Network Layer - Addressing

Dr. T. Venkatesh
Dept of CSE, IIT Guwahati

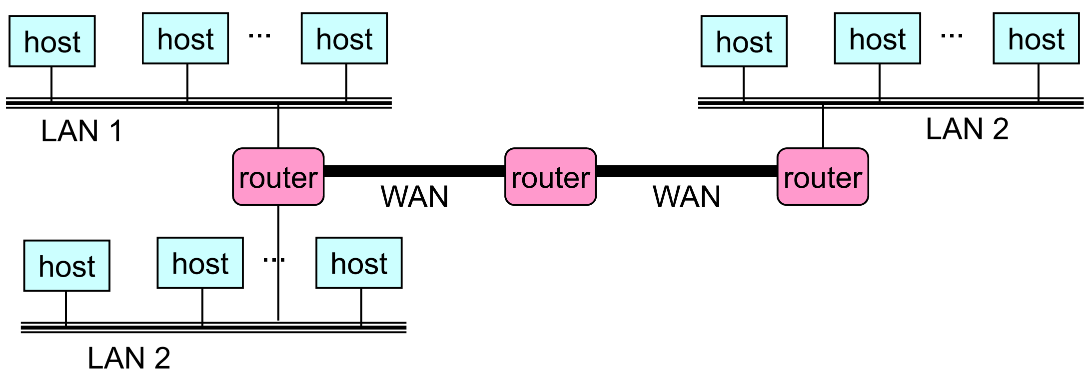
Slides from lectures of Prof. Srini Seshan, CMU and Prof. Jim Kurose, UMass, Amherst

IP Addressing - Outline

- IP addresses
  - Dotted-quad notation
  - Requirements of Addressing
  - Address allocation
  - IP prefixes for aggregation
  - Classful addresses
  - Classless InterDomain Routing (CIDR)
  - Special-purpose address blocks
  - Forwarding using IP address
  - Insufficiency of Addresses

Addressing Hosts in the Internet

- The Internet is an "inter-network"
  - Used to connect *networks* together, not *hosts*
  - Needs a way to address a network (i.e., group of hosts)
  - Address for each interface connected to a network



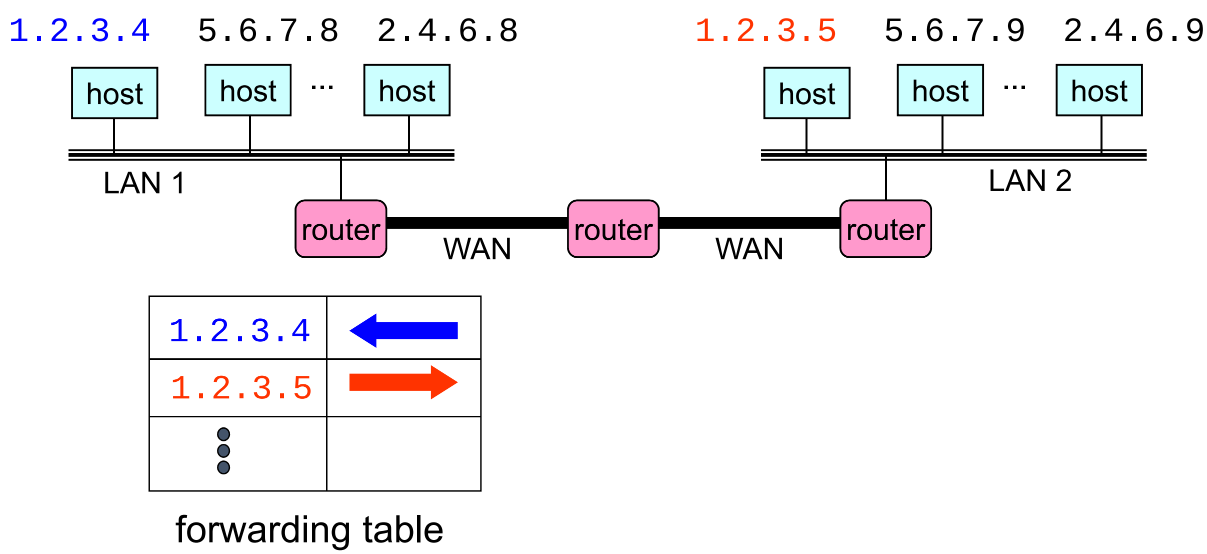
IP Addresses (IPv4)

- A unique 32-bit number
- Identifies an interface (on a host, on a router, …)
- Represented in dotted-decimal notation. E.g, **12.34.158.5**:

