Network Layer
Where we are in the Course

• Moving on up to the Network Layer!
Network Layer

• How to connect different link layer networks
  • Routing as the primary concern
Why do we need a Network layer?

• We can already build networks with links and switches and send frames between hosts ...
Shortcomings of Switches

1. Don’t scale to large networks
   • Blow up of routing table, broadcast

Table for all destinations in the world!

Broadcast new destinations to the whole world!
Shortcomings of Switches (2)

2. Don’t work across more than one link layer technology
   • Hosts on Ethernet + 3G + 802.11 ...
Shortcomings of Switches (3)

3. Don’t give much traffic control
   • Want to plan routes / bandwidth

That was lame.
Network Layer Approach

• Scaling:
  • Hierarchy, in the form of prefixes

• Heterogeneity:
  • IP for internetworking

• Bandwidth Control:
  • Lowest-cost routing
  • Later QOS (Quality of Service)
Topics

• Network service models
  • Datagrams (packets), virtual circuits

• IP (Internet Protocol)
  • Internetworking
  • Forwarding (Longest Matching Prefix)
  • Helpers: ARP and DHCP
  • Fragmentation and MTU discovery
  • Errors: ICMP (traceroute!)
  • IPv6, scaling IP to the world
  • NAT, and “middleboxes”

• Routing Algorithms
Routing vs. Forwarding

- **Routing** is the process of deciding in which direction to send traffic
  - Network wide (global) and expensive
Routing vs. Forwarding (2)

- **Forwarding** is the process of sending a packet
  - Node process (local) and fast
Networking Services
Topic

• What kind of service does the Network layer provide to the Transport layer?
  • How is it implemented at routers?
Two Network Service Models

• Datagrams, or connectionless service
  • Like postal letters
  • (IP as an example)

• Virtual circuits, or connection-oriented service
  • Like a telephone call
Store-and-Forward Packet Switching

• Both models are implemented with **store-and-forward packet switching**
  • Routers receive a complete packet, storing it temporarily if necessary before forwarding it onwards
  • We use statistical multiplexing to share link bandwidth over time
Store-and-Forward (2)

• Switching element has internal buffering for contention
Store-and-Forward (3)

- Simplified view with per port output buffering
  - Buffer is typically a FIFO (First In First Out) queue
  - If full, packets are discarded (congestion, later)
Datagram Model

• Packets contain a destination address; each router uses it to forward packets, maybe on different paths.
Datagram Model (2)

- Each router has a forwarding table keyed by address
- Gives next hop for each destination address; may change

A’s table (initially)     A’s table (later)    C’s Table          E’s Table

<table>
<thead>
<tr>
<th>Dest.</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>D</td>
<td>B</td>
</tr>
<tr>
<td>E</td>
<td>C</td>
</tr>
<tr>
<td>F</td>
<td>C</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dest.</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>D</td>
<td>B</td>
</tr>
<tr>
<td>E</td>
<td>B</td>
</tr>
<tr>
<td>F</td>
<td>B</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dest.</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>D</td>
<td>E</td>
</tr>
<tr>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>F</td>
<td>E</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dest.</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>B</td>
<td>D</td>
</tr>
<tr>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>
IP (Internet Protocol)

- Network layer of the Internet, uses datagrams (next)
  - IPv4 carries 32-bit addresses, can handle data (often 1.5 KB)
Virtual Circuit Model

• Three phases:
  1. Connection establishment, circuit is set up
     • Path is chosen, circuit information stored in routers
  2. Data transfer, circuit is used
     • Packets are forwarded along the path
  3. Connection teardown, circuit is deleted
     • Circuit information is removed from routers

• Just like a telephone circuit, but virtual in that no bandwidth need be reserved; statistical sharing of links
Virtual Circuits (2)

- Packets contain a short label to identify the circuit
- Labels don’t have global meaning, only unique for a link
Virtual Circuits (4)

- Each router has a forwarding table keyed by circuit
- Gives output line and next label to place on packet

