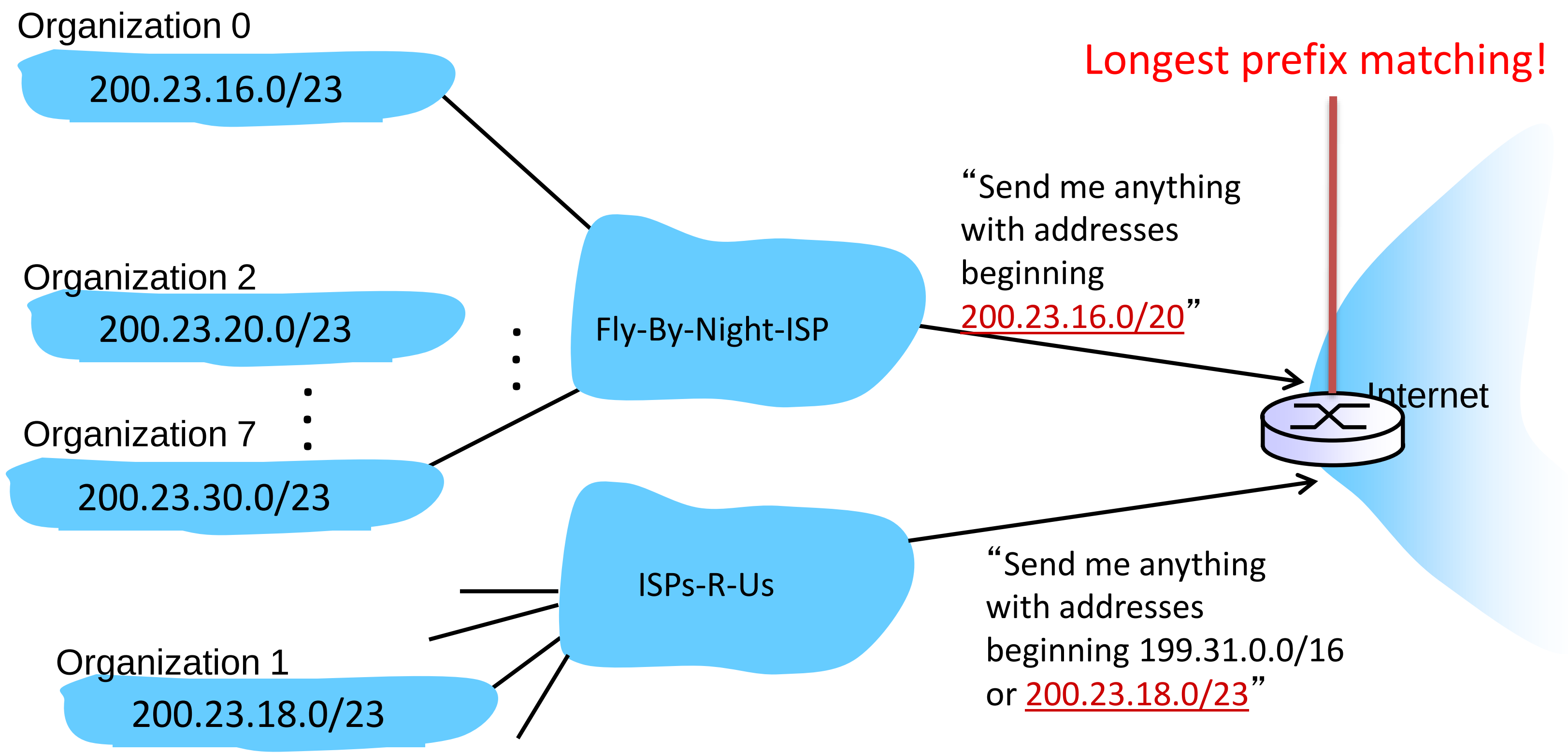
Hierarchical addressing: More Specific Routes

ISPs-R-Us has a more specific route to Organization 1



Hierarchical addressing: More Specific Routes

ISPs-R-Us has a more specific route to Organization 1



Outline

- IP header format
- Subnets and IP addressing
  - CIDR
  - Route aggregation
- DHCP: Assigning an IP address to an interface
- Fragmentation

How does an end host get an IP address?

