Internetworking and Internet

Global Addresses

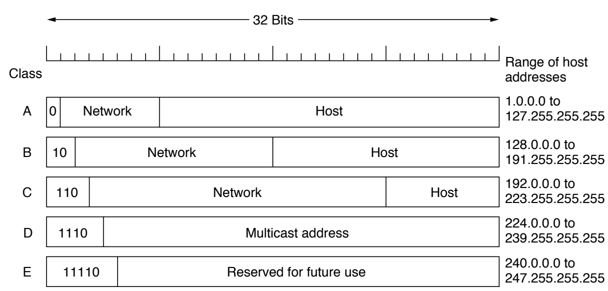
IP servcie model has two parts –

- Datagram (connectionless) packet delivery model
- **Global addressing scheme** – a way to identify **all H** in the internetwork

- Properties
  - **globally unique**
  - **hierarchical** – network part + host part

$\longleftrightarrow$ Ethernet's address is globally unique but *flat*

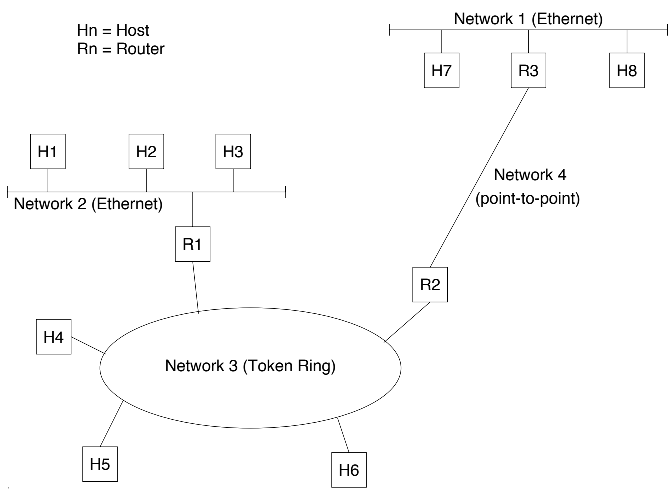
- Format



- $\boxed{\textbf{IP address belongs to interface than host}}$

Datagram Forwarding

- The 'network part' of an IP address uniquely identifies a single physical network
- All H and R sharing the same 'network part' of their address are connected to the same physical network and can thus communicate by sending *frames* over that network



Example – R2's routing table

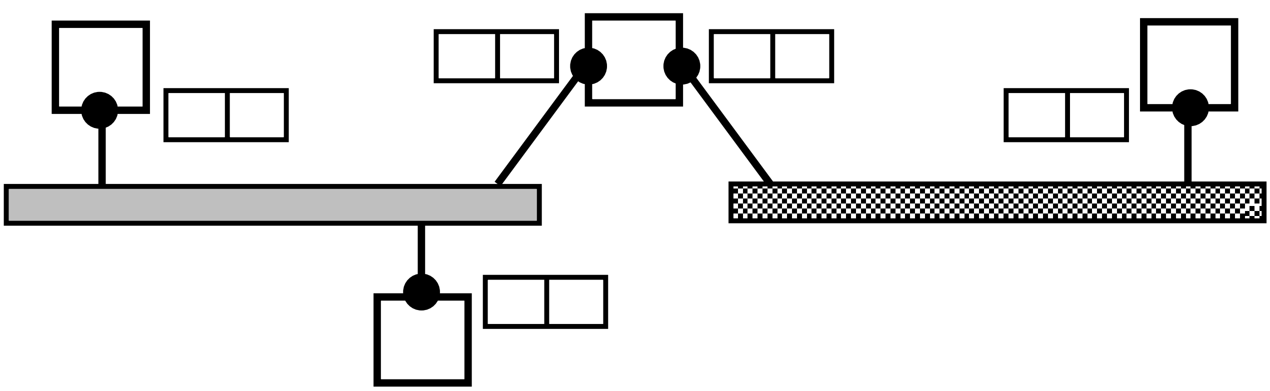
Network Part Number	Next Hop
1	R3
2	R1
3	interface 1
4	interface 0