<table>
<thead>
<tr>
<th>Circuit #1</th>
<th>Circuit #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>H1</td>
<td>H3</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A's table</th>
<th>C's Table</th>
<th>E's Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>H1 1</td>
<td>A 5</td>
<td>C 1</td>
</tr>
<tr>
<td>H3 1</td>
<td>A 2</td>
<td>F 1</td>
</tr>
<tr>
<td>C 5</td>
<td>E 1</td>
<td>C 2</td>
</tr>
<tr>
<td>C 2</td>
<td>E 2</td>
<td>F 2</td>
</tr>
<tr>
<td>In</td>
<td>Out</td>
<td>F</td>
</tr>
</tbody>
</table>
MPLS (Multi-Protocol Label Switching, §5.6.5)

• A virtual-circuit like technology widely used by ISPs
  • ISP sets up circuits inside their backbone ahead of time
  • ISP adds MPLS label to IP packet at ingress, undo at egress
# Datagrams vs Virtual Circuits

- Complementary strengths

<table>
<thead>
<tr>
<th>Issue</th>
<th>Datagrams</th>
<th>Virtual Circuits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setup phase</td>
<td>Not needed</td>
<td>Required</td>
</tr>
<tr>
<td>Router state</td>
<td>Per destination</td>
<td>Per connection</td>
</tr>
<tr>
<td>Addresses</td>
<td>Packet carries full address</td>
<td>Packet carries short label</td>
</tr>
<tr>
<td>Routing</td>
<td>Per packet</td>
<td>Per circuit</td>
</tr>
<tr>
<td>Failures</td>
<td>Easier to mask</td>
<td>Difficult to mask</td>
</tr>
<tr>
<td>Quality of service</td>
<td>Difficult to add</td>
<td>Easier to add</td>
</tr>
</tbody>
</table>
Internetworking (IP)
Topic

- How do we connect different networks together?
  - This is called internetworking
  - We’ll look at how IP does it
How Networks May Differ

• Basically, in a lot of ways:
  • Service model (datagrams, VCs)
  • Addressing (what kind)
  • QOS (priorities, no priorities)
  • Packet sizes
  • Security (whether encrypted)

• Internetworking hides the differences with a common protocol. (Uh oh.)
Connecting Datagram and VC networks

• An example to show that it’s not so easy
  • Need to map destination address to a VC and vice-versa
  • A bit of a “road bump”, e.g., might have to set up a VC
Internetworking – Cerf and Kahn

• Pioneers: Cerf and Kahn
  • “Fathers of the Internet”
  • In 1974, later led to TCP/IP
• Tackled the problems of interconnecting networks
  • Instead of mandating a single network technology
Internet Reference Model

• Internet Protocol (IP) is the “narrow waist”
  • Supports many different links below and apps above

1. Link
   Ethernet
   Cable
   DSL
   802.11

2. Internet
   IP

3. Transport
   TCP
   UDP

4. Application
   SMTP
   HTTP
   RTP
   DNS

3G
DSL
Cable
IP as a Lowest Common Denominator

• Suppose only some networks support QOS or security etc.
  • Difficult for internetwork to support
• Pushes IP to be a “lowest common denominator”
  • Asks little of lower-layer networks
  • Gives little as a higher layer service
IPv4 (Internet Protocol)

- Various fields to meet straightforward needs
  - Version, Header (IHL), Total length, Protocol, and Header Checksum

![IPv4 Header Diagram]

Payload (e.g., TCP segment)
IPv4 (2)

• Some fields to handle packet size differences (later)
  • Identification, Fragment offset, Fragment control bits
IPv4 (3)

- Other fields to meet other needs (later, later)
  - Differentiated Services, Time to live (TTL)

Later, with QOS

Later, with ICMP

Payload (e.g., TCP segment)
IPv4 (4)

- Network layer of the Internet, uses datagrams
  - Provides a layer of addressing above link addresses (next)
IP Addresses

• IPv4 uses 32-bit addresses
  • Later we’ll see IPv6, which uses 128-bit addresses

• Written in “dotted quad” notation
  • Four 8-bit numbers separated by dots

\[ \begin{align*}
\text{aaaaaaa} & \text{abbbbbbbbbccccccccccddd\ldots} & \leftrightarrow & \text{A.B.C.D} \\
00010010000111110000000000000001 & \leftrightarrow 
\end{align*} \]
IP Prefixes

• Addresses are allocated in blocks called prefixes
  • Addresses in an L-bit prefix have the same top L bits
  • There are $2^{32-L}$ addresses aligned on $2^{32-L}$ boundary
IP Prefixes (2)

• Written in “IP address/length” notation
  • Address is lowest address in the prefix, length is prefix bits
  • E.g., 128.13.0.0/16 is 128.13.0.0 to 128.13.255.255
  • So a /24 ("slash 24") is 256 addresses, and a /32 is one address

00010010000111110000000000xxxxxxxx ↔

↔ 128.13.0.0/16
Classful IP Addressing

- Originally, IP addresses came in fixed size blocks with the class/size encoded in the high-order bits
  - They still do, but the classes are now ignored

<table>
<thead>
<tr>
<th>Network portion</th>
<th>Host portion</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td></td>
</tr>
</tbody>
</table>

Class A, $2^{24}$ addresses
Class B, $2^{16}$ addresses
Class C, $2^8$ addresses
Classful IP Addressing

• This is an ARPANet assignment.
IP Forwarding

• Addresses on one network belong to a unique prefix
• Node uses a table that lists the next hop for prefixes

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.24.0.0/19</td>
<td>D</td>
</tr>
<tr>
<td>192.24.12.0/22</td>
<td>B</td>
</tr>
</tbody>
</table>
Longest Matching Prefix

• Prefixes in the table might overlap!
  • Combines hierarchy with flexibility

• Longest matching prefix forwarding rule:
  • For each packet, find the longest prefix that contains the destination address, i.e., the most specific entry
  • Forward the packet to the next hop router for that prefix
### Longest Matching Prefix (2)

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.24.0.0/19</td>
<td>D</td>
</tr>
<tr>
<td>192.24.12.0/22</td>
<td>B</td>
</tr>
</tbody>
</table>

192.24.6.0 → ?
192.24.14.32 → ?
192.24.54.0 → ?

![Diagram showing IP addresses and prefixes]

- **PREFIX**: 192.24.0.0/19
- **Next Hop**: D
- **PREFIX**: 192.24.12.0/22
- **Next Hop**: B

**IP Address**
- 192.24.63.255
- 192.24.15.255
- 192.24.0.0

**More specific**
IP Address Work Slide:

• Route to D = 192.00011x.x.x
• Route to B = 192.00011000.000011x.x
• 192.24.6.0 = 192.00011000.00000110.00000000
• 192.24.14.32 = 192.00011000.00001110.00010000
• 192.24.54.0 = 192.00011000.00110110.00000000
Longest Matching Prefix (2)

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.24.0.0/19</td>
<td>D</td>
</tr>
<tr>
<td>192.24.12.0/22</td>
<td>B</td>
</tr>
</tbody>
</table>

192.24.6.0 $\rightarrow$ D
192.24.14.32 $\rightarrow$ B
192.24.54.0 $\rightarrow$ D
Host/Router Distinction

• In the Internet:
  • Routers do the routing, know way to all destinations
  • Hosts send remote traffic (out of prefix) to nearest router

Not for my network? Send it to the gateway (router)

It’s my job to know which way to go...
Host Networking

- Consists of 4 pieces of data:
  - IP Address
  - Subnet Mask
    - Defines local addresses
  - Gateway
    - Who (local) to send non-local packets to for routing
  - DNS Server (Later)
Host Forwarding Table

• Give using longest matching prefix
  • 0.0.0.0/0 is a default route that catches all IP addresses

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>My network prefix</td>
<td>Send to that IP</td>
</tr>
<tr>
<td>0.0.0.0/0</td>
<td>Send to my router</td>
</tr>
</tbody>
</table>
Flexibility of Longest Matching Prefix

• Can provide default behavior, with less specifics
  • Send traffic going outside an organization to a border router (gateway)

• Can special case behavior, with more specifics
  • For performance, economics, security, ...
Performance of Longest Matching Prefix

• Uses hierarchy for a compact table
  • Relies on use of large prefixes

• Lookup more complex than table
  • Used to be a concern for fast routers
  • Not an issue in practice these days
Issues?