- Static IP: hard-coded
  - Windows: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config

- DHCP: Dynamic Host Configuration Protocol: dynamically get address from as server
  - “plug-and-play”

DHCP: Dynamic Host Configuration Protocol

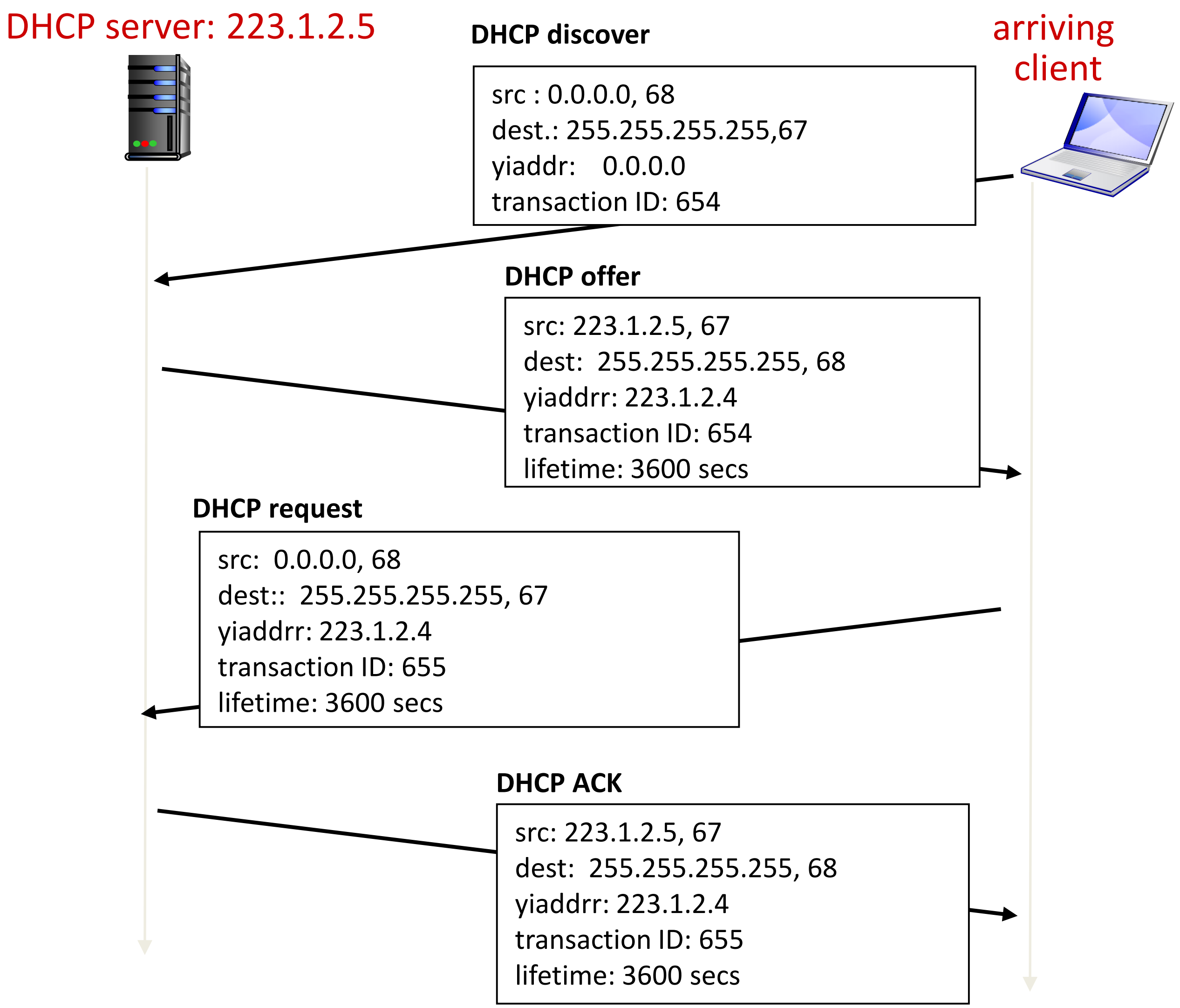
Goal: allow host to *dynamically* obtain its IP address from network server when it joins network