12	34	158	5
00001100	00100010	10011110	00000101

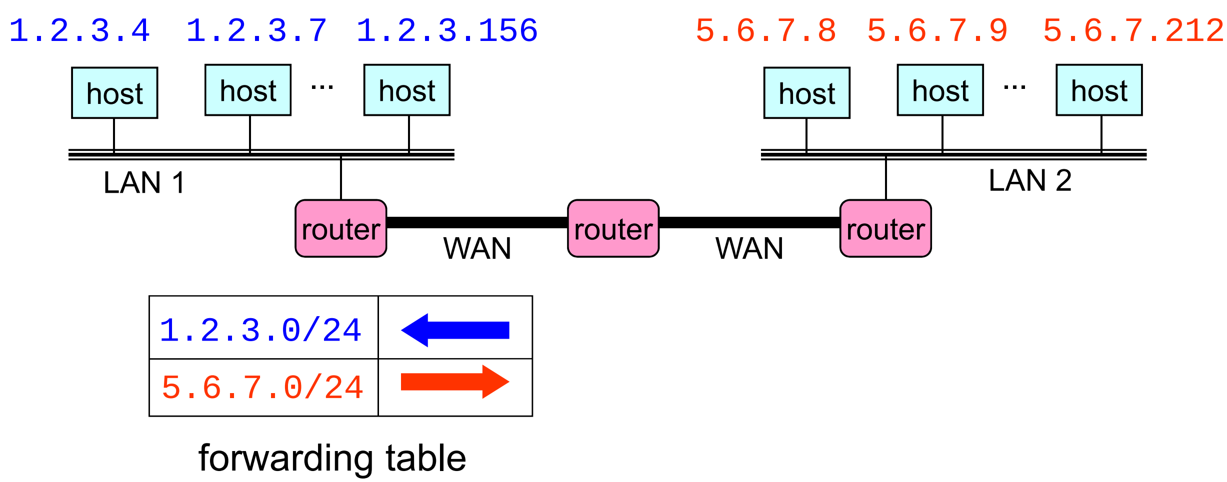
Addressing - Scalability Challenge

- Suppose hosts had arbitrary addresses
  - Then every router would need a lot of information
  - …to know how to direct packets toward the host



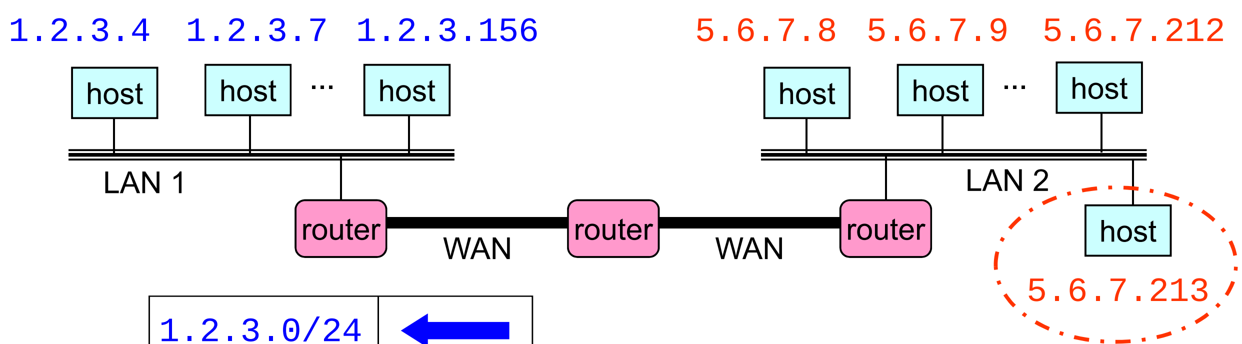
Scalability Improved

- Number related hosts from a common subnet
  - 1.2.3.0/24 on the left LAN
  - 5.6.7.0/24 on the right LAN



Easy to Add New Hosts

- No need to update the routers
  - E.g., adding a new host 5.6.7.213 on the right
  - Doesn't require adding a new forwarding entry



1.2.3.0/24	←
5.6.7.0/24	→

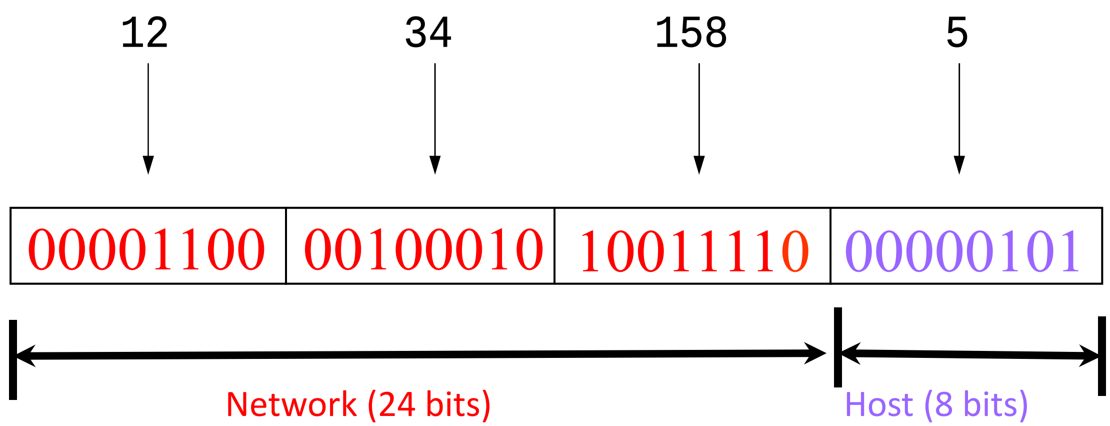
forwarding table

Important Concepts

- Hierarchical addressing critical for scalable system
  - Don’t require everyone to know everyone else
  - Reduces number of updates when something changes
  - Interaction with routing tables

Hierarchical Addressing: IP Prefixes

- Divided into network (left) & host portions (right)
- 12.34.158.0/24 is a 24-bit prefix with 2^9 addresses
  - Terminology: “Slash 24”



IP Address and a 24-bit Subnet Mask

Address

12	34	158	5
00001100	00100010	10011110	00000101
11111111	11111111	11111111	00000000
255	255	255	0

