CS 352
Network Address Translation

CS 352, Lecture 17.1
http://www.cs.rutgers.edu/~sn624/352

Srinivas Narayana
The main function of the network layer is to move packets from one endpoint to another.
Background: The Internet’s growing pains

• Networks had incompatible addressing
  • IPv4 versus other network-layer protocols (X.25)
  • Routable address ranges different across networks

• Entire networks were changing their Internet Service Providers
  • ISPs didn’t have to route directly to internal endpoints, just to the gateway

• IPv4 address exhaustion
  • Insufficient large IP blocks even for large networks
  • Rutgers (AS46) has > 130,000 publicly routable IP addresses
  • IIT Madras (a well-known public university in India, AS141340) has 512

(Source: ipinfo.io)
Network Address Translation

• When a router modifies fields in an IP packet to:
• Enable communication across networks with different (network-layer) addressing formats and address ranges
• Allow a network to change its connectivity to the Internet en masse by modifying the source IP to a (publicly-visible) gateway IP address
• **Masquerade** as an entire network of endpoints using (say) one publicly visible IP address
  • Effect: use fewer IP addresses for more endpoints!
Typical NAT setup

- The gateway’s IP, 138.76.29.7 is publicly visible
- The local endpoint IP addresses in 10.0.0/24 are private
- All datagrams leaving local network have the same source IP as the gateway
Typical NAT setup

That is, for the rest of the Internet, the gateway **masquerades** as a single endpoint representing (hiding) all the private endpoints. The entire network just needs one (or a few) public IP addresses.
The NAT gateway router accomplishes this by using a different transport port for each distinct (transport-level) conversation between the local network and the Internet.
Typical NAT setup

1: host 10.0.0.1 sends datagram to an external host, 128.119.40.186, at port 80

<table>
<thead>
<tr>
<th>Internet-side</th>
<th>Local side</th>
</tr>
</thead>
<tbody>
<tr>
<td>138.76.29.7, 5001</td>
<td>10.0.0.1, 3345</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>S: 128.119.40.186, 80</td>
<td>D: 138.76.29.7, 5001</td>
</tr>
</tbody>
</table>

2: NAT router changes datagram src addr, port from 10.0.0.1, 3345 to 138.76.29.7, 5001, Updates table

3: Reply arrives to dst addr, port 138.76.29.7, 5001

4: NAT gateway changes datagram dest addr, port from 138.76.29.7, 5001 to 10.0.0.1, 3345
Features of IP-masquerading NAT

• Use one or a few public IPs: You don’t need a lot of addresses from your ISP
• Change addresses of devices inside the local network freely, without notifying the rest of the Internet
• Change the public IP address freely independent of network-local endpoints
• Devices inside the local network are not publicly visible, routable, or accessible
• Most IP masquerading NATs block incoming connections originating from the Internet
  • Only way to communicate is if the internal host initiates the conversation
If you’re home, you’re likely behind NAT

• Most access routers (e.g., your home WiFi router) implement network address translation

• You can check this by comparing your local address (visible from `ifconfig`) and your externally-visible IP address (e.g., type “what’s my IP address?” on your browser search bar)
If you’re home, you’re likely behind NAT

```
[flow:352-S20]$ ifconfig en0
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
   ether f0:18:98:1c:fc:36
   inet6 fe80::1036:7dea:82ee:e868%en0 prefixlen 64 secured scopeid 0xa
   inet 192.168.1.151 netmask 0xffffff00 broadcast 192.168.1.255
   nd6 options=201<PERFORMNUD,DAD>
   media: autoselect
   status: active
[flow:352-S20]$  
```

What's my IP address

Your IP address is 74.102.79.209 in New Brunswick, New Jersey, United States (08901)
Limitations of IP-masquerading NATs

• Connection limit due to 16-bit port-number field
  • ~64K total simultaneous connections with a single public IP address

• NAT can be controversial
  • “Routers should only manipulate headers up to the network layer, not modify headers at the transport layer!”

• Application developers must take NAT into account
  • e.g., peer-to-peer applications like Skype

• Purists: address shortage should instead be solved by IPv6
  • (subject of the next module)
CS 352

Internet Protocol: Version 6

CS 352, Lecture 17.2

http://www.cs.rutgers.edu/~sn624/352

Srinivas Narayana
IPv4 address space exhaustion

• The Internet has run out of IPv4 address blocks to allocate

• Yet, demand for more (public) IP addresses is increasing
  • More organizations moving online, new services, more replication
  • More devices: your phone, your watch, your smart refrigerator

• Fundamental issue: 32-bit addresses are not numerous enough

• IP version 6 (IPv6)
IPv6: Main changes from IPv4

- **Large address space:** 128-bit addresses (16 bytes)
  - Allows up to $3.4 \times 10^{38}$ unique addresses

- **Fixed length headers (40 bytes):**
  - Improves the speed of packet processing in routers
  - IPv6 options processing happens through a separate mechanism: using the field corresponding to the upper-layer protocol

- **New control message protocol:** ICMPv6

- **No datagram fragmentation**
  - The ICMPv6 *packet too big* control message informs the source
IPv6: Main changes from IPv4

• **New quality of service bits:** flow label and traffic class
  • Flow label: denotes packets belonging to the same conversation
  • How the field is populated (ie: what exactly belongs to a “flow”) isn’t specified
  • Routers may choose to provide performance guarantees to flows of specific traffic classes (more on this later)

• **No IP checksum:** remove redundant error detection mechanisms
  • Checksums exist already on common transport (TCP/UDP) and link layer (Ethernet) headers
IPv6 datagram format

- Version: 6
- Class and flow label: for traffic differentiation at routers
- Next header: same as the upper-layer protocol in IPv4. Also used to include IPv6 options
- Hop limit: same as TTL in IPv4
IPv6 datagram format

- Version: 6
- Class and flow label: for traffic differentiation at routers
- Next header: same as the upper-layer protocol in IPv4. Also used to include IPv6 options
- Hop limit: same as TTL in IPv4
Can you spot the differences?