- can renew its lease on address in use
- allows reuse of addresses
- support for mobile users who want to join network

DHCP overview:

- host broadcasts "DHCP discover" msg [optional]
- DHCP server responds with "DHCP offer" msg [optional]
- host requests IP address: "DHCP request" msg
- DHCP server sends address: "DHCP ack" msg

DHCP client-server scenario



src : 0.0.0.0, 68
dest.: 255.255.255.255,67
yiaddr: 0.0.0.0
transaction ID: 654

src: 223.1.2.5, 67
dest: 255.255.255.255, 68
yiaddrr: 223.1.2.4
transaction ID: 654
lifetime: 3600 secs

src: 0.0.0.0, 68
dest:: 255.255.255.255, 67
yiaddrr: 223.1.2.4
transaction ID: 655
lifetime: 3600 secs

src: 223.1.2.5, 67
dest: 255.255.255.255, 68
yiaddrr: 223.1.2.4
transaction ID: 655
lifetime: 3600 secs

DHCP: More than IP Addresses

DHCP can return more than just allocated IP address on subnet:

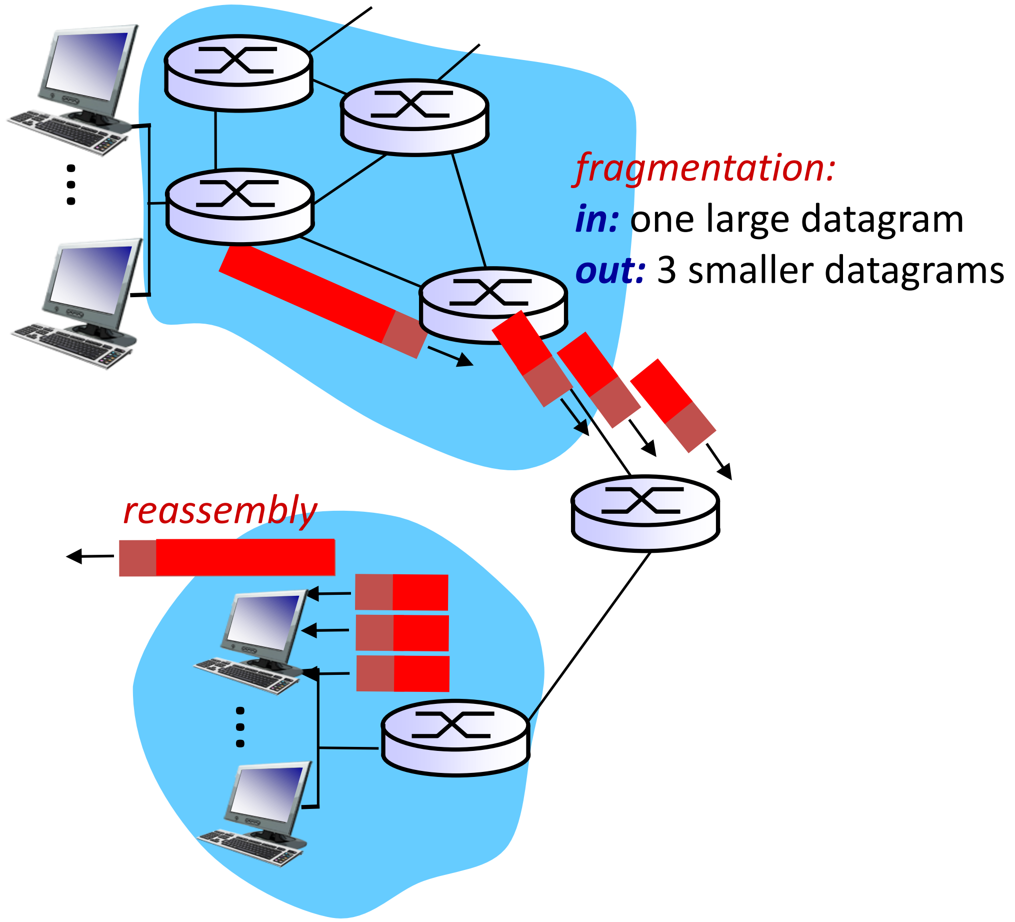
- address of first-hop router for client (default GW)
- name and IP address of DNS server(s)
- subnet mask

