Internetworking

- Multiple networks connected by routers.
- Networks share some features
  - IP protocol, addressing...
- But differ in many other ways
  - Technology, ownerships, usage policies, scale, ...

Hop-by-Hop Packet Forwarding in the Internet

Router Table Size

- One entry for every host on the Internet
  - 440M (7/06) entries, doubling every 2.5 years
- One entry for every LAN
  - Every host on LAN shares prefix
  - Still too many and growing quickly
- One entry for every organization
  - Every host in organization shares prefix
  - Requires careful address allocation
  - Still grows very quickly!
Addressing Considerations

- Hierarchical vs. flat
  - Pennsylvania / Pittsburgh / Oakland / CMU / CS
  - CS: (412)268-0000
- Scaling is key challenge
  - How well does Ethernet solution scale??
  - Hierarchy is a known effective solution
- Also want local administration -> hierarchical
- What type of Hierarchy?
  - How many levels?
  - Same hierarchy depth for everyone?
  - Same segment size for similar partition?

IP Addresses

- Fixed length: 32 bits
- Initial classful structure (1981) (not relevant now!!)
- Total IP address size: 4 billion
  - Class A: 128 networks, 16M hosts
  - Class B: 16K networks, 64K hosts
  - Class C: 2M networks, 256 hosts

<table>
<thead>
<tr>
<th>High Order Bits</th>
<th>Format</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>7 bits of net, 24 bits of host</td>
<td>A</td>
</tr>
<tr>
<td>10</td>
<td>14 bits of net, 16 bits of host</td>
<td>B</td>
</tr>
<tr>
<td>110</td>
<td>21 bits of net, 8 bits of host</td>
<td>C</td>
</tr>
</tbody>
</table>

IP Address Classes
(Some are Obsolete)

- Original IP Route Lookup

  - Address specifies prefix for forwarding table
    - Simple lookup
  - www.cmu.edu address 128.2.11.43
    - Class B address + network is 128.2
    - Lookup 128.2 in forwarding table
    - Prefix – part of address that really matters for routing
  - Forwarding table contains
    - List of class+network entries
    - A few fixed prefix lengths (8/16/24)
  - Large tables
    - 2 Million class C networks

Subnet Addressing
RFC917 (1984)

- Class A & B networks too big
  - Very few LANs have close to 64K hosts
  - For electrical/LAN limitations, performance or administrative reasons
- Bridging has scaling limitations
  - What are they?
- Need simple way to get multiple “networks”
  - Multiple IP networks within a single network – often called subnets
  - Networks often follow organization boundaries

Subnetting

- Add another layer to hierarchy
- Variable length subnet masks
  - Could subnet a class B into several chunks
Subnetting Example

- Assume an organization was assigned address 150.100
- Assume < 100 hosts per subnet
- How many host bits do we need?
  » Seven
- What is the network mask?
  » 11111111 11111111 11111111 10000000
  » 255.255.255.128

Forwarding Example

- Assume a packet arrives with address 150.100.12.176
- Step 1: AND address with class + subnet mask

Important Concepts

- Hierarchical addressing critical for scalable system
  » Don't require everyone to know everyone else
  » Forwarding based on prefix
  » Reduces number of updates when something changes

Outline

- Traditional IP addressing
- CIDR IP addressing
- Forwarding examples
- IP packet format

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

IP Address Utilization (’97)

http://www.caida.org/outreach/resources/learn/ipv4space/
Size of Complete Routing Table

Source: www.cidr-report.org
Shows that CIDR has kept # table entries in check
- Currently require 124,894 entries for a complete table
- Only required by backbone routers

Classless Inter-Domain Routing (CIDR) – RFC1338

- 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) → How?
  - Use single entry for range in forwarding tables
  - Combined forwarding entries when possible

CIDR Example

- Network is allocated 8 class C chunks, 200.10.0.0 to 200.10.7.255
  - Allocation uses 3 bits of class C space
  - Remaining 20 bits are network number, written as 201.10.0.0/21
- Replaces 8 class C routing entries with 1 combined entry
  - Routing protocols carry prefix with destination network address
  - Longest prefix match for forwarding

IP Addresses: How to Get One?

Network (network portion): Get allocated portion of ISP’s address space:

| ISP’s block | 11001000 00101111 00010000 00000000 200.23.16.0/20 |
| Organization 0 | 11001010 00101111 00010000 00000000 200.23.16.0/23 |
| Organization 1 | 11001010 00101111 00010100 00000000 200.23.18.0/23 |
| Organization 2 | 11001010 00101111 00010100 00000000 200.23.20.0/23 |
| Organization 7 | 11001010 00101111 00011110 00000000 200.23.30.0/23 |

IP Addresses: How to Get One?

