IP : Internet Protocol

Chapters 18,19

Introduction

- One key aspect of virtual network is single, uniform address format
  - Can't use hardware addresses because different technologies have different address formats
  - Address format must be independent of any particular hardware address format
- When we address a packet
  - Sending host puts destination internet address in packet
  - Destination address can be interpreted by any intermediate router
  - Routers examine address and forward packet on to the destination
- All of these addresses are virtual; they are defined in software, not hardware

TCP/IP Addresses

- Addressing in TCP/IP is specified by the *Internet Protocol* (IP)
- Each host is assigned a 32-bit number
  - Called the *IP address* or *Internet address*
  - Unique across entire Internet
- Each IP address is divided into a prefix and a suffix
  - Prefix identifies network to which computer is attached
    - No two networks can be assigned the same network number
  - Suffix identifies computer within that network
    - No two computers on the same network can have the same suffix, but computers on different networks can have the same suffix
  - Address format makes routing efficient

Properties of IP Addresses

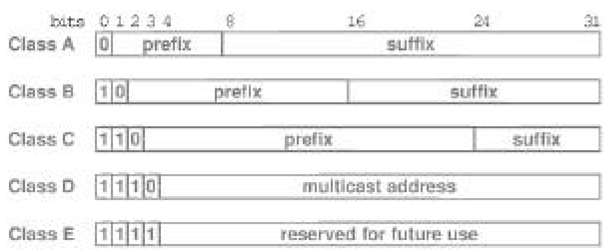
- Network numbers (prefix) are unique
  - Assignment of network numbers must be coordinated globally to ensure uniqueness
  - Assigned by ICANN (formerly InterNIC/Network Solutions)
- Host addresses (suffix) may be reused on different networks
  - combination of network number prefix and host address suffix will be unique
  - assignment of host addresses can be managed locally
    - E.g. your ISP can set the host addresses and names within the purview of their assigned network by ICANN

Designing IP

- IP designers chose 32-bit addresses
  - Still 32 bits in IP version 4 used today
  - May cause a problem soon, not a large enough address space!
- Allocate some bits for prefix, some for suffix
  - Large prefix, small suffix - many networks, few hosts per network
  - Small prefix, large suffix - few networks, many hosts per network
- Because of variety of technologies, need to allow for both large and small networks

Classful Addressing

- Designers chose a compromise - multiple address formats that allow both large and small prefixes
- Each format is called an address *class*
- Class of an address is identified by first four bits



Address Classes

- Class A, B and C are *primary classes*
  - Used for ordinary host addressing
  - Owner of a class is assigned a prefix, gets to pick what machines they want in the suffix
  - Classes A and B are all allocated!
- Class D is used for multicast, a limited form of broadcast
  - Internet hosts join a *multicast group*
  - Packets are delivered to all members of group
  - Routers manage delivery of single packet from source to all members of multicast group
  - Used for *mbone* (multicast backbone)
- Class E is reserved for future use

Dotted Decimal Notation

- Class A, B and C all break between prefix and suffix on byte boundary
- *Dotted decimal notation* is a convention for representing 32-bit internet addresses in decimal
- Convert each byte of address into decimal; display separated by periods (``dots")

32-bit Binary Number	Equivalent Dotted Decimal
10000001 00110100 00000110 00000000	129 . 52 . 6 . 0
11000000 00000101 00110000 00000011	192 . 5 . 48 . 3
00001010 00000010 00000000 00100101	10 . 2 . 0 . 37
10000000 00001010 00000010 00000011	128 . 10 . 2 . 3
10000000 10000000 11111111 00000000	128 . 128 . 255 . 0

Addressing at UA

- The University of Alaska has a single Class B network address, 137.229.0.0
- All hosts at UA have a 137.229 prefix:
  - – 137.229.134.207 – mazzy.math.uaa.alaska.edu
  - – 137.229.16.20 – www.uaf.alaska.edu
  - – 137.229.150.11 – vor.uas.alaska.edu
- Suffix bytes are used to determine local network and host through *subnetting*

Address Class at a Glance

- While dotted decimal makes separating network address from host address easier, determining class is not so obvious
- Look at first dotted decimal number, and use this table:

Class	Range of Values
A	0 through 127
B	128 through 191
C	192 through 223
D	224 through 239
E	240 through 255