Outline

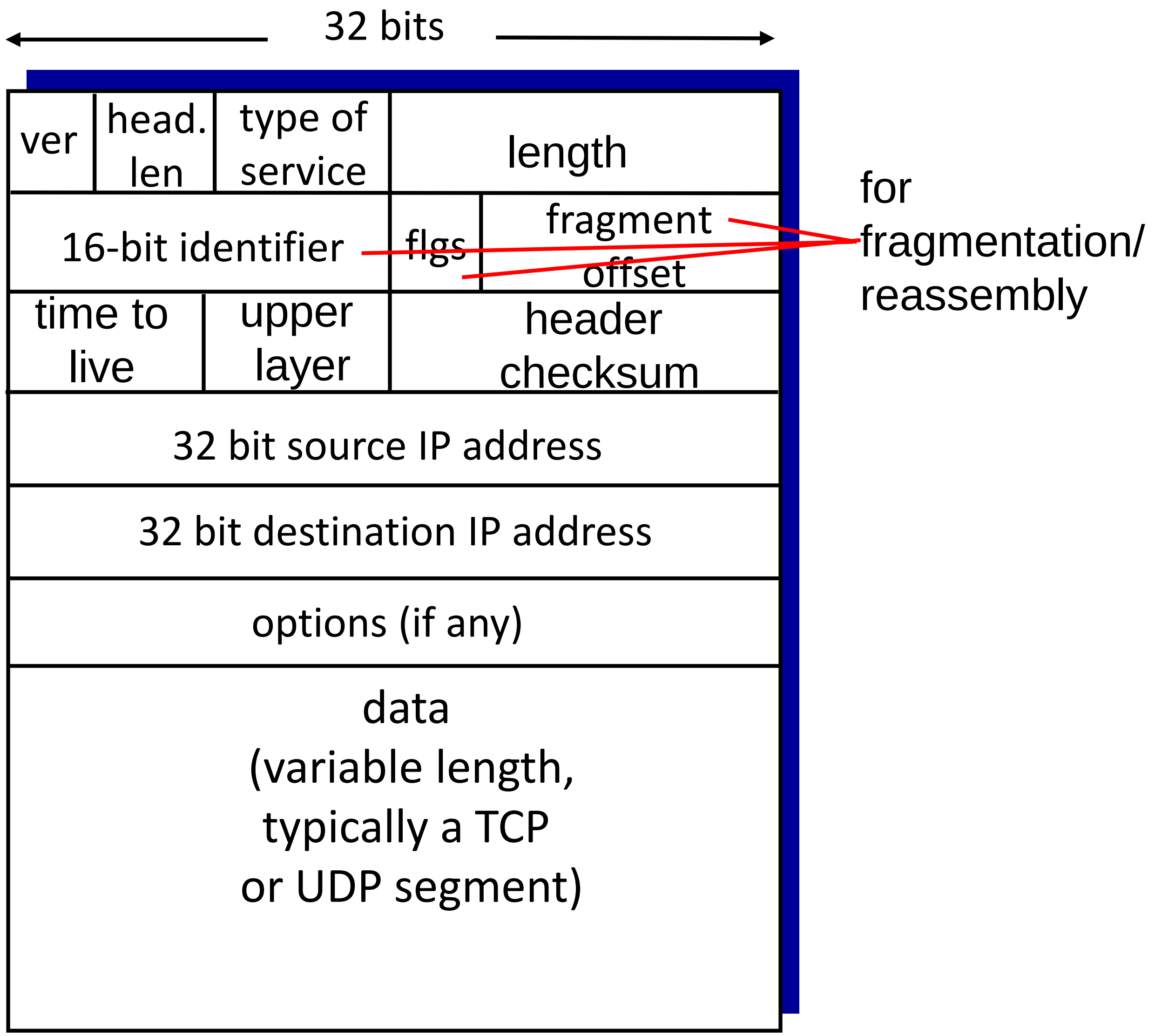
- IP header format
- Subnets and IP addressing
  - CIDR
  - Route aggregation
- DHCP: Assigning an IP address to an interface
- Fragmentation