Examples – H1 $\longrightarrow$ H2 and H1 $\longrightarrow$ H8

Hierarchical addressing – reduce routing table size to achieve *scalability*

Datagram Forwarding – a closer look

```
if (destination network == network part of an interface)
  deliver packet to destination over that interface
else
  if (network part of destination in routing table)
    deliver packet to NextHop router
  else
    deliver packet to default router
```



Lesson learned – to achieve scalability, need to reduce the amount of information stored in each node and exchanged between nodes
$\Longleftarrow$ **hierarchical aggregation**

Address Resolution Protocol (ARP)

Deliver over Physical Networks

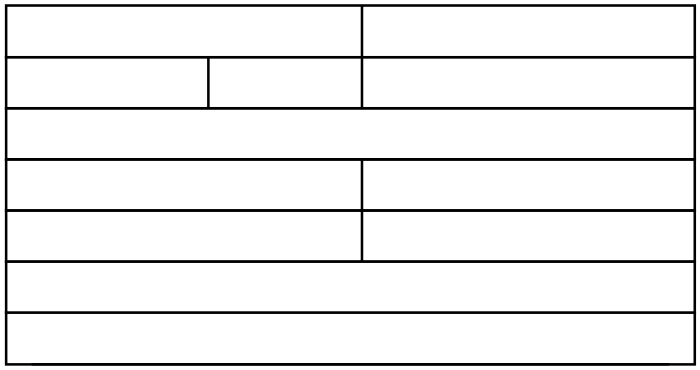
1. get IP packet to the right physical *network*

2. get IP packet to the right destination *node* (R or H)

⇒ physical interface only understand network technology specific addressing scheme (data-link layer address)

- **Map IP address (of destination H or next hop R) into physical (Data Link Layer) address**
  - Techniques
    - encode physical address in **host part** of IP address
    - table of [IP address, physical address] pairs
  - **Address Resolution Protocol (ARP)**
    - each node build table of IP to physical address bindings *dynamically*
    - **broadcast** an **ARP Query [target IP address]** if the IP address is not in table
    - **target machine with matching IP address** responds with its physical address
- **Encapsulate IP packet inside a frame containing the DLL address for delivery**

ARP request format for IP-to-**Ethernet** address mapping



- HardwareType – type of physical network (e.g., Ethernet)
- ProtocolType – type of higher layer protocol (e.g., IP)
- HLEN & PLEN – length of physical and protocol (IP) addresses
- Operation – request or response
- Source/Target Physical/Protocol addresses

Reverse ARP

- ARP – IP address $\longrightarrow$ DLL address
- RARP – DLL address $\longrightarrow$ IP address
  - Boot a diskless workstation
  - Use Ethernet broadcast address (all 1s)
    $\Rightarrow$ restricted within the same LAN (a RARP server is needed for each network)
- BOOTP – use UDP to get more configuration information (*e.g.* IP addresses of default router and file server)

Summary – Lessons Learned

- Heterogeneity
  - best-effort service model
    - makes minimum assumption about underlying networks
    - is based on unreliable datagram
  - + a common packet format with fragmentation and reassembly for different MTU
  - + a global address space for identifying all hosts with ARP for different physical addresses
- Scale
  - hierarchical aggregation to reduce routing information
  - IP address is partitioned into network and host components with packet first routed toward the destination network and then delivered to the correct host on that network