Mask

Obtaining a Block of Addresses

- Separation of control
  - Prefix: assigned *to* an institution
  - Addresses: assigned *by* the institution to their nodes
- Who assigns prefixes?
  - Internet Corporation for Assigned Names and Numbers
    - Allocates large address blocks to *Regional Internet Registries*
  - Regional Internet Registries (RIRs)
    - E.g., APNIC (Asia-Pacific Network Information Centre)
    - Allocates address blocks within their regions
    - Allocated to Internet Service Providers and large institutions
  - Internet Service Providers (ISPs)
    - Allocate address blocks to their customers (could be recursive)

Classful Addressing

- Class A: if first byte in [0..127], assume /8 (top bit = 0)

0*******	********	********	********

  - Very large blocks (e.g., MIT has 18.0.0.0/8)
- Class B: first byte in [128..191] ⇒ assume /16 (top bits = 10)

10******	********	********	********

  - Large blocks (e.g,. UCB has 128.32.0.0/16)
- Class C: [192..223] ⇒ assume /24 (top bits = 110)

110*****	********	********	********

  - Small blocks (e.g., ICIR has 192.150.187.0/24)
  - The “swamp” (many European networks, due to history)

Classful Addressing (cont'd)

- Class D: [224..239] (top bits 1110)

1110****	********	********	********

  - Multicast groups
- Class E: [240..255] (top bits 11110)

11110***	********	********	********

  - Reserved for future use

- What problems can classful addressing lead to?
  - Only comes in 3 sizes
  - Routers can end up knowing about a **lot** of class C's

IP Address Problem (1991)

- Address space depletion
  - In danger of running out of classes A and B
  - Why?
    - Class C too small for most domains
    - Very few class A – very careful about giving them out
    - Class B – greatest problem
- Class B sparsely populated
  - But people refuse to give it back
- Large forwarding tables
  - 2 Million possible class C groups

Classless Inter-Domain Routing (CIDR)

- Allows arbitrary split between network & host part of address
  - Do not use classes to determine network ID
  - Use common part of address as network number
  - E.g., addresses 192.4.16 - 192.4.31 have the first 20 bits in common. Thus, we use these 20 bits as the network number → 192.4.16/20
- Enables more efficient usage of address space (and router tables)
  - Use single entry for range in forwarding tables
  - Combined forwarding entries when possible

Classless Inter-Domain Routing (CIDR)

Use arbitrary length prefixes

Use two 32-bit numbers to represent a network.
Network number = IP address + Mask

IP Address : 12.4.0.0 IP Mask: 255.254.0.0

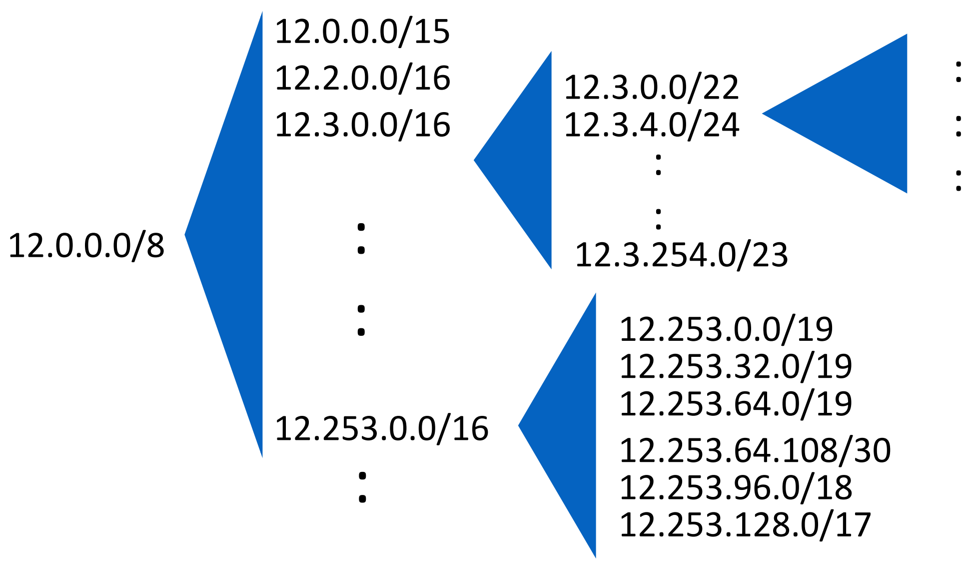
Address	00001100	00000100	00000000	00000000
Mask	11111111	11111110	00000000	00000000