IP Fragmentation, Reassembly

- Network links have MTU (max transfer size) - largest possible link-level frame
  - Different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - One datagram becomes several datagrams
  - Reassembled only at final destination
  - IP header bits used to identify, order related fragments



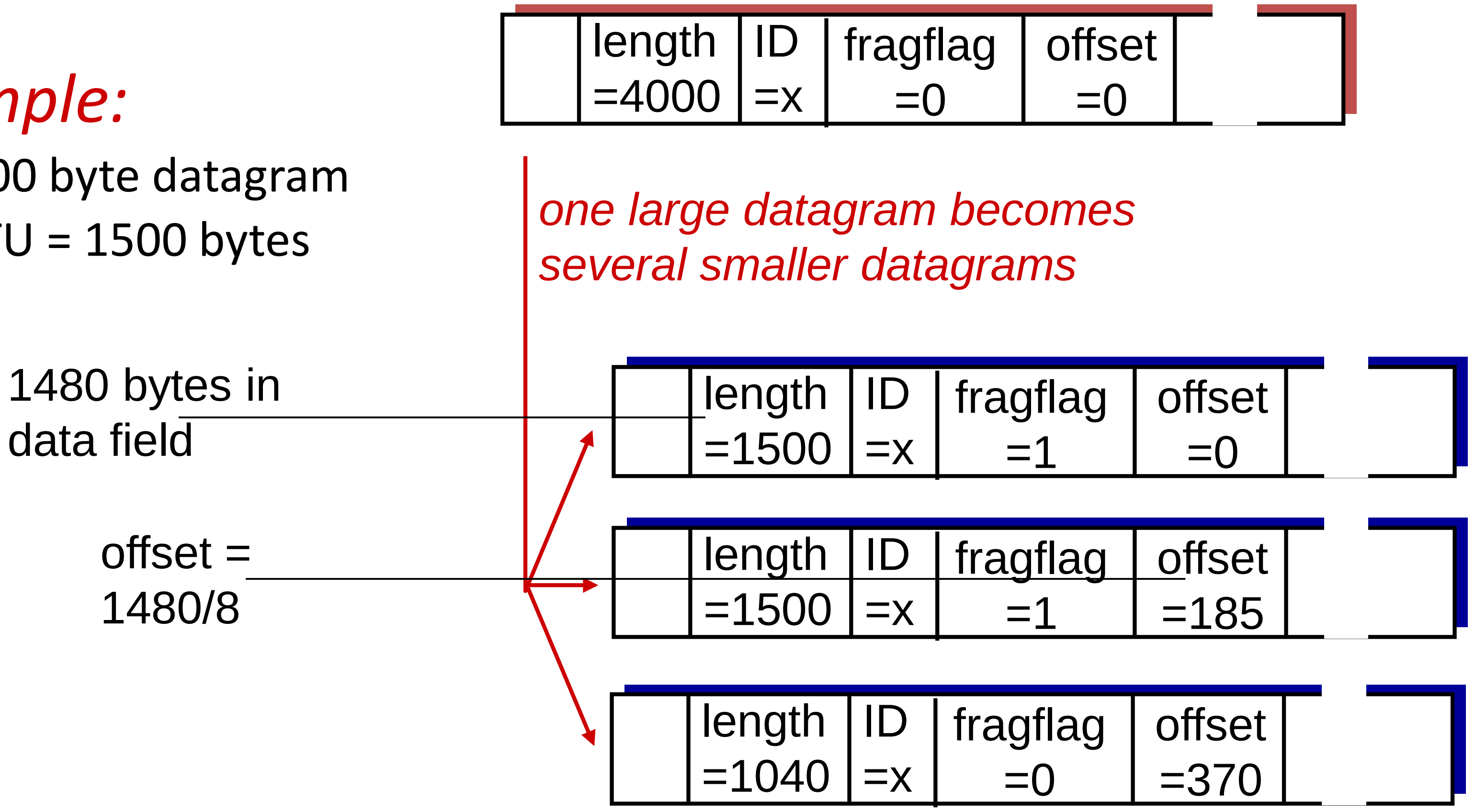
IP datagram format



IP Fragmentation, Reassembly

Example:

- 4000 byte datagram
- MTU = 1500 bytes



How can we use this for evil?

A. Send fragments that overlap.

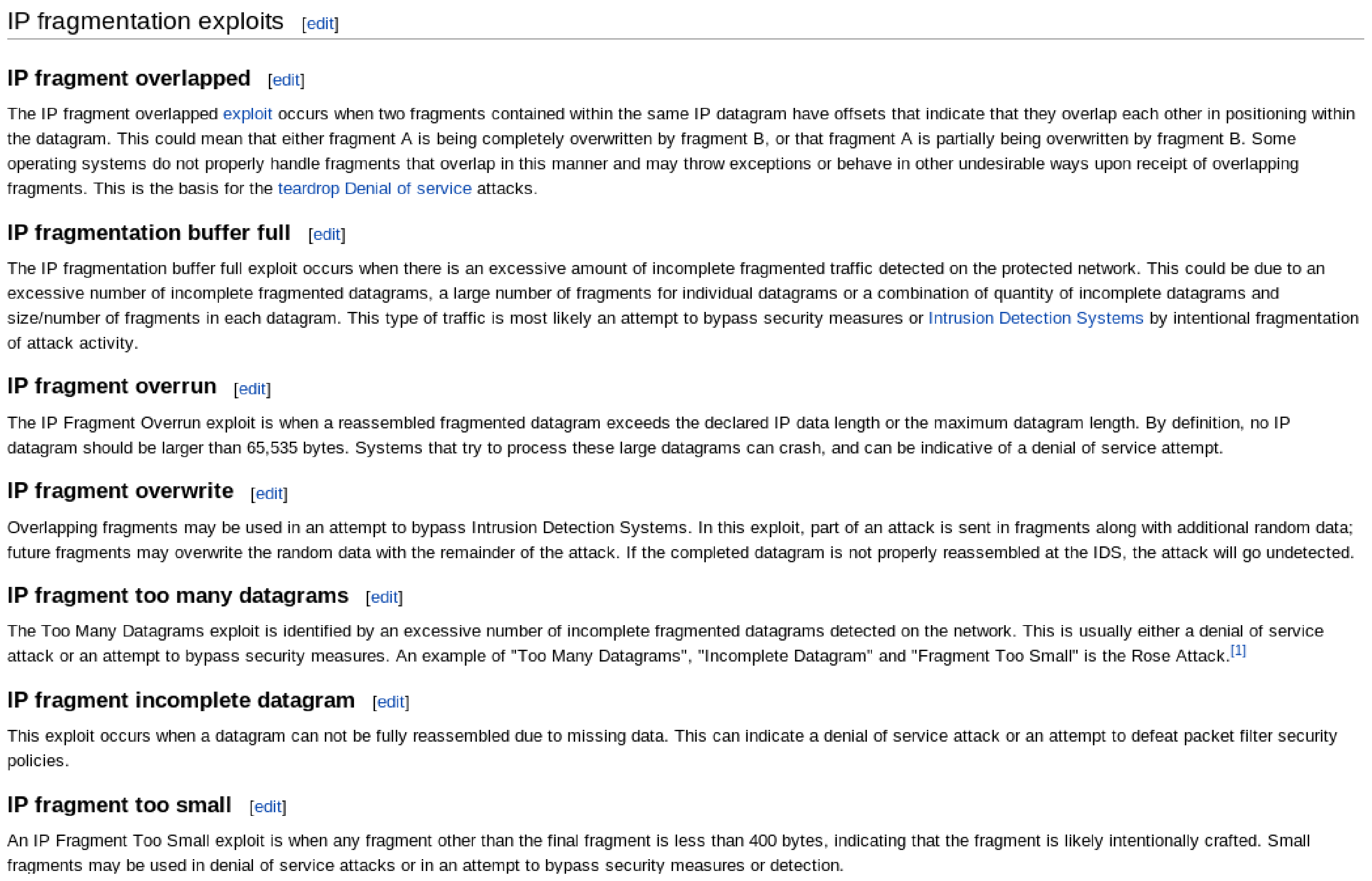
B. Send many tiny fragments, none of which have offset 0.