• Where does this break down?
1) Form a group of 3
   1.5) Introduce yourselves if you don’t already know each other

2) In your group, answer the following review questions

- Why would we want another set of addresses for internetworking when we already have unique hardware addresses at L2?
  - Related… what makes an IP address different from an L2 (i.e. ethernet MAC) address?

- What new issues do we have to mask at the internetwork layer on top of many links to provide IP’s datagram service?
Where does the IP service model break down?

- Bootstrapping (DHCP)
- Finding Link nodes (ARP)
- Really big packets (Fragmentation)
- Errors in the network (ICMP)
- Running out of addresses (IPv6, NAT)
Dynamic Host Configuration Protocol (DHCP)
Bootstrapping

• Problem:
  • A node wakes up for the first time ...
  • What is its IP address? What’s the IP address of its router?
  • At least Ethernet address is on NIC
Bootstrapping (2)

1. Manual configuration (old days)
   • Can’t be factory set, depends on use
2. DHCP: Automatically configure addresses
   • Shifts burden from users to IT folk
DHCP

• DHCP (Dynamic Host Configuration Protocol), from 1993, widely used
• It leases IP address to nodes
• Provides other parameters too
  • Network prefix
  • Address of local router
  • DNS server, time server, etc.
DHCP Protocol Stack

• DHCP is a client-server application
  • Uses UDP ports 67, 68
DHCP Addressing

• Bootstrap issue:
  • How does node send a message to DHCP server before it is configured?

• Answer:
  • Node sends broadcast messages that delivered to all nodes on the network
  • Broadcast address is all 1s
  • IP (32 bit): 255.255.255.255
  • Ethernet (48 bit): ff:ff:ff:ff:ff:ff
DHCP Messages

Client  

One link  

Server
DHCP Messages (2)

Client

DISCOVER

OFFER

REQUEST

ACK

Server

All Broadcast (255.255.255.255)
DHCP Messages (3)

• To renew an existing lease, an abbreviated sequence is used:
  • REQUEST, followed by ACK

• Protocol also supports replicated servers for reliability
Address Resolution Protocol (ARP)
Sending an IP Packet

• Problem:
  • A node needs Link layer addresses to send a frame over the local link
  • How does it get the destination link address from a destination IP address?
ARP (Address Resolution Protocol)

• Node uses to map a local IP address to its Link layer addresses
### ARP Protocol Stack

- ARP sits on top of link layer (L3→L2 translation)
  - No servers, just asks node with target IP to identify itself
  - Uses broadcast to reach all nodes

| ARP | Ethernet |
What layer is ARP?

Answer:
In the real world life is messy and layers break down… *Not a perfect model*

ARP uses L2 headers and L2 broadcast, but knows about L3 addresses… Maybe a layer 2.5?
ARP Messages

Node — One link — Target
ARP Messages (2)

[Diagram showing a tree with a node and a target, requesting the IP 1.2.3.4, with an ARP who-has message and a reply with MAC address]

```
[root@host ~]# tcpdump -lni any arp &
(sleep 1; arp -d 10.0.0.254; ping -c1 -n 10.0.0.254)
listening on any, link-type LINUX_SLL (Linux cooked), capture size 96 bytes
17:58:02.155495 arp who-has 10.2.1.224 tell 10.2.1.253
17:58:02.317444 arp who-has 10.0.0.96 tell 10.0.0.253
17:58:02.370446 arp who-has 10.3.1.12 tell 10.3.1.61
```

Node

Who has IP 1.2.3.4?