|← Network Prefix →|← for hosts →|

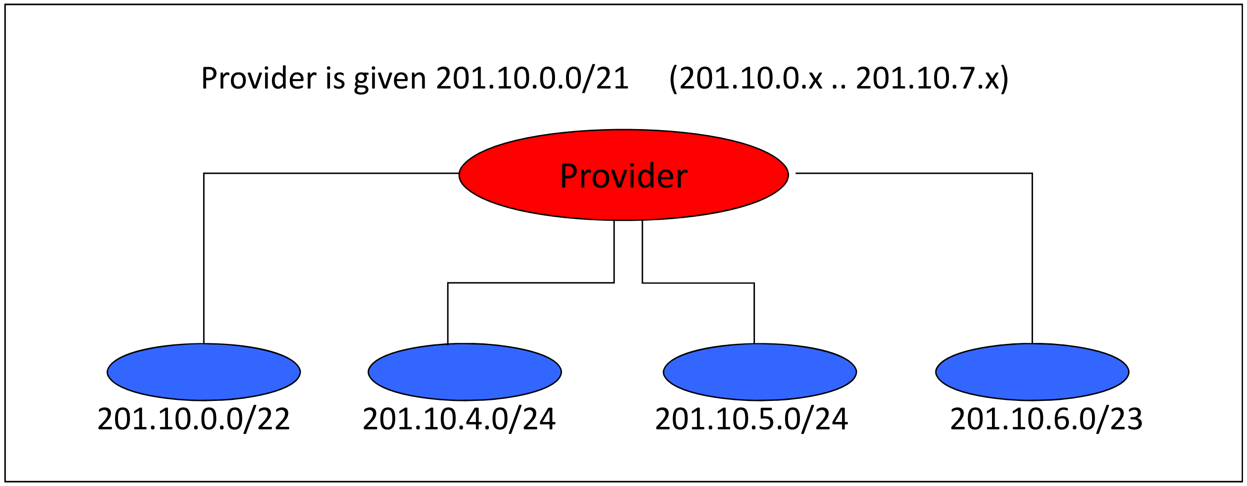
Written as 12.4.0.0/15 or 12.4/15

CIDR: Hierarchal Address Allocation

- Prefixes are key to Internet scalability
  - Addresses allocated in contiguous chunks (prefixes)
  - Routing protocols and packet forwarding based on prefixes

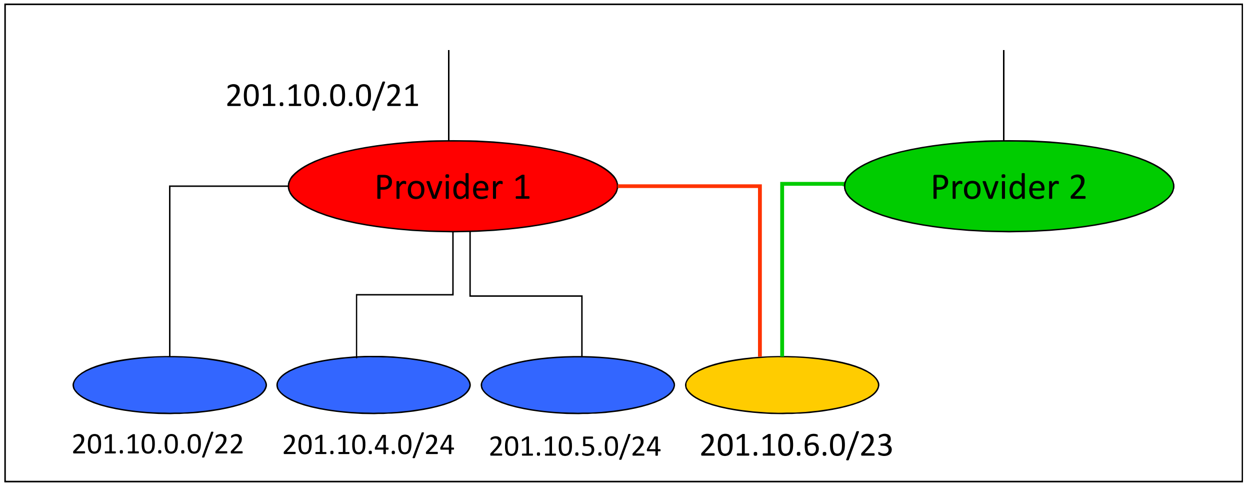


Scalability: Address Aggregation



Routers in the rest of the Internet just need to know how to reach 201.10.0.0/21. The provider can direct the IP packets to the appropriate customer.

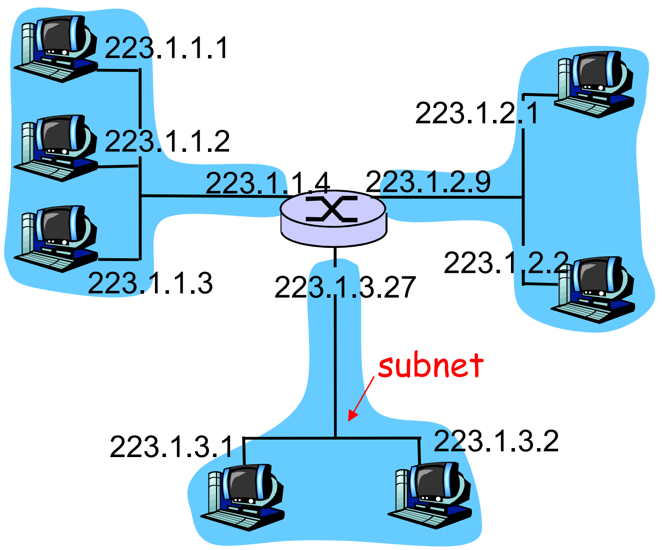
Aggregation Not Always Possible



Multi-homed customer with 201.10.6.0/23 has two providers. Other parts of the Internet need to know how to reach these destinations through *both* providers.
⇒ /23 route must be globally visible