Just a little binary number conversion based on first 4 bits

Division of the Address Space

- Classing scheme does not yield equal number of networks in each class
- Class A:
  - First bit must be 0
  - 7 remaining bits identify Class A net
  - 2^7 (= 128) possible class A nets
- For the rest:

Address Class	Bits In Prefix	Maximum Number of Networks	Bits In Suffix	Maximum Number Of Hosts Per Network
A	7	128	24	16777216
B	14	16384	16	65536
C	21	2097152	8	256

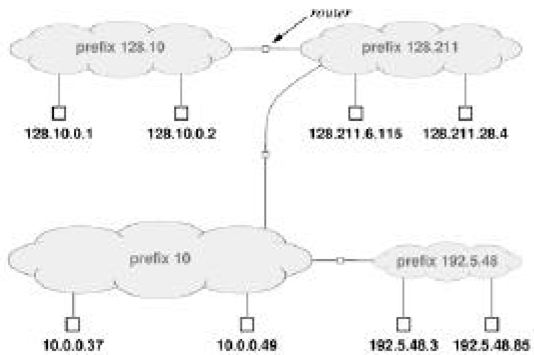
Running out of IP's? Not allocated very efficiently
2^{32} = ~4 billion possible hosts
But not all of the hosts being used! Some IP's reserved too
Does UA have 65536 hosts? Stanford used to own a Class A
Solutions : Classless addressing, private address space

Reserved Addresses

- RFC 1597 – Address Allocation for Private Internets
  - The following blocks are reserved for private internets
    - 10.0.0.0 - 10.255.255.255 Class A
    - 172.16.0.0 - 172.31.255.255 Class B
    - 192.168.0.0 - 192.168.255.255 Class B
    - We'll need a gateway to translate from these addresses to the Internet addresses, and we'll examine this later… common way to connect a home network to the Internet
    - We should never see a machine on the Internet with one of these network prefixes (hopefully)

Classful Addressing Example

- Organization with four networks, completely private network
  - Assign Class A,B,C addresses as appropriate for number of hosts on each network



More Special IP Addresses

- Network Address
  - The network itself is assigned an address
  - So no host can have all zero’s as its IP address suffix
  - Prefix is the network prefix, suffix is all zero’s
- Directed Broadcast Address
  - Broadcast message to a network
  - Prefix is the network prefix, suffix is all one’s
  - So no host can have all one’s as its IP address suffix
- Limited Broadcast Address
  - Broadcast on the local LAN
  - Entire address is all 1’s, i.e. 255.255.255.255

More Special IP Addresses

- This Computer Address
  - To obtain an address automatically when booting, we may use IP to communicate… but we don't have a correct IP address yet
  - Use an address of all zero's to indicate "this computer"
- Loopback Address
  - Any address beginning with 127 indicates the local computer
  - E.g. 127.0.0.1 most common, but could be 127.0.44.53
  - Use for testing, no packets leave the computer