Broadcast

REPLY

I do at 1:2:3:4:5:6
ARP Table

```
# arp -an | grep 10
? (10.241.1.114) at 00:25:90:3e:dc:fc [ether] on vlan241
? (10.252.1.8) at 00:c0:b7:76:ac:19 [ether] on vlan244
? (10.252.1.9) at 00:c0:b7:76:ae:56 [ether] on vlan244
? (10.252.1.6) at 00:c0:b7:74:fb:9a [ether] on vlan244
? (10.241.1.121) at 00:25:90:2c:d4:f7 [ether] on vlan241
[...]```

Or in the modern IPv4/6 world `ip neigh`
Discovery Protocols

• Help nodes find each other
  • There are more of them!
    • E.g., Zeroconf/Bonjour/Avahi/mDNS

• Often involve broadcast
  • Since nodes aren’t introduced
  • Very handy glue
Any discovery questions?
Fragmentation...
Fragmentation

• Problem: How do we connect networks with different maximum packet sizes?
  • Need to split up packets, or discover the largest size to use
Packet Size Problem

• Different networks have different max packet sizes
  • Or MTU (Maximum Transmission Unit)
  • E.g., Ethernet 1.5K, WiFi 2.3K

• Prefer large packets for efficiency (why?)
  • But what size is too large?
  • Difficult as node doesn’t know complete network path
Packet Size Solutions

• Fragmentation (now)
  • Split up large packets in if they are too big to send
  • Classic method, dated

• Discovery (next)
  • Find the largest packet that fits on the network path
  • IP uses today instead of fragmentation
IPv4 Fragmentation

• Routers fragment packets too large to forward
• Receiving host reassembles to reduce load on routers

Fits on first link
IPv4 Fragmentation Fields

- Header fields used to handle packet size differences
  - Identification, Fragment offset, MF/DF control bits
IPv4 Fragmentation Procedure

• Routers split a packet that is too large:
  • Typically break into large pieces
  • Copy IP header to pieces
  • Adjust length on pieces
  • Set offset to indicate position
  • Set MF (More Fragments) on all pieces except last

• Receiving hosts reassembles the pieces:
  • Identification field links pieces together, MF tells receiver when complete
IPv4 Fragmentation (2)

Before
MTU = 2300
ID = 0x12ef
Data Len = 2300
Offset = 0
MF = 0

After
MTU = 1500
ID =
Data Len =
Offset =
MF =

( Ignore length of headers)
IPv4 Fragmentation (3)

Before
MTU = 2300

ID = 0x12ef
Data Len = 2300
Offset = 0
MF = 0

After
MTU = 1500

ID = 0x12ef
Data Len = 1500
Offset = 0
MF = 1

ID = 0x12ef
Data Len = 800
Offset = 1500
MF = 0
IPv4 Fragmentation (4)

• It works!
  • Allows repeated fragmentation

• But fragmentation is undesirable
  • More work for routers, hosts
  • Tends to magnify loss rate
  • Security vulnerabilities too
Path MTU Discovery

• Discover the MTU that will fit
  • So we can avoid fragmentation
  • The method in use today

• Host tests path with large packet
  • Routers provide feedback if too large; they tell host what size would have fit
Path MTU Discovery (2)

MTU=1400 bytes

MTU=1200 bytes

MTU=900 bytes
Path MTU Discovery (3)
Path MTU Discovery (4)

• Process may seem involved
  • But usually quick to find right size
  • MTUs smaller on edges of network
• Path MTU depends on the path and can change
  • Search is ongoing
• Implemented with ICMP (next)
  • Set DF (Don’t Fragment) bit in IP header to get feedback
Questions on fragmentation & MTU???
Internet Control Message Protocol (ICMP)
Topic

• Problem: What happens when something goes wrong during forwarding?
  • Need to be able to find the problem
Internet Control Message Protocol

• ICMP is a companion protocol to IP
  • They are implemented together
  • Sits on top of IP (IP Protocol=1)

• Provides error report and testing
  • Error is at router while forwarding
  • Also testing that hosts can use
ICMP Errors

• When router encounters an error while forwarding:
  • It sends an ICMP error report back to the IP source
  • It discards the problematic packet; host needs to rectify
ICMP Message Format (2)

- Each ICMP message has a Type, Code, and Checksum
- Often carry the start of the offending packet as payload
- Each message is carried in an IP packet
Traceroute

• IP header contains TTL (Time to live) field
  • Decremented every router hop, with ICMP error at zero
  • Protects against forwarding loops
Traceroute (2)

• Traceroute repurposes TTL and ICMP functionality
  • Sends probe packets increasing TTL starting from 1
  • ICMP errors identify routers on the path
What’s another example of a commonly used tool built around ICMP?

```
# ping -4 xkcd.com
PING xkcd.com (151.101.0.67) 56(84) bytes of data.
64 bytes from 151.101.0.67 (151.101.0.67): icmp_seq=1 ttl=53 time=1.12 ms
64 bytes from 151.101.0.67 (151.101.0.67): icmp_seq=2 ttl=53 time=0.934 ms
64 bytes from 151.101.0.67 (151.101.0.67): icmp_seq=3 ttl=53 time=1.15 ms
^C
--- xkcd.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.934/1.066/1.147/0.094 ms
```
**Example ICMP Messages**

(there are others)

<table>
<thead>
<tr>
<th>Name</th>
<th>Type / Code</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dest. Unreachable (Net or Host)</td>
<td>3 / 0 or 1</td>
<td>Lack of connectivity</td>
</tr>
<tr>
<td>Dest. Unreachable (Fragment)</td>
<td>3 / 4</td>
<td>Path MTU Discovery</td>
</tr>
<tr>
<td>Time Exceeded (Transit)</td>
<td>11 / 0</td>
<td>Traceroute</td>
</tr>
<tr>
<td>Echo Request or Reply</td>
<td>8 or 0 / 0</td>
<td>Ping</td>
</tr>
</tbody>
</table>

Testing, not a forwarding error: Host sends Echo Request, and destination responds with an Echo Reply
Any general ICMP questions?
Network Address Translation (NAT)
Problem: Internet Growth

• Many billions of hosts

• And we’re using 32-bit addresses!

Source: Internet Systems Consortium (www.isc.org)
The End of New IPv4 Addresses

• Now running on leftover blocks held by the regional registries; much tighter allocation policies

Exhausted on 4/11 and 9/12!

IANA (All IPs)

Exhausted on 2/11!

ARIN (US, Canada)

APNIC (Asia Pacific)

RIPE (Europe)

LACNIC (Latin Amer.)

AfriNIC (Africa)

ISP

Companies

End of the world? 12/21/12?
Solution 1: Network Address Translation (NAT)

• Basic idea: Map many “Private” IP addresses to one “Public” IP.

• Allocate IPs for private use (192.168.x, 10.x)
Layering Review

• Remember how layering is meant to work?
  • “Routers don’t look beyond the IP header.” Well ...
Middleboxes

• Sit “inside the network” but perform “more than IP” processing on packets to add new functionality
  • NAT box, Firewall / Intrusion Detection System
Middleboxes (2)

- **Advantages**
  - A possible rapid deployment path when no other option
  - Control over many hosts (IT)

- **Disadvantages**
  - Breaking layering interferes with connectivity
    - strange side effects
  - Poor vantage point for many tasks
NAT (Network Address Translation) Box

• NAT box maps an internal IP to an external IP
  • Many internal hosts connected using few external addresses
  • Middlebox that “translates addresses”

• Motivated by IP address scarcity
  • Controversial at first, now accepted
NAT (2)

• Common scenario:
  • Home computers use “private” IP addresses
  • NAT (in AP/firewall) connects home to ISP using a single external IP address
How NAT Works

• Keeps an internal/external translation table
  • Typically uses IP address + TCP port
  • This is address and port translation

<table>
<thead>
<tr>
<th>What host thinks</th>
<th>What ISP thinks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Internal IP:port</strong></td>
<td><strong>External IP : port</strong></td>
</tr>
<tr>
<td>192.168.1.12 : 5523</td>
<td>44.25.80.3 : 1500</td>
</tr>
<tr>
<td>192.168.1.13 : 1234</td>
<td>44.25.80.3 : 1501</td>
</tr>
<tr>
<td>192.168.2.20 : 1234</td>
<td>44.25.80.3 : 1502</td>
</tr>
</tbody>
</table>

• Need ports to make mapping 1-1 since there are fewer external IPs
How NAT Works (2)

- **Internal → External:**
  - Look up and rewrite Source IP/port

<table>
<thead>
<tr>
<th>Internal source</th>
<th>Internal IP:port</th>
<th>External IP : port</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.1.12 : 5523</td>
<td>44.25.80.3 : 1500</td>
<td></td>
</tr>
</tbody>
</table>

- **External destination**
  - IP=X, port=Y

![Diagram of NAT](image-url)
How NAT Works (3)

• External → Internal
  • Look up and rewrite Destination IP/port

<table>
<thead>
<tr>
<th>Internal destination</th>
<th>Internal IP:port</th>
<th>External IP : port</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>192.168.1.12 : 5523</td>
<td>44.25.80.3 : 1500</td>
</tr>
</tbody>
</table>

External source
IP=X, port=Y

Src = Dst =

\[\text{NAT box}\]
How NAT Works (4)

• Need to enter translations in the table for it to work
  • Create external name when host makes a TCP connection

<table>
<thead>
<tr>
<th>Internal source</th>
<th>Internal IP:port</th>
<th>External IP : port</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.1.12 : 5523</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

External destination IP=X, port=Y

![Diagram of NAT process]

Src = Dst =

Src = Dst =

Src = Dst =
NAT Downsides

• Connectivity has been broken!
  • Can only send incoming packets after an outgoing connection is set up
  • Difficult to run servers or peer-to-peer apps (Skype)
• Doesn’t work when there are no connections (UDP)
• Breaks apps that expose their IP addresses (FTP)
NAT Upsides

• Relieves much IP address pressure
  • Many home hosts behind NATs

• Easy to deploy
  • Rapidly, and by you alone

• Useful functionality
  • Firewall, helps with privacy

• Kinks will get worked out eventually
  • “NAT Traversal” for incoming traffic
IPv6
Problem: Internet Growth

• Many billions of hosts

• And we’re using 32-bit addresses!

Source: Internet Systems Consortium (www.isc.org)
IP Version 6 to the Rescue

• Effort started by the IETF in 1994
  • Much larger addresses (128 bits)
  • Many sundry improvements

• Became an IETF standard in 1998
  • Nothing much happened for a decade
  • Hampered by deployment issues, and a lack of adoption incentives
  • Big push ~2011 as exhaustion looms
IPv6

• Features large addresses
  • 128 bits, most of header

• New notation
  • 8 groups of 4 hex digits (16 bits)
  • Omit leading zeros, groups of zeros

Ex: 2001:0db8:0000:0000:0000:ff00:0042:8329
→ 2001:db8::ff00:42:8329
IPv6 (2)

• Lots of other changes
  • Only public addresses
    • No more NAT!
  • Streamlined header processing
    • No checksum (why’s that faster?)
  • Flow label to group of packets
  • IPSec by default
  • Better fit with “advanced” features (mobility, multicasting, security)
IPv6 Stateless Autoconfiguration (SLAAC)

- Replaces DHCP (sorta...)
- Uses ICMPv6
- Process:
  - Send broadcast message
  - Get prefix from router
  - Attach MAC to router Prefix
IPv6 Transition

• The Big Problem:
  • How to deploy IPv6?
  • Fundamentally incompatible with IPv4

• Dozens of approaches proposed
  • Dual stack (speak IPv4 and IPv6)
  • Translators (convert packets)
  • Tunnels (carry IPv6 over IPv4)
Tunneling

• Native IPv6 islands connected via IPv4
  • Tunnel carries IPv6 packets across IPv4 network
Tunneling (2)

• Tunnel acts as a single link across IPv4 network
Tunneling (3)

- Tunnel acts as a single link across IPv4 network
- Difficulty is to set up tunnel endpoints and routing