- How does an ISP get block of addresses?
  - From Regional Internet Registries (RIRs)
    - ARIN (North America, Southern Africa), APNIC (Asia-Pacific), RIPE (Europe, Northern Africa), LACNIC (South America)
- How about a single host?
  - Hard-coded by system admin in a file, or
    - Host broadcasts “DHCP discover” msg
    - DHCP server responds with “DHCP offer” msg
    - Host requests IP address: “DHCP request” msg
    - DHCP server sends address: “DHCP ack” msg

CIDR Illustration

Provider is given 201.10.0.0/21
CIDR Implications

- Longest prefix match!!

```
Provider 1

201.10.0.0/24 201.10.4.0/24
201.10.5.0/24
201.10.6.0/23 or Provider 2 address
```

Outline

- Traditional IP addressing
  - CIDR IP addressing
  - Forwarding examples
  - IP packet format

Addressing in IP

- IP addresses are names of interfaces
  - E.g., 128.2.1.1
- Domain Name System (DNS) names are names of hosts
  - E.g., www.cmu.edu
- DNS binds host names to interfaces
- Routing binds interface names to paths

Aside: Interaction with Link Layer

- How does one find the Ethernet address of a IP host?
- ARP: Address Resolution Protocol
  - Broadcast search for IP address
    - E.g., "who-has 128.2.184.45 tell 128.2.206.138" sent to Ethernet broadcast (all FF address)
    - Destination responds (only to requester using unicast) with appropriate 48-bit Ethernet address
      - E.g. "reply 128.2.184.45 is-at 0:d0:bc:f2:18:58" sent to 0:c0:4f:d:ed:c6

Host Routing Table Example

<table>
<thead>
<tr>
<th>Destination</th>
<th>Gateway</th>
<th>Genmask</th>
<th>Iface</th>
</tr>
</thead>
<tbody>
<tr>
<td>128.2.209.100</td>
<td>0.0.0.0</td>
<td>255.255.255.255</td>
<td>eth0</td>
</tr>
<tr>
<td>128.2.0.0</td>
<td>0.0.0.0</td>
<td>255.255.0.0</td>
<td>eth0</td>
</tr>
<tr>
<td>127.0.0.0</td>
<td>0.0.0.0</td>
<td>255.0.0.0</td>
<td>lo</td>
</tr>
<tr>
<td>0.0.0.0</td>
<td>128.2.254.36</td>
<td>0.0.0.0</td>
<td>eth0</td>
</tr>
</tbody>
</table>

- From "netstat –rn"
- Host 128.2.209.100 when plugged into CS ethernet
- Dest 128.2.209.100 → routing to same machine
- Dest 128.2.0.0 → other hosts on same ethernet
- Dest 127.0.0.0 → special loopback address
- Dest 0.0.0.0 → default route to rest of Internet

Routing to the Network

- Packet to 10.1.1.3 arrives at R2 from provider
- Path is R2 – R1 – H1 – H2

```
R1
10.1.1.2
10.1.1.4
R2
10.1.8.4
10.1.8.1
10.1.2.1
10.1.16.1
R1
10.1.1.3
10.1.1.5
R2
10.1.8.5
10.1.8.2
10.1.2.2
10.1.16.2
Provider
10.1.8.3
10.1.2.3
10.1.16.3
10.1.16.4
```

- Main CS router: gigrouter.net.cs.cmu.edu (128.2.254.36)
**Routing Within the Subnet**

- Packet to 10.1.1.3
- Matches 10.1.0.0/23

Routing table at R2

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.1</td>
<td></td>
<td>lo0</td>
</tr>
<tr>
<td>Default or 0</td>
<td>10.1.1.1</td>
<td></td>
</tr>
<tr>
<td>10.1.0.0/24</td>
<td>10.1.0.1</td>
<td>10.1.0.1</td>
</tr>
<tr>
<td>10.1.0.2/23</td>
<td>10.1.0.1</td>
<td>10.1.0.1</td>
</tr>
</tbody>
</table>

- Packet to 10.1.1.3
- Matches 10.1.1.1/31

Routing table at R1

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.1</td>
<td></td>
<td>lo0</td>
</tr>
<tr>
<td>Default or 0</td>
<td>10.1.2.1</td>
<td>10.1.2.1</td>
</tr>
<tr>
<td>10.1.0.0/24</td>
<td>10.1.1.1</td>
<td>10.1.1.1</td>
</tr>
<tr>
<td>10.1.2.0/24</td>
<td>10.1.1.1</td>
<td>10.1.1.1</td>
</tr>
<tr>
<td>10.1.2.1/31</td>
<td>10.1.1.2</td>
<td>10.1.1.2</td>
</tr>
</tbody>
</table>