C. Send fragments that when assembled, are bigger than the maximum IP datagram.

D. More than one of the above.

E. Nah, networks (and operating systems) are too robust for this to cause problems.

IP Fragmentation Attacks…



IP fragmentation exploits [edit]

IP fragment overlapped [edit]

The IP fragment overlapped exploit occurs when two fragments contained within the same IP datagram have offsets that indicate that they overlap each other in positioning within the datagram. This could mean that either fragment A is being completely overwritten by fragment B, or that fragment A is partially being overwritten by fragment B. Some operating systems do not properly handle fragments that overlap in this manner and may throw exceptions or behave in other undesirable ways upon receipt of overlapping fragments. This is the basis for the teardrop Denial of service attacks.

IP fragmentation buffer full [edit]

The IP fragmentation buffer full exploit occurs when there is an excessive amount of incomplete fragmented traffic detected on the protected network. This could be due to an excessive number of incomplete fragmented datagrams, a large number of fragments for individual datagrams or a combination of quantity of incomplete datagrams and size/number of fragments in each datagram. This type of traffic is most likely an attempt to bypass security measures or Intrusion Detection Systems by intentional fragmentation of attack activity.

IP fragment overrun [edit]

The IP Fragment Overrun exploit is when a reassembled fragmented datagram exceeds the declared IP data length or the maximum datagram length. By definition, no IP datagram should be larger than 65,535 bytes. Systems that try to process these large datagrams can crash, and can be indicative of a denial of service attempt.

IP fragment overwrite [edit]

Overlapping fragments may be used in an attempt to bypass Intrusion Detection Systems. In this exploit, part of an attack is sent in fragments along with additional random data; future fragments may overwrite the random data with the remainder of the attack. If the completed datagram is not properly reassembled at the IDS, the attack will go undetected.

IP fragment too many datagrams [edit]

The Too Many Datagrams exploit is identified by an excessive number of incomplete fragmented datagrams detected on the network. This is usually either a denial of service attack or an attempt to bypass security measures. An example of "Too Many Datagrams", "Incomplete Datagram" and "Fragment Too Small" is the Rose Attack.[1]

IP fragment incomplete datagram [edit]

This exploit occurs when a datagram can not be fully reassembled due to missing data. This can indicate a denial of service attack or an attempt to defeat packet filter security policies.

IP fragment too small [edit]

An IP Fragment Too Small exploit is when any fragment other than the final fragment is less than 400 bytes, indicating that the fragment is likely intentionally crafted. Small fragments may be used in denial of service attacks or in an attempt to bypass security measures or detection.

Summary

- 2^{32} addresses is not that many…
- CIDR helps give out finer granularity
  - Divide bits among **network** and **host**
  - Longest prefix matching allows blocks to be divded
- IP supports fragmentation – usually bad news
  - These days, most links have common MTU