Subnets

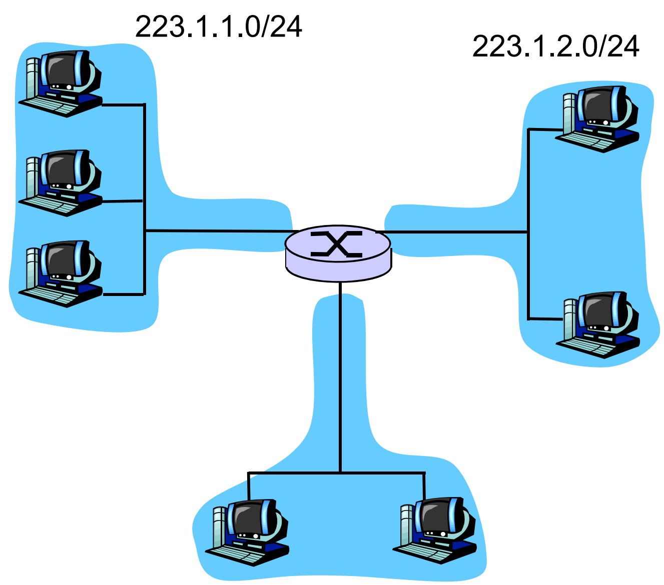
- IP address:
  - subnet part (high order bits)
  - host part (low order bits)
- *What's a subnet ?*
  - device interfaces with same subnet part of IP address
  - can physically reach each other without intervening router



network consisting of 3 subnets

Subnets

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



Subnet mask: /24

IP Datagram Forwarding

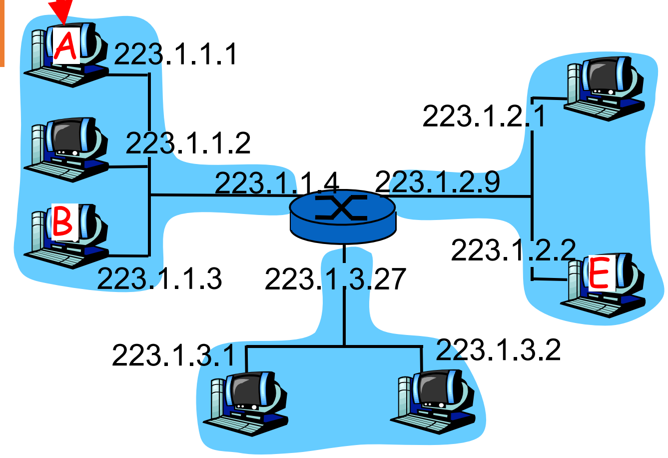
IP datagram:

misc fields	source IP addr	dest IP addr	data

- datagram remains unchanged, as it travels source to destination
- addr fields of interest here

forwarding table in A

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2

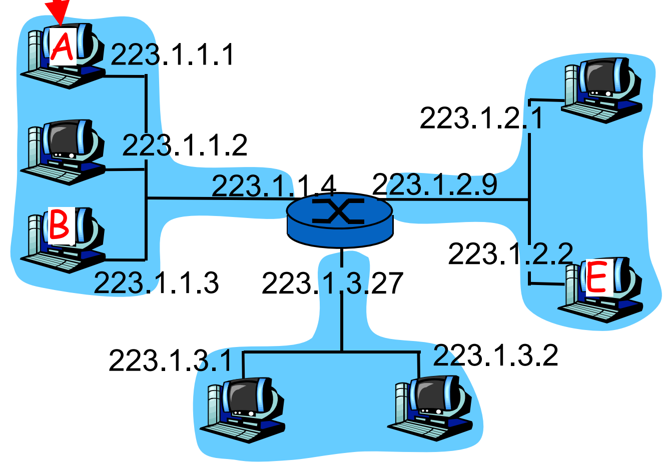


IP Forwarding: Destination in Same Net

misc fields	223.1.1.1	223.1.1.3	data

forwarding table in A

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Starting at A, send IP datagram addressed to B:

- look up net. address of B in forwarding table
- find B is on same net. as A
- link layer will send datagram directly to B inside link-layer frame
  - B and A are directly connected

IP Forwarding: Destination in Diff. Net

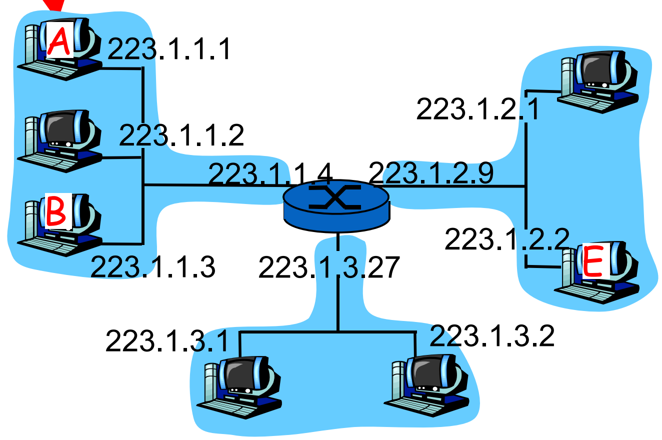
misc fields	223.1.1.1	223.1.2.3	data

Starting at A, dest. E:

- look up network address of E in forwarding table
- E on *different* network
  - A, E not directly attached
- routing table: next hop router to E is 223.1.1.4
- link layer sends datagram to router 223.1.1.4 inside link-layer frame
- datagram arrives at 223.1.1.4
- continued…..