Global Internet

So far, we have learned how to

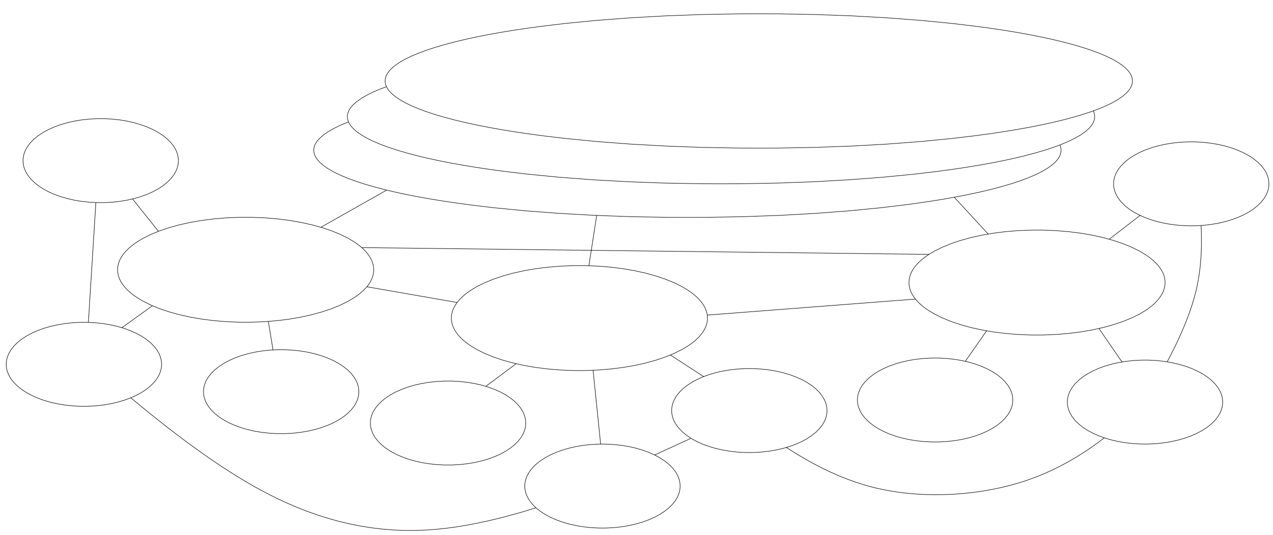
- connect **heterogeneous** networks to create an internetwork
- use the **hierarchy** of IP address to make routing *scalable*

→ each R needs *not* know **all the hosts**

→ however, each R needs to know **all the networks**

⟹ not enough for *the* **Internet**

Hierarchical structure of the Internet



- multiple sites
- nearby sites are interconnected by regional networks
- regional networks are connected by nationwide backbones (*e.g.* NSFNET)

Scalability Issues and Solutions

- **Address utilization - inefficient use of address space**
  - class B network with 256 hosts ($256/65535 = 0.39\%$ efficient)
  - class C network with 2 hosts ($2/255 = 0.78\%$ efficient)

  $\rightarrow$ *use up IP address space quickly*

  $\rightarrow$ *the more network #, the bigger the routing table*

  $\Rightarrow$ solution – **subnetting**
- **Routing scalability – too many networks**
  - today's Internet has tens of thousands of networks
  - routing tables do not scale
  - rotuing protocols do not scale

  $\Rightarrow$ solution – **route propagation**
- **Balance routing information against address utilization**

  $\Rightarrow$ solution – **classless routing (CIDR)**

♠ All based on the principle of **hierarchy**

Subnetting

Problem – **address assignment inefficiency**

- assignment of one network # per physical network uses up IP address space much faster than we would like

⇒ the more network #, the bigger the routing table

Solution – **subnetting** to reduce the total number of network # assigned

- Take a single IP network # and allocate IP addresses with that network # to several physical networks (**subnets**)
  - subnets are *close* to each other with *single point of entry*
  - introduce a subnet # via **subnet mask** – all H on the same physical network have the same subnet #
  - 3-part IP address – network part, subnet part, and H part