**Routing Within the Subnet**

- Packet to 10.1.1.3
- Direct route
  - Longest prefix match

Routing table at H1

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.1</td>
<td></td>
<td>lo0</td>
</tr>
<tr>
<td>Default or 0</td>
<td>10.1.1.1</td>
<td>10.1.1.1</td>
</tr>
<tr>
<td>10.1.0.0/24</td>
<td>10.1.1.1</td>
<td>10.1.1.1</td>
</tr>
<tr>
<td>10.1.1.0/24</td>
<td>10.1.1.1</td>
<td>10.1.1.1</td>
</tr>
</tbody>
</table>

**IP Service Model**

- Datagram service model provided by Internet
  - Each packet self-contained
    - All information needed to get to destination
    - No advance setup or connection maintenance
  - Analogous to letter or telegram

**Outline**

- Traditional IP addressing
- CIDR IP addressing
- Forwarding examples
- IP packet format

**IPv4 Header Fields**

- Version: IP Version
  - 4 for IPv4
- HLen: Header Length
  - 32-bit words (typically 5)
- TOS: Type of Service
  - Priority information
- Length: Packet Length
  - Bytes (including header)
- Header format can change with versions
  - First byte identifies version
- Length field limits packets to 65,535 bytes
  - In practice, break into much smaller packets for network performance considerations
IPv4 Header Fields

- **Identifier, flags, fragment offset** - used primarily for fragmentation
- **Time to live**
  - Must be decremented at each router
  - Packets with TTL=0 are thrown away
  - Ensure packets exit the network
- **Protocol**
  - Demultiplexing to higher layer protocols
  - TCP = 6, ICMP = 1, UDP = 17...
- **Header checksum**
  - Ensures some degree of header integrity
  - Relatively weak – 16 bit
- **Options**
  - E.g. Source routing, record route, etc.
  - Performance issues - Poorly supported

IP Delivery Model

- **Best effort service**
  - Network will do its best to get packet to destination
- **Does NOT guarantee**:
  - Any maximum latency or even ultimate success
  - Sender will be informed if packet doesn’t make it
  - Packets will arrive in same order as sent
  - Just one copy of packet will arrive
- **Implications**
  - Scales very well
  - Higher level protocols must make up for shortcomings
    - Reliably delivering ordered sequence of bytes \( \rightarrow \) TCP
  - Some services not feasible
    - Latency or bandwidth guarantees

Fragmentation Related Fields

- **Length**
  - Length of IP fragment
- **Identification**
  - To match up with other fragments
- **Flags**
  - Don’t fragment flag
  - More fragments flag
- **Fragment offset**
  - Where this fragment lies in entire IP datagram
  - Measured in 8 octet units (13 bit field)

IP Fragmentation

- **Every network has own Maximum Transmission Unit (MTU)**
  - Largest IP datagram it can carry within its own packet frame
    - E.g., Ethernet is 1500 bytes
  - Don’t know MTUs of all intermediate networks in advance
- **IP Solution**
  - When hit network with small MTU, fragment packets
IP Fragmentation Example #2

MTU = 2000

IP Length = 3820, M=0

IP Header

IP Data

1820 bytes

IP Data

3800 bytes

IP Header

IP Data

1820 bytes

IP Data

1980 bytes

IP Header

IP Data

1840 bytes

IP Data

520 bytes

IP Fragmentation Example #3

MTU = 1500

IP Length = 2000, M=1, Offset = 0

IP Header

IP Data

1480 bytes

IP Data

1980 bytes

IP Header

IP Data

1500 bytes

IP Data

520 bytes

IP Header

IP Data

1840 bytes

IP Data

360 bytes

IP Reassembly

- Fragments might arrive out-of-order
  - Don’t know how much memory required until receive final fragment
- Some fragments may be duplicated
  - Keep only one copy
- Some fragments may never arrive
  - After a while, give up entire process

Fragmentation and Reassembly Concepts

- Demonstrates many Internet concepts
- Decentralized
  - Every network can choose MTU
- Connectionless
  - Each (fragment of) packet contains full routing information
  - Fragments can proceed independently and along different routes
- Best effort
  - Fail by dropping packet
  - Destination can give up on reassembly
  - No need to signal sender that failure occurred
- Complex endpoints and simple routers
  - Reassembly at endpoints

Fragmentation is Harmful

- Uses resources poorly
  - Forwarding costs per packet
  - Best if we can send large chunks of data
  - Worst case: packet just bigger than MTU
- Poor end-to-end performance
  - Loss of a fragment
- Path MTU discovery protocol → determines minimum MTU along route
  - Uses ICMP error messages
- Common theme in system design
  - Assure correctness by implementing complete protocol
  - Optimize common cases to avoid full complexity

Where to do Reassembly?

- End nodes or at routers?
- End nodes
  - Avoids unnecessary work where large packets are fragmented multiple times
  - If any fragment missing, delete entire packet
- Dangerous to do at intermediate nodes
  - How much buffer space required at routers?
  - What if routes in network change?
    - Multiple paths through network
    - All fragments only required to go through destination