forwarding table in A

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



IP Forwarding: Destination in Diff. Net …

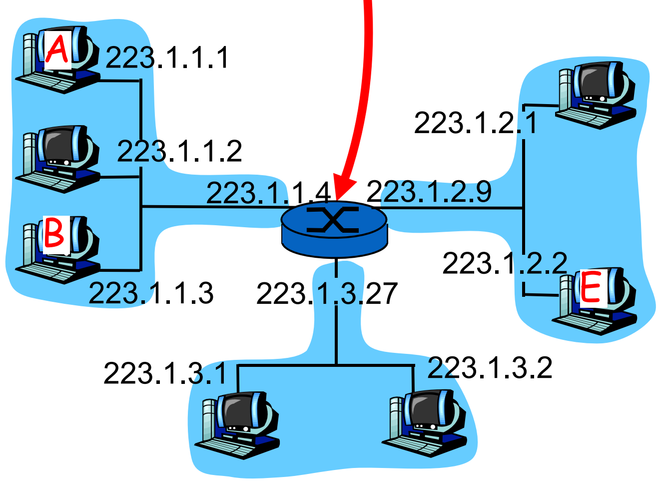
misc fields	223.1.1.1	223.1.2.3	data

forwarding table in router

Dest. Net	router	Nhops	interface
223.1.1	-	1	223.1.1.4
223.1.2	-	1	223.1.2.9
223.1.3	-	1	223.1.3.27

Arriving at 223.1.4, destined for 223.1.2.2

- look up network address of E in router's forwarding table
- E on *same* network as router's interface 223.1.2.9
  - router, E directly attached
- link layer sends datagram to 223.1.2.2 inside link-layer frame via interface 223.1.2.9



Are 32-bit Addresses Enough?

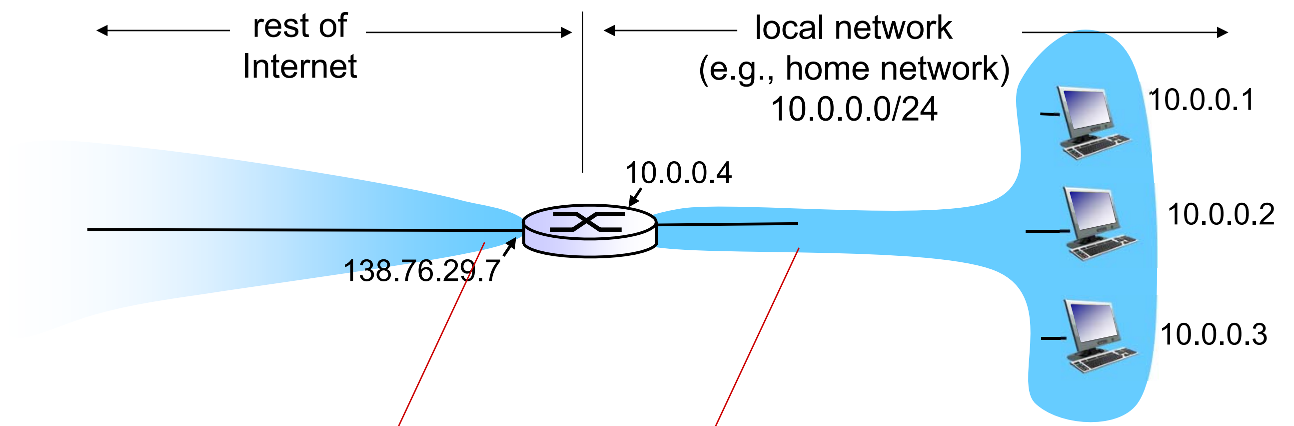
- Not all that many unique addresses
  - 2^{32} = 4,294,967,296 (just over four billion)
  - Many reserved for special purposes
  - Addresses are allocated in larger blocks
- And, many devices need IP addresses
  - Computers, Mobile phones, vehicles, appliances, …
- Long-term solution (perhaps): larger address space
  - IPv6 has 128-bit addresses ($2^{128} = 3.403 \times 10^{38}$)
- Short-term solutions: limping along with IPv4
  - Private addresses
  - Network address translation (NAT)
  - Dynamically-assigned addresses (DHCP)

Special-Purpose Address Blocks

- Private addresses
  - By agreement, not routed in the public Internet
  - For networks not meant for general Internet connectivity
  - Blocks: **10.0.0.0/8**, **172.16.0.0/12**, **192.168.0.0/16**
- Link-local
  - By agreement, not forwarded by any router
  - Used for single-link communication only
  - Intent: autoconfiguration (especially when *DHCP* fails)
  - Block: **169.254.0.0/16**
- Loopback
  - Address blocks that refer to the local machine
  - Block: **127.0.0.0/8**
  - Usually only **127.0.0.1/32** is used
- Limited broadcast
  - Sent to every host attached to the local network
  - Block: **255.255.255.255/32**

NAT (Network Address Translation)

A fix to limited IP address space:



all datagrams *leaving* local network have *same* single source NAT IP address: 138.76.29.7,different source port numbers

datagrams with source or destination in this network have 10.0.0.0/24 address for source, destination (as usual)

NAT (Network Address Translation)

motivation: local network uses just one IP address as far as outside world is concerned:

- range of addresses not needed from ISP: just one IP address for all devices
- can change addresses of devices in local network without notifying outside world
- can change ISP without changing addresses of devices in local network
- devices inside local net not explicitly addressable, visible by outside world (a security plus)

Properties of Firewalls with NAT

- Advantages
  - Hides IP addresses used in internal network
    - Easy to change ISP: only NAT box needs to have IP address
    - Fewer registered IP addresses required
  - Basic protection against remote attack
    - Does not expose internal structure to outside world
    - Can control what packets come in and out of system
    - Can reliably determine whether packet from inside or outside
- Disadvantages
  - Contrary to the “open addressing” scheme envisioned for IP addressing
  - Hard to support peer-to-peer applications