Network Number	Host Number

Class B address

111111111111111111111111	00000000

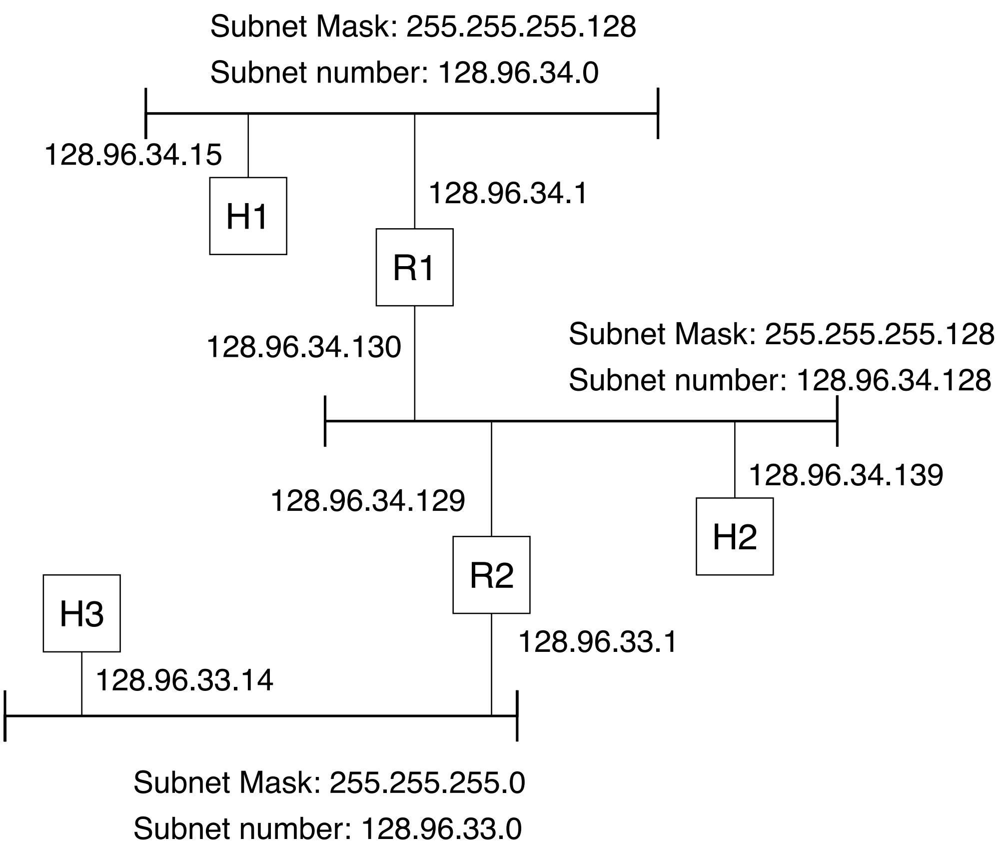
Subnet Mask (255.255.255.0)

Network Number	Subnet ID	Host ID

Subnetted Address

◇ Add another level of hierarchy into IP address

- H is configured with both an $\boxed{\text{IP address}}$ and a $\boxed{\text{subnet mask}}$
- all H on a subnet are configured with the same subnet #
- *Subnet masks* define variable partition of host part of Class B and C addresses
- Subnets visible only within site



Routing table at router R1

Subnet Number	Subnet Mask	Next Hop
128.96.34.0	255.255.255.128	interface 0
128.96.34.128	255.255.255.128	interface 1
128.96.33.0	255.255.255.0	R2

Forwarding algorithm

```
D = destination IP address
for each entry ⟨SubnetNum, SubnetMask, NextHop⟩
    D1 = SubnetMask & D      // logical AND
    if D1 = SubnetNum
        if NextHop is an interface
            deliver datagram directly to destination
        else
            deliver datagram to NextHop (a router)
```

- Not necessary for all 1s in subnet mask to be contiguous
- Can put multiple subnets on one physical network
  ⇒ force H on the same network to communicate via a R
- Subnets *not visible* from the rest of the Internet
  ⇒ seen as a single network
  ⟹ **not all parts of the Internet see exactly the same routing information**

Subnetting helps solve the scalability problem in two ways

- Improve address assignment efficiency
  - reduce the total number of network numbers assigned
  - not use up an entire class B/C address every time a physical network is introduced
- Aggregate information – { subnets } as a single network
  - reduce routing information stored in each router

CIDR – Supernetting

Problems

- Growth of backbone routing table
- Exhaustion of 32-bit IP address space

⇒ Subnetting did not resolve the issue that any AS with more than 255 H wants a Class B address

⇒ Exhaustion of IP address centers on Class B address

♠ assign an appropriate # of Class C addresses
⇒ more accurately match the amount of address space consumed to the size of an AS
⟹ larger routing table

⇒ 16 class C addresses (16 entries in routing table) vs. 1 class B address with efficiency $16 \times 255/65{,}536 = 6.2\%$

Need to balance the desire to **minimize the # of routes that a router needs to know** against **the need to hand out addresses efficiently**

Classless Inter-Domain Routing – address 2 problems *at the same time*

- Idea – use a single routing table entry to tell how to reach a lot of different networks – *route aggregation*
- Mechanism – break rigid boundaries between address classes
- Assign a block of **contiguous** network numbers to near-by networks, where each block must contain a number of Class C addresses that is a *power of 2 Rightarrow* addresses that share a common prefix

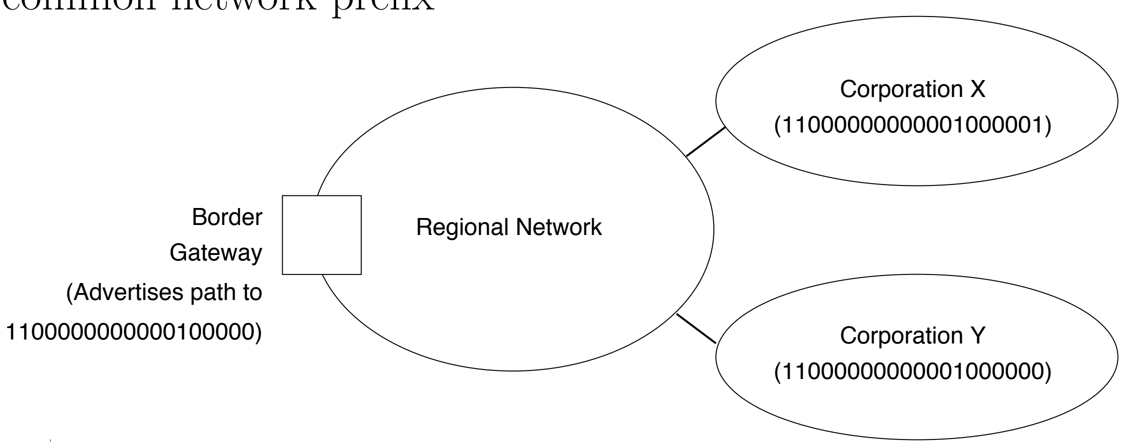
- *e.g.* – 16 contiguous Class C addresses (192.4.16 – 192.4.31) ⇒ 20-bit network # (Class C < # of H < Class B)

⇒ **high address efficiency and single network # in RT**

- network # ≡ ⟨ `mask, IP address` ⟩ with **longest match**
- BGP-4 routers understand CIDR (classless) addressing scheme

◇ Subnetting – share one address among multiple physical networks

◇ CIDR (**supernetting**) – collapse multiple addresses assigned to a single AS onto one address so that BGP advertises the common network prefix



Internet Scalability – Two main problems

- the amount of routing information transmitted between and stored in each node grows as the network grows
- the efficient use of address space

⇒ **add levels of hierarchy**, and allow enough flexibility to accommodate the complex structure of the Internet

Internetworking – Summary

- Goal – build large networks by interconnecting smaller ones
- Issues – **heterogeneity** and **scale**
- Key tool – Internet Protocol (IP)
- Heterogeneity – a simple service model for an internetwork
  - best-effort IP datagram delivery

  $\Rightarrow$ simple enough for any networking technology
  - global addressing scheme

  $\Rightarrow$ ARP: IP address $\longmapsto$ DLL address
- Scale
  - Issues
    - efficient use of address space
    - growth of routing table as the Internet grows
  - Solutions
    - hierarchical IP address format (network + host)
    - subnetting – add one more level of hierarchy to address $\Rightarrow$ make more efficient use of network # and consolidate routing information
    - autonomous system – partition the routing problem into two parts $\Rightarrow$ intra-domain and inter-domain
    - CIDR and BGP-4 – introduce more levels of hierarchy and achieve further routing aggregation
- IPv6 – Next Generation IP