Special Address Summary

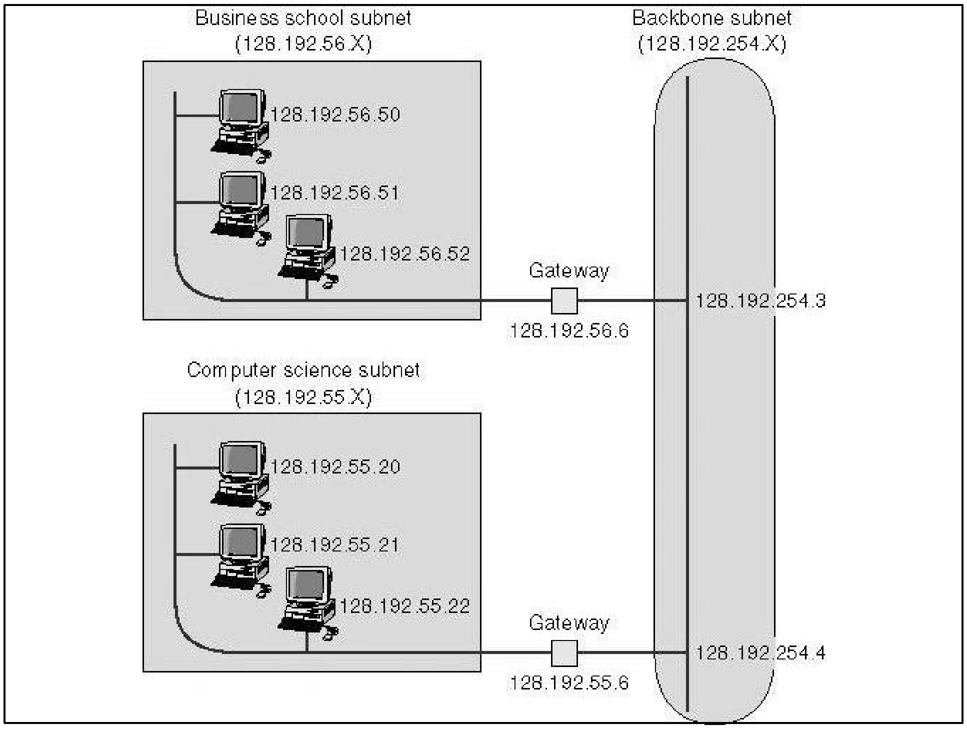
Prefix	Suffix	Type Of Address	Purpose
all-0s	all-0s	this computer	used during bootstrap
network	all-0s	network	identifies a network
network	all-1s	directed broadcast	broadcast on specified net
all-1s	all-1s	limited broadcast	broadcast on local net
127	any	loopback	testing

Subnet and Classless Addressing

- Classful Addressing can be wasteful if we don't utilize all the hosts within a network Class
- Two mechanisms to overcome the limitations
  - Subnetting
  - Classless Addressing
- Instead of three distinct network classes, allow the division between prefix and suffix to occur on arbitrary bit boundaries
- Example:
  - Network with 9 hosts; only need 4 bits of suffix to encode
  - But the smallest class is class C, 254 hosts, so we'll waste 245 possible IP addresses
  - Classless addressing lets have the 4 bits of suffix we need

Subnets

- In general, IP addresses are assigned so that all computers on the same local area network are on the same **subnet**
- It is customary to use the last byte of the IP address to indicate different subnets
  - i.e. class C address
- But any portion of the IP address can be designated as a subnet by using a **subnet mask**. IP addresses are binary numbers, so partial bytes can be used as subnets.



Subnet Masks

- Typical subnet mask: 255.255.255.0
  - First 3 bytes all on the same subnet, only varies in the last byte
  - 255 = 11111111 in binary. The bits form a “mask” – where there is a 1, this means the subnet is the same.
    - 192.168.0.1, 192.168.0.2, 192.168.0.55
- Subnet mask: 255.255.254.0
  - Mask = 11111111 11111111 11111110 00000000
  - 1.1.1.5 on the same subnet as 1.1.0.5 ?
  - 00000001 00000001 00000001 00000101 vs
  - 00000001 00000001 00000000 00000101 **YES**
  - 1.1.1.5 on the same subnet as 1.1.2.5 ?
  - 00000001 00000001 00000001 00000101 vs
  - 00000001 00000001 00000010 00000101 **NO**

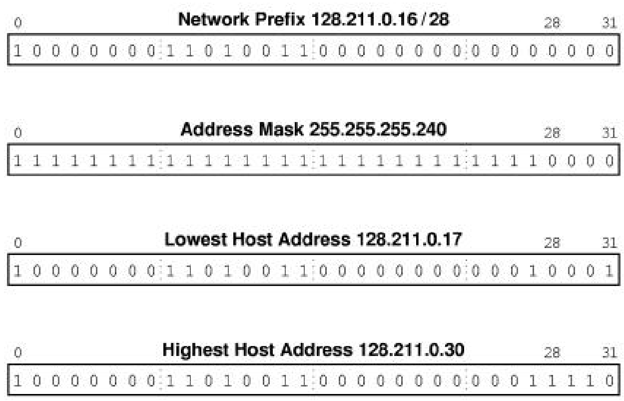
Subnets and Subnet Masks

- Subnet mask lets us divide the address space on arbitrary bit boundaries
  - Good for better allocation of IP addresses
- How will computers and routers use the subnet mask?
  - A host
    - When sending to a destination IP address
    - Compare destination IP address to our own subnet mask
    - If dest is on the same subnet, just broadcast data on LAN
    - If dest on another subnet, send to a gateway or router
- Local routers route within subnetted network
  - Use subnet mask with the network address of attached networks to determine proper destination

