IPv6
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)

<table>
<thead>
<tr>
<th>Version</th>
<th>Diff. Serv.</th>
<th>Flow label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Payload length</td>
<td>Next header</td>
</tr>
</tbody>
</table>

32 bits

Source address (16 bytes)

Destination address (16 bytes)
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
Network Layer (Routing)
Recap: Why do we need a Network layer?

• Internetworking
  • Need to connect different link layer networks

• Addressing
  • Need a globally unique way to “address” hosts

• Routing and forwarding
  • Need to find and traverse paths between hosts
Recap: Routing versus Forwarding

• **Forwarding** is the process of sending a packet on its way

  ![Forwarding Diagram]

• **Routing** is the process of deciding in which direction to send traffic

  ![Routing Diagram]
Overview of Internet Routing and Forwarding

- Hosts on same network have IPs in the same IP prefix
- Hosts send off-network traffic to the gateway router

- Routers discover routes to different prefixes (routing)
- Routers use longest prefix matching to send packets to the right next hop (forwarding)
Longest Prefix Matching

• Prefixes in the forwarding table can overlap

• **Longest prefix matching** 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

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0.0.0/0</td>
<td>A</td>
</tr>
<tr>
<td>192.24.0.0/19</td>
<td>B</td>
</tr>
<tr>
<td>192.24.12.0/22</td>
<td>C</td>
</tr>
</tbody>
</table>
Longest Prefix Matching (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 → ?
Flexibility of Longest Prefix Matching

• 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 Prefix Matching

• 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
# Goals of Routing Algorithms

• **We want several properties of any routing scheme:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correctness</td>
<td>Finds paths that work</td>
</tr>
<tr>
<td>Efficient paths</td>
<td>Uses network bandwidth well</td>
</tr>
<tr>
<td>Fair paths</td>
<td>Doesn’t starve any nodes</td>
</tr>
<tr>
<td>Fast convergence</td>
<td>Recovers quickly after changes</td>
</tr>
<tr>
<td>Scalability</td>
<td>Works well as network grows large</td>
</tr>
</tbody>
</table>
Rules of Fully Distributed Routing

• All nodes are alike; no controller
• Nodes learn by exchanging messages with neighbors
• Nodes operate concurrently
• There may be node/link/message failures
Simple routing that obeys the rules

• Send out routes for hosts you have paths to
  • And the routes they’ve sent you

• This works
  • All routers find a path to all hosts
• But scales poorly!
Recall: Internet Size

- Over 4 billion people
- 50B devices connect
Impact of Network Growth

1. Forwarding tables grow
   • Larger router memories, may increase lookup time
2. Routing messages grow
   • Need to keeps all nodes informed of larger topology
3. Routing computation grows
   • Shortest path calculations grow faster than the network
Techniques to Scale Routing

• First: Network hierarchy
  • Route to network regions

• Next: IP prefix aggregation
  • Combine, and split, prefixes
Scaling Idea 1: Hierarchical Routing
Idea

• Scale routing using hierarchy with regions
  • Route to regions, not individual nodes
Hierarchical Routing

• Introduce a larger routing unit
  • IP prefix (hosts) ← from one host
  • Region, e.g., ISP network

• Route first to the region, then to the IP prefix within the region
  • Hide details within a region from outside of the region
Hierarchical Routing (2)