<table>
<thead>
<tr>
<th>ver</th>
<th>class</th>
<th>flow label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>payload len</td>
<td>next hdr</td>
<td>hop limit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>source address (128 bits)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>destination address (128 bits)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>data</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

32 bits

<table>
<thead>
<tr>
<th>ver</th>
<th>hdr</th>
<th>type of service</th>
<th>length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16-bit identifier</td>
<td>flags</td>
<td>fragment offset</td>
<td></td>
</tr>
<tr>
<td>time to live</td>
<td>upper protocol</td>
<td>header checksum</td>
<td></td>
</tr>
</tbody>
</table>

32-bit source IP address

32-bit destination IP address

Options (if any)

data
(variable length, typically a TCP or UDP segment)

32 bits
IPv6 addresses

- IPv6 uses IPv4-CIDR-like (classless) addressing
  - $x = 4$-bit hex number
  - Contiguous 0s are compressed: 47CD::A456:0124
- An IPv4-compatible IPv6 address has a prefix of 96 0-bits
  - Example: ::128.64.18.87
- Globally routable unicast addresses must start with bits 001
- CIDR prefixes written the usual way:
  - Example: 2000::/48 can contain $2^{80}$ endpoints
IPv6 adoption

When IP became a mainstream network-layer protocol, IPv4 was baked into router hardware.

~0% of Internet hosts used IPv6 for a long time (about 30 years)

In 2012, Google and a bunch of large orgs decided to support IPv6 irrevocably.

Now, about 1/3rd of Internet hosts (contacting Google) support IPv6.

IPv6 adoption has been trending up.

CS 352
Address Resolution Protocol

CS 352, Lecture 17.3
http://www.cs.rutgers.edu/~sn624/352

Srinivas Narayana
Background: Let’s peek into the link layer

• Each network adapter has a **hardware address** or a **MAC address**
  • E.g., the Wi-Fi adapter on your laptop has one
• Assigned by the manufacturer, not expected to vary over time
  • Think about it as an identifier for the device
• To communicate over a **single link**, a sender needs the destination **hardware** address
• Directory mechanisms like DNS and bootstrapping mechanisms like DHCP provide IP addresses
• Given an IP address, how does an endpoint find the **hardware** address?
Address Resolution Protocol (ARP)

• ARP solves the following problem. Given an IP, find the machine’s hardware address
  • IP $\rightarrow$ MAC resolution

• All endpoints that are looked up are expected to be within the same network

• Hence, address resolution can use broadcast:
  • We don’t need to develop directory mechanisms like DNS
  • Send (ARP) queries to everyone, asking for a MAC given an IP
ARP packet format

- **Hardware type**: link-layer protocol
  - Example: Ethernet (1)
- **Hardware address length**:
  - Example: Ethernet = 6 bytes
- **Protocol Type**: network-layer protocol
  - Example: IPv4 (0x0800)
- **Protocol address length**
  - Example: IPv4 = 4 bytes
- **Operation**:
  - ARP request: 1, reply: 2
- **Sender’s addresses**
- **Address to be resolved (or response)**
ARP operation

Who has 128.195.1.38?
Tell 128.195.1.20

A
Ethernet Address: 05:23:f4:3d:e1:04
IP Address: 128.195.1.20

B
Ethernet Address: 12:04:2c:6e:11:9c
IP Address: 128.195.1.122

C
Ethernet Address: 98:22:ee:f1:90:1a
IP Address: 128.195.1.38

Wants to transmit to 128.195.1.38

Different target IP address: ignore ARP

Matching target IP: send reply

Hardware type: Ethernet
Protocol type: IPv4
Hardware addr length: 6
Protocol addr length: 4
Operation: 2 (reply)
Sender hardware addr: 05:23:f4:3d:e1:04
Sender protocol addr: 128.195.1.20
Target HW addr: 98:22:ee:f1:90:1a
Target protocol addr: 128.195.1.38
Communicating outside the local net?

• Suppose endpoint A wants to communicate with endpoint B that is in a different network

• ARP broadcast outside the local network is too expensive
  • How does one limit the scope of the broadcast? Internet-wide?

• Besides, the hardware address format used by B’s network might be different from that of A’s network!

• ARPs are not meaningful across network boundaries

• Communicating to a network-external endpoint just means sending the packet to the gateway router
  • Host can know that a destination is external using IP addr and netmask
  • Host can talk to the gateway using DHCP (to get IP) and ARP (to get MAC)
Summary of ARP

• A useful mechanism to allow hosts inside a network to communicate:

• ARP protocol helps resolve IP addresses into MAC addresses using a broadcast mechanism

• Communication outside the local network requires ARP-ing for and sending packets to the gateway