CIDR Notation

- CIDR = Classless Inter-Domain Routing
  - Notation for classless addresses
  - Specifies the mask associated with an address
  - Appends a slash to the address with the size of the mask in decimal
    - Examples:
      - 128.10.0.0/16
        - » says 16 bit prefix, 16 bit suffix (Class B)
      - 128.211.0.16/28
        - » 28 bit prefix, 4 bit suffix

CIDR Example



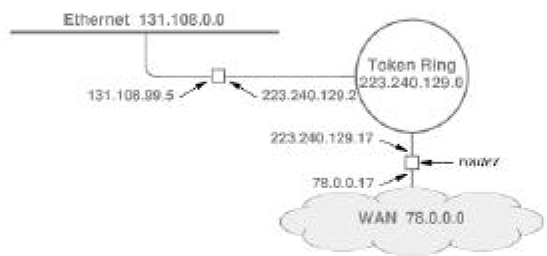
Why didn't the lowest start at 0000 and highest at 1111?
One disadvantage; not so easy to read host values from decimal

Dynamic Addressing

- Every computer knows its network layer address from a configuration file. However, each time the computer is moved, or its network is assigned a new address, the software on each individual computer must be updated.
- Solution: dynamic addressing. With this approach, a server is designated to supply a network layer address to a computer each time the computer connects to the network from an available "pool" of addresses
  - Assigns "Next Available" address
  - Simplifies roaming computers, dial-up
- Two standards for dynamic addressing are commonly used in TCP/IP networks:
  - Bootstrap Protocol (bootp)
  - Dynamic Host Control Protocol (DHCP)

Routers and IP Addresses

- IP address depends on network address
- What about routers - connected to two networks?
- IP address specifies an *interface*, or network attachment point, not necessarily a computer
  - Router has multiple IP addresses - one for each interface



Multi-Homed Hosts

- Hosts (that do not forward packets) can also be connected to multiple networks
  - If it forwards packets, it’s a router
- Can increase reliability and performance
  - If one network fails, the host can still reach the internet through the second connection
  - Can also send data on multiple networks for increased performance
- Multi-homed hosts also have one address for each interface

Binding Protocol Addresses

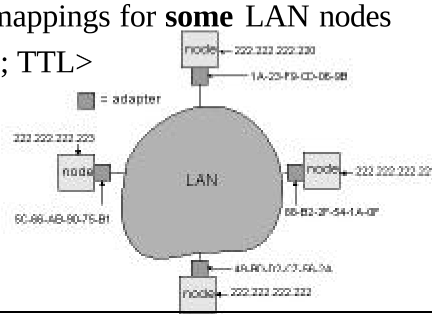
- Upper levels of protocol stack use *protocol addresses* , e.g. IP Address
- Network hardware must use *hardware address* for eventual delivery
- Protocol address must be translated into hardware address for delivery; how can this be done?
  - This address translation is only necessary for the local LAN.
    - Why? Consider if we need the hardware address if sending to a remote computer across the Internet…

Ethernet ARP: Address Resolution Protocol

- Ethernet (MAC) addresses are different than IP addresses. Given an IP address for a machine on the same LAN, what is its MAC address?
  - Solution: **ARP -** Address Resolution Protocol
- Each IP node (Host, Router) on the LAN has ARP module and Table. This operates at the **Data Link** layer
- ARP Table: IP/MAC address mappings for **some** LAN nodes

  < IP address; MAC address; TTL>

  …



- TTL (Time To Live): timer, typically 20 min

ARP (more)

- Host A wants to send packet to destination IP addr XYZ on same LAN
- Source Host first checks own ARP Table for IP addr XYZ
- If XYZ **not** in the ARP Table, ARP module **broadcasts** ARP pkt:
- < XYZ, MAC (?) >
- ALL nodes on the LAN accept and inspect the ARP packet
- Node XYZ responds with **unicast** ARP pkt carrying own MAC addr:
- < XYZ, MAC (XYZ) >
- MAC address **cached** in ARP Table for future use
- Host A then sends **unicast** data packet to MAC(XYZ)

- Why not broadcast the data in the first place rather than incur three separate transmissions?

ARP Table Lookup

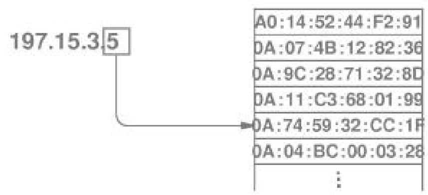
- Use a simple list containing IP address and hardware address for each host on net
- Search on IP address and extract corresponding hardware address

IP Address	Hardware Address
197.15.3.2	0A:07:4B:12:82:36
197.15.3.3	0A:9C:28:71:32:8D
197.15.3.4	0A:11:C3:68:01:99
197.15.3.5	0A:74:59:32:CC:1F
197.15.3.6	0A:04:BC:00:03:28
197.15.3.7	0A:77:81:0E:52:FA

  - Note that all IP addresses have same prefix; can save space by dropping prefix

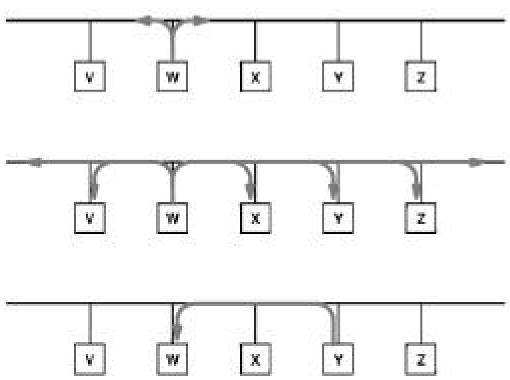
ARP Table Lookup

- Sequential search may be prohibitively expensive ($O(n)$)
- Can use *indexing* or *hashing* for $O(1)$ lookup
- Indexing - use *hostid* part of IP address as list (array) index



- Hashing - use hashing function on *hostid* to generate list index

ARP Message Exchange Example



W ARP's for Y's IP Address
Everyone hears the ARP
Only Y responds with a Unicast message (still broadcast on bus)

ARP Message Contents

- Maps *protocol address* to *hardware address*
- Both protocol address and hardware address sizes are variable
  - – Ethernet = 6 octets
  - – IP = 4 octets
- Can be used for other protocols and hardware types but in reality just used for Ethernet/IP

ARP Message Format

0	8	16	24 31
HARDWARE ADDRESS TYPE		PROTOCOL ADDRESS TYPE	
HADDR LEN	PADDR LEN	OPERATION	
SENDER HADDR (first 4 octets)			
SENDER HADDR (last 2 octets)		SENDER PADDR (first 2 octets)	
SENDER PADDR (last 2 octets)		TARGET HADDR (first 2 octets)	
TARGET HADDR (last 4 octets)			
TARGET PADDR (all 4 octets)			

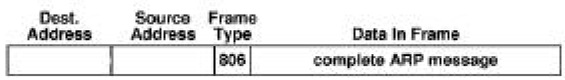
•*HARDWARE ADDRESS TYPE* = 1 for Ethernet
•*PROTOCOL ADDRESS TYPE* = 0x0800 for IP
•*OPERATION* = 1 for request, 2 for response

  •Contains both *target* and *sender* mappings from protocol address to hardware address
  •Request sets hardware address of target to 0
  •Target can extract hardware address of sender (saving an ARP request)
  •Target exchanges sender/target in response

Sending an ARP Request

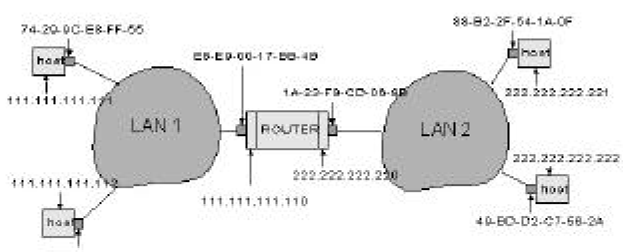
- Sender constructs ARP message
- ARP message carried as data in hardware frame - *encapsulation*

- Uses separate frame type
- Ethernet uses type 0x0806



Routing Packet to another LAN

- Say, route packet from source IP addr <111.111.111.111> to destination addr <222.222.222.222>



- In routing table at source Host, find router 111.111.111.110
- In ARP table at source, find MAC address E6-E9-00-17-BB-4B, etc. **ARP does not span LANs!**