Full table for 1A

<table>
<thead>
<tr>
<th>Dest.</th>
<th>Line</th>
<th>Hops</th>
</tr>
</thead>
<tbody>
<tr>
<td>1A</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1B</td>
<td>1B</td>
<td>1</td>
</tr>
<tr>
<td>1C</td>
<td>1C</td>
<td>1</td>
</tr>
<tr>
<td>2A</td>
<td>1B</td>
<td>2</td>
</tr>
<tr>
<td>2B</td>
<td>1B</td>
<td>3</td>
</tr>
<tr>
<td>2C</td>
<td>1B</td>
<td>3</td>
</tr>
<tr>
<td>2D</td>
<td>1B</td>
<td>4</td>
</tr>
<tr>
<td>3A</td>
<td>1C</td>
<td>3</td>
</tr>
<tr>
<td>3B</td>
<td>1C</td>
<td>2</td>
</tr>
<tr>
<td>4A</td>
<td>1C</td>
<td>3</td>
</tr>
<tr>
<td>4B</td>
<td>1C</td>
<td>4</td>
</tr>
<tr>
<td>4C</td>
<td>1C</td>
<td>4</td>
</tr>
<tr>
<td>5A</td>
<td>1C</td>
<td>4</td>
</tr>
<tr>
<td>5B</td>
<td>1C</td>
<td>5</td>
</tr>
<tr>
<td>5C</td>
<td>1B</td>
<td>5</td>
</tr>
<tr>
<td>5D</td>
<td>1C</td>
<td>6</td>
</tr>
<tr>
<td>5E</td>
<td>1C</td>
<td>5</td>
</tr>
</tbody>
</table>

Hierarchical table for 1A

<table>
<thead>
<tr>
<th>Dest.</th>
<th>Line</th>
<th>Hops</th>
</tr>
</thead>
<tbody>
<tr>
<td>1A</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1B</td>
<td>1B</td>
<td>1</td>
</tr>
<tr>
<td>1C</td>
<td>1B</td>
<td>1</td>
</tr>
<tr>
<td>1C</td>
<td>1C</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1B</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>1C</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>1C</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>1C</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>1C</td>
<td>4</td>
</tr>
</tbody>
</table>
Hierarchical Routing (3)
Hierarchical Routing (4)

• Penalty is longer paths

1C is best route to region 5, except for destination 5C
Observations

• Outside a region, nodes have one route to all hosts within the region
  • This gives savings in table size, messages and computation

• However, each node may have a different route to an outside region
  • Routing decisions are still made by individual nodes; there is no single decision made by a region
Scaling Idea 2: IP Prefix Aggregation and Subnets
Idea

• Scale routing by adjusting the size of IP prefixes
  • Split (subnets) and join (aggregation)
Recall

• IP addresses are allocated in blocks called IP prefixes, e.g., 18.31.0.0/16
  • Hosts on one network in same prefix
• “/N” prefix has the first N bits fixed and contains $2^{32-N}$ addresses
  • E.g., a “/24” has 256 addresses
• Routers keep track of prefix lengths
  • Use it as part of longest prefix matching

Routers can change prefix lengths without affecting hosts
Prefixes and Hierarchy

• IP prefixes help to scale routing, but can go further
  • Use a less specific (larger) IP prefix as a name for a region

<table>
<thead>
<tr>
<th></th>
<th>IP1 /19</th>
<th>IP2 /18</th>
<th>IP3 /17</th>
</tr>
</thead>
<tbody>
<tr>
<td>Region</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Subnets and Aggregation

• Two use cases for adjusting the size of IP prefixes; both reduce routing table

1. Subnets
   • Internally split one large prefix into multiple smaller ones

2. Aggregation
   • Join multiple smaller prefixes into one large prefix
Subnets

• Internally split up one IP prefix

Internally:

- EE: 16K addresses (128.208.0.0/18)
- CS: 8K addresses (128.208.128.0/17)
- Art: 4K addresses (128.208.96.0/19)

External:

- One prefix sent to rest of Internet (128.208.0.0/16)

Note: The diagram shows the company's network split into subnets with different address ranges, and one prefix sent to the rest of the Internet.
Aggregation

- Externally join multiple separate IP prefixes

One prefix sent to rest of Internet

Rest of Internet   ISP
Routing Process

1. Ship these prefixes or regions around to nearby routers
2. Receive multiple prefixes and the paths of how you got them
3. Build a global routing table
Internet Routing Growth

Growth of the BGP Table - 1994 to Present

Source: bgp.potaroo.net