Basic IPv6

Luc De Ghein

Escalation, EMEA
Background
Short History of IPv6

1991  ROAD group formed to address routing.
1993  IPng Proposals solicitation (RFC 1550).
1994  CATNIP, SIPP, TUBA analyzed. SIPP+ chosen. IPng wg started.
1996  6bone started.
1997  First attempt for provider-based address format.
2001  Vendor Support in Main Product Lines
Why IPv6?  
(Current Business Reasons)

- Demand from particular regions
  Asia, EU
  Technical, geo-political, and business reasons
  Demand is now
- Demand for particular services
  Cellular wireless (especially 3GPP[2] standards)
  Internet gaming (e.g., Sony Playstation 2)
  Use is >= 1.5 years away (but testbeds needed now)
- Potential move to IPv6 by Microsoft?
  IPv6 included in Windows XP, but not enabled by default
  To be enabled by default in next major Windows release
  Use is >= 1.5 years away
Why IPv6?  
(Theoretical Reasons)

Only compelling reason: **more IP addresses!**

- For billions of new users  
  (Japan, China, India,…)

- For billions of new devices  
  (mobile phones, cars, