IP Addressing and Forwarding

COS 461: Computer Networks
Spring 2010 (MW 3:00-4:20 in COS 105)

Michael Freedman

http://www.cs.princeton.edu/courses/archive/spring10/cos461/
Goals of Today’s Lecture

• IP addresses
  – Dotted-quad notation
  – IP prefixes for aggregation

• Address allocation
  – Classful addresses
  – Classless InterDomain Routing (CIDR)
  – Growth in the number of prefixes over time

• Packet forwarding
  – Forwarding tables
  – Longest-prefix match forwarding
  – Where forwarding tables come from
IP Address (IPv4)

- A unique 32-bit number
- Identifies an interface (on a host, on a router, ...)
- Represented in dotted-quad notation

```
12 34 158 5
```
```
00001100 00100010 10011110 00000101
```
Grouping Related Hosts

• The Internet is an “inter-network”
  – Used to connect *networks* together, not *hosts*
  – Needs way to address a network (i.e., group of hosts)

LAN = Local Area Network
WAN = Wide Area Network
Scalability Challenge

• Suppose hosts had arbitrary addresses
  – Then every router would need a lot of information
  – ...to know how to direct packets toward every host
Scalability Challenge

• Suppose hosts had arbitrary addresses
  – Then every router would need a lot of information
  – ...to know how to direct packets toward every host

• Back of envelop calculations
  – 32-bit IP address: 4.29 billion \(2^{32}\) possibilities
  – How much storage?
    • Minimum: 4B address + 2B forwarding info per line
    • Total: 24.58 GB just for forwarding table
  – What happens if a network link gets cut?
Standard CS Trick

Have a scalability problem?
Introduce hierarchy...
Hierarchical Addressing in U.S. Mail

• **Addressing in the U.S. mail**
  – Zip code: 08540
  – Street: Olden Street
  – Building on street: 35
  – Room in building: 308
  – Name of occupant: Mike Freedman

• **Forwarding the U.S. mail**
  – Deliver letter to the post office in the zip code
  – Assign letter to mailman covering the street
  – Drop letter into mailbox for the building/room
  – Give letter to the appropriate person
Hierarchical Addressing: IP Prefixes

- IP addresses can be divided into two portions
  - Network (left) and host (right)
- 12.34.158.0/24 is a 24-bit prefix
  - Which covers $2^8$ addresses (e.g., up to 255 hosts)
Expressing IP prefixes

IP prefix = IP address (AND) subnet mask
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-table entry
Address Allocation
Classful Addressing

• In the olden days, only fixed allocation sizes
  – Class A: 0*
    • Very large /8 blocks (e.g., MIT has 18.0.0.0/8)
  – Class B: 10*
    • Large /16 blocks (e.g., Princeton has 128.112.0.0/16)
  – Class C: 110*
    • Small /24 blocks (e.g., AT&T Labs has 192.20.225.0/24)
  – Class D: 1110*
    • Multicast groups
  – Class E: 11110*
    • Reserved for future use

• This is why folks use dotted-quad notation!
Classless Inter-Domain Routing (CIDR)

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

Introduced in 1993
RFC 1518-1519
CIDR: Hierarchal Address Allocation

- Prefixes are key to Internet scalability
  - Address allocated in contiguous chunks (prefixes)
  - Routing protocols and packet forwarding based on prefixes
  - Today, routing tables contain ~200,000 prefixes (vs. 4B)
Scalability: Address Aggregation

Provider is given 201.10.0.0/21

Routers in rest of Internet just need to know how to reach 201.10.0.0/21. Provider can direct IP packets to appropriate customer.
But, Aggregation Not Always Possible

Multi-homed customer (201.10.6.0/23) has two providers. Other parts of the Internet need to know how to reach these destinations through both providers.
Scalability Through Hierarchy

• Hierarchical addressing
  – Critical for scalable system
  – Don’t require everyone to know everyone else
  – Reduces amount of updating when something changes

• Non-uniform hierarchy
  – Useful for heterogeneous networks of different sizes
  – Initial class-based addressing was far too coarse
  – Classless InterDomain Routing (CIDR) helps

• Next few slides
  – History of the number of globally-visible prefixes
  – Plots are # of prefixes vs. time

Growth faster than improvements in equipment capability
CIDR Deployed (1994-1996): Much Flatter

Efforts to aggregate (even decreases after IETF meetings!)

Good use of aggregation, and peer pressure in CIDR report

Internet boom and increased multi-homing
Long-Term View (1989-2005): Post-Boom
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 Corp. for Assigned Names and Numbers (IANA)
    • Allocates large address blocks to Regional Internet Registries
  – Regional Internet Registries (RIRs)
    • E.g., ARIN (American Registry for Internet Numbers)
    • Allocates address blocks within their regions
    • Allocated to Internet Service Providers and large institutions
  – Internet Service Providers (ISPs)
    • Allocate address blocks to their customers
    • Who may, in turn, allocate to their customers...
Figuring Out Who Owns an Address

• Address registries
  – Public record of address allocations
  – Internet Service Providers (ISPs) should update when giving addresses to customers
  – However, records are notoriously out-of-date

• Ways to query
  – UNIX: “whois –h whois.arin.net 128.112.136.35”
  – http://www.arin.net/whois/
  – ...

Example Output for 128.112.136.35

OrgName: Princeton University
OrgID: PRNU
Address: Office of Information Technology
Address: 87 Prospect Avenue
City: Princeton
StateProv: NJ
PostalCode: 08540
Country: US

NetRange: 128.112.0.0 - 128.112.255.255
CIDR: 128.112.0.0/16
NetName: PRINCETON
NetHandle: NET-128-112-0-0-1
Parent: NET-128-0-0-0-0
NetType: Direct Allocation
NameServer: DNS.PRINCETON.EDU
NameServer: NS1.FAST.NET
NameServer: NS2.FAST.NET
NameServer: NS1.UCSC.EDU
NameServer: ARIZONA.EDU
NameServer: NS3.NIC.FR

Comment:
RegDate: 1986-02-24
Updated: 2007-02-27
Are 32-bit Addresses Enough?

• Not all that many unique addresses
  – \(2^{32} = 4,294,967,296\) (just over four billion)
  – Plus, some are reserved for special purposes
  – And, addresses are allocated in larger blocks
    • My fraternity/dorm at MIT had as many IP addr as Princeton!

• And, many devices need IP addresses
  – Computers, PDAs, routers, tanks, toasters, ...

• Long-term solution: a larger address space
  – IPv6 has 128-bit addresses \(2^{128} = 3.403 \times 10^{38}\)

• Short-term solutions: limping along with IPv4
  – Private addresses (RFC 1918):
    • 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
  – Network address translation (NAT)
  – Dynamically-assigned addresses (DHCP)
Hard Policy Questions

• How much address space per geographic region?
  – Equal amount per country?
  – Proportional to the population?
  – What about addresses already allocated?
    • MIT still has >> IP addresses than most countries?

• Address space portability?
  – Keep your address block when you change providers?
  – Pro: avoid having to renumber your equipment
  – Con: reduces the effectiveness of address aggregation

• Keeping the address registries up to date?
  – What about mergers and acquisitions?
  – Delegation of address blocks to customers?
  – As a result, the registries are horribly out of date
Packet Forwarding
Hop-by-Hop Packet Forwarding

• Each router has a forwarding table
  – Maps destination addresses...
  – ... to outgoing interfaces

• Upon receiving a packet
  – Inspect the destination IP address in the header
  – Index into the table
  – Determine the outgoing interface
  – Forward the packet out that interface

• Then, the next router in the path repeats
  – And the packet travels along the path to destination
Separate Table Entries Per Address

- If a router had a forwarding entry per IP addr
  - Match *destination address* of incoming packet
  - ... to the *forwarding-table entry*
  - ... to determine the *outgoing interface*
Separate Entry Per 24-bit Prefix

- If the router had an entry per 24-bit prefix
  - Look only at the top 24 bits of the destination address
  - Index into the table to determine the next-hop interface

<table>
<thead>
<tr>
<th>1.2.3.4</th>
<th>1.2.3.7</th>
<th>1.2.3.156</th>
<th>5.6.7.8</th>
<th>5.6.7.9</th>
<th>5.6.7.212</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>host</td>
<td>...</td>
<td>host</td>
<td>host</td>
<td>...</td>
</tr>
</tbody>
</table>

LAN 1 → router → WAN → router → LAN

<table>
<thead>
<tr>
<th>1.2.3.0/24</th>
<th>5.6.7.0/24</th>
</tr>
</thead>
<tbody>
<tr>
<td>forwarding table</td>
<td></td>
</tr>
</tbody>
</table>
Separate Entry Classful Address

• If the router had an entry per classful prefix
  – Mixture of Class A, B, and C addresses
  – Depends on the first couple of bits of the destination

• Identify the mask automatically from the address
  – First bit of 0: class A address (/8)
  – First two bits of 10: class B address (/16)
  – First three bits of 110: class C address (/24)

• Then, look in the forwarding table for the match
  – E.g., 1.2.3.4 maps to 1.2.3.0/24
  – Then, look up the entry for 1.2.3.0/24
  – ... to identify the outgoing interface

• So far, everything is exact matching
CIDR Makes Packet Forwarding Harder

• There’s no such thing as a free lunch
  – CIDR allows efficient use of limited address space
  – But, CIDR makes packet forwarding much harder

• Forwarding table may have many matches
  – E.g., entries for 201.10.0.0/21 and 201.10.6.0/23
  – The IP address 201.10.6.17 would match both!
Longest Prefix Match Forwarding

• Forwarding tables in IP routers
  – Maps each IP prefix to next-hop link(s)
• Destination-based forwarding
  – Packet has a destination address
  – Router identifies longest-matching prefix
  – Cute algorithmic problem: very fast lookups

destination

201.10.6.17

forwarding table

4.0.0.0/8
4.83.128.0/17
201.10.0.0/21
201.10.6.0/23
126.255.103.0/24

outgoing link

Serial0/0.1

201.10.6.17
Another reason FIBs get large

- If customer 201.10.6.0/23 prefers to receive traffic from Provider 1 (it may be cheaper), then P1 needs to announce 201.10.6.0/23, not 201.10.0.0/21

- Can’t always aggregate! [See “Geographic Locality of IP Prefixes” M. Freedman, M. Vutukuru, N. Feamster, and H. Balakrishnan. Internet Measurement Conference (IMC), 2005]
Simplest Algorithm is Too Slow

• Scan the forwarding table one entry at a time
  – See if the destination matches the entry
  – If so, check the size of the mask for the prefix
  – Keep track of the entry with longest-matching prefix

• Overhead is linear in size of the forwarding table
  – Today, that means 200,000 entries!
  – How much time do you have to process?
    • Consider 10Gbps routers and 64B packets
    • $10,000,000,000 / 8 / 64$: 19,531,250 packets per second
    • 51 nanoseconds per packet

• Need greater efficiency to keep up with line rate
  – Better algorithms
  – Hardware implementations
Patricia Tree (1968)

• **Store the prefixes as a tree**
  – One bit for each level of the tree
  – Some nodes correspond to valid prefixes
  – ... which have next-hop interfaces in a table

• **When a packet arrives**
  – Traverse the tree based on the destination address
  – Stop upon reaching the longest matching prefix
Even Faster Lookups

• Patricia tree is faster than linear scan
  – Proportional to number of bits in the address
• Patricia tree can be made faster
  – Can make a k-ary tree
    • E.g., 4-ary tree with four children (00, 01, 10, and 11)
    – Faster lookup, though requires more space
• Can use special hardware
  – Content Addressable Memories (CAMs)
  – Allows look-ups on a key rather than flat address
• Huge innovations in the mid-to-late 1990s
  – After CIDR was introduced (in 1994)
  – ... and longest-prefix match was a major bottleneck
Where do Forwarding Tables Come From?

• Routers have forwarding tables
  – Map prefix to outgoing link(s)
• Entries can be statically configured
  – E.g., “map 12.34.158.0/24 to Serial0/0.1”
• But, this doesn’t adapt
  – To failures
  – To new equipment
  – To the need to balance load
  – ...
• That is where other technologies come in...
  – Routing protocols, DHCP, and ARP (later in course)
How Do End Hosts Forward Packets?

• End host with single network interface
  – PC with an Ethernet link
  – Laptop with a wireless link
• Don’t need to run a routing protocol
  – Packets to the host itself (e.g., 1.2.3.4/32)
    • Delivered locally
  – Packets to other hosts on the LAN (e.g., 1.2.3.0/24)
    • Sent out the interface: Broadcast medium!
  – Packets to external hosts (e.g., 0.0.0.0/0)
    • Sent out interface to local gateway
• How this information is learned
  – Static setting of address, subnet mask, and gateway
  – Dynamic Host Configuration Protocol (DHCP)
What About Reaching the End Hosts?

• How does the last router reach the destination?

1.2.3.4 1.2.3.7 1.2.3.156

- host
- host
- ... host
- router
- LAN

• Each interface has a persistent, global identifier
  - MAC (Media Access Control) address
  - Burned in to the adaptors Read-Only Memory (ROM)
  - Flat address structure (i.e., no hierarchy)

• Constructing an address resolution table
  - Mapping MAC address to/from IP address
  - Address Resolution Protocol (ARP)
Conclusions

• **IP address**
  – A 32-bit number
  – Allocated in prefixes
  – Non-uniform hierarchy for scalability and flexibility

• **Packet forwarding**
  – Based on IP prefixes
  – Longest-prefix-match forwarding

• **Next lecture**
  – Transmission Control Protocol (TCP)

• **We’ll cover some topics later**
  – Routing protocols, DHCP, and ARP