NAT Considerations

- NAT has to be consistent during a session.
  - Set up mapping at the beginning of a session and maintain it during the session
    - What happens if your NAT reboots or fails?
  - Recycle the mapping at the end of the session
    - May be hard to detect
- NAT only works for certain applications.
  - Some applications (e.g. ftp) pass IP information in payload
  - Need application level gateways to do a matching translation
  - Breaks a lot of applications.
- NAT is loved and hated
  - \- Breaks many apps (FTP)
  - \- Inhibits deployment of new applications like p2p (but so do firewalls!)
  - \+ Little NAT boxes make home networking simple.
  - \+ Saves addresses. Makes allocation simple.

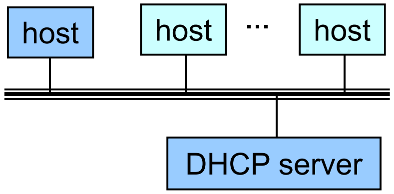
IP Addresses: Bootstrapping

Q: How does *host* get IP address?

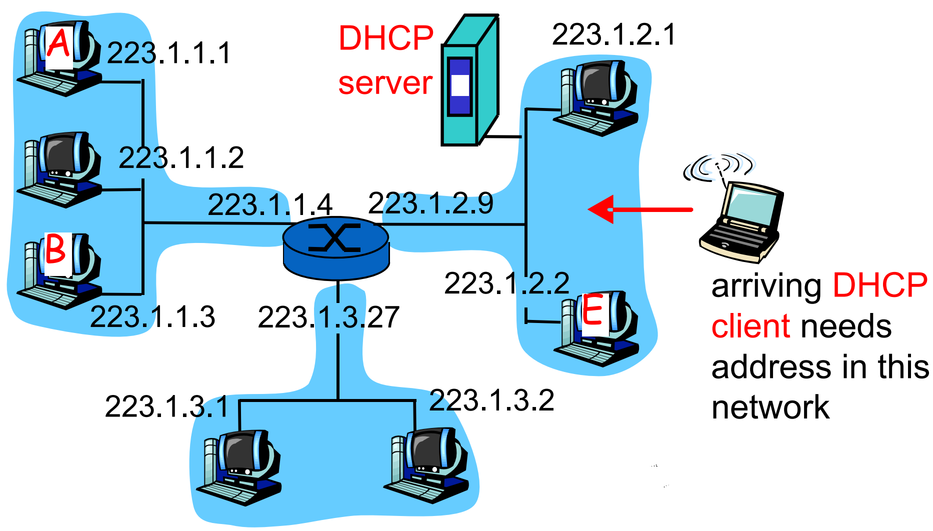
- "static" assigned: i.e., hard-coded in a file
  - Win: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- Dynamically assigned: using DHCP (Dynamic Host Configuration Protocol)
  - dynamically get address from a server
  - "plug-and-play"

Bootstrapping Problem

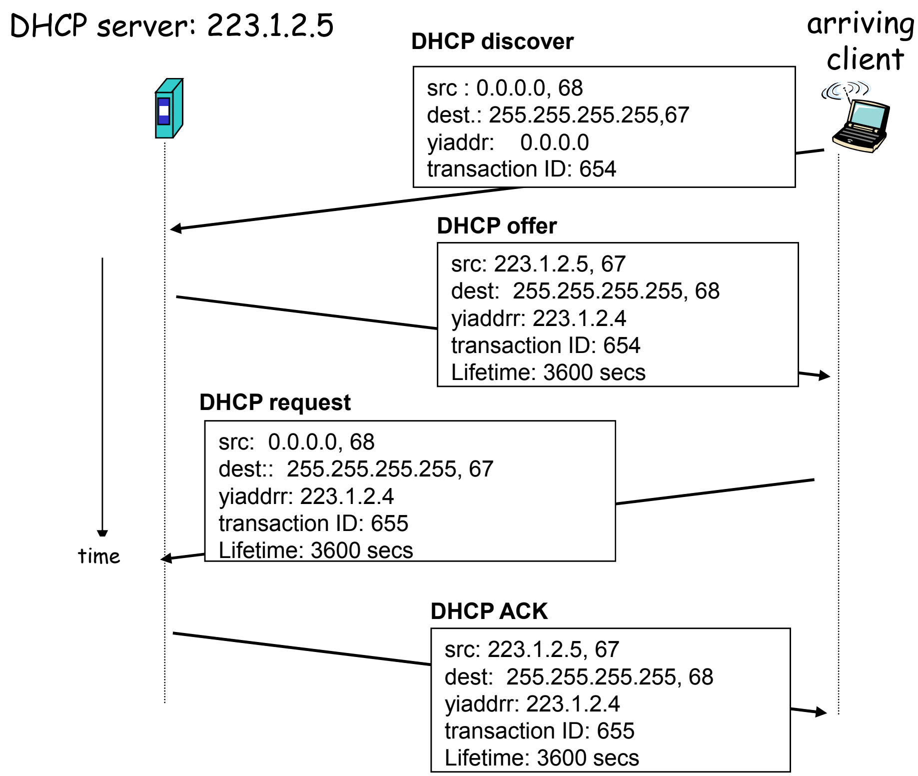
- Host doesn’t have an IP address yet
  - So, host doesn’t know what source to use
- Host doesn’t know who to ask for an IP address
  - So, host doesn’t know what destination to use
- Solution: discover a server who can help
  - Broadcast a DHCP server-discovery message
  - Server sends a DHCP “offer” offering an address



DHCP Client-Server Scenario



DHCP Client-Server Scenario



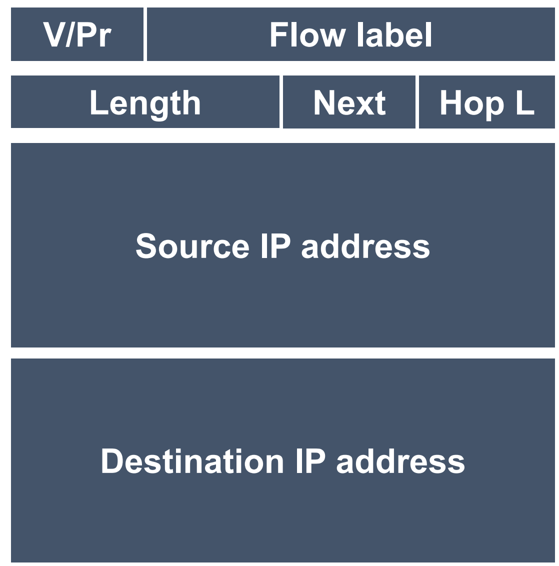
DHCP: Additional Functions

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

- address of first-hop router for client
- name and IP address of DNS sever
- network mask (indicating network versus host portion of address)

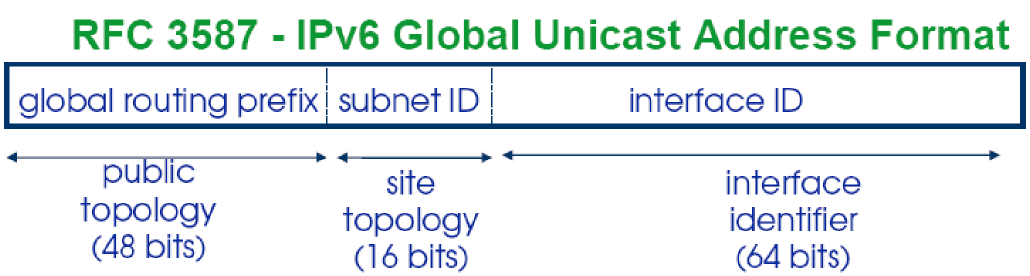
IPv6

- “Next generation” IP.
- Most urgent issue: increasing address space.
  - 128 bit addresses
- Simplified header for faster processing:
  - No checksum
  - No fragmentation
- Support for guaranteed services: priority and flow id
- Options handled as “next header”
  - reduces overhead of handling options



IPv6 Unicast Address

- Global routing prefix
  - A (typically hierarchically-structured) value assigned to a site (a cluster of subnets/links)
- Subnet ID
  - An identifier of a subnet within the site
- Interface ID
  - Constructed in Modified EUI-64 format



IPv6 Autoconfiguration

- Serverless ("Stateless"). No manual config at all.
  - Only configures addressing items, NOT other host things
    - If you want that, use DHCP.
- Link-local address
  - 1111 1110 10 :: 64 bit interface ID (usually from Ethernet addr)
    - (fe80::/64 prefix)
  - Uniqueness test ("anyone using this address?")
  - Router contact (solicit, or wait for announcement)
    - Contains globally unique prefix
    - Usually: Concatenate this prefix with local ID → globally unique IPv6 ID

IPv6 Header Cleanup

- Different options handling
- IPv4 options: Variable length header field. 32 different options.
  - Rarely used
  - No development / many hosts/routers do not support
    - Worse than useless: Packets w/options often even get dropped!
  - Processed in “slow path”.
- IPv6 options: “Next header” pointer
  - Combines “protocol” and “options” handling
    - Next header: “TCP”, “UDP”, etc.
  - Extensions header: Chained together
  - Makes it easy to implement host-based options
  - One value “hop-by-hop” examined by intermediate routers
    - Things like “source route” implemented only at intermediate hops

IPv6 Header Cleanup

- No checksum
- Why checksum just the IP header?
  - Efficiency: If packet corrupted at hop 1, don't waste b/w transmitting on hops 2..N.
  - Useful when corruption frequent, b/w expensive
  - Today: Corruption rare, b/w cheap

IPv6 Fragmentation Cleanup

- IPv6:
  - Discard packets, send ICMP “Packet Too Big”
    - Similar to IPv4 “Don’t Fragment” bit handling
  - Sender must support Path MTU discovery
    - Receive “Packet too Big” messages and send smaller packets
  - Increased minimum packet size
    - Link must support 1280 bytes;
    - 1500 bytes if link supports variable sizes
- Reduced packet processing and network complexity.
- Increased MTU a boon to application writers
- Hosts can still fragment - using fragmentation header. Routers don’t deal with it any more.

Migration from IPv4 to IPv6

- Alternative mechanisms:
  - Dual stack operation: IP v6 nodes support both address types
  - Translation:
    - Use form of NAT to connect to the outside world
    - NAT must not only translate addresses but also translate between IPv4 and IPv6 protocols
  - **Tunneling**: tunnel IP v6 packets through IP v4 clouds
    - Encapsulate IPv6 packets in IPv4 datagrams before sending on IPv4 network
    - Decapsulate at the end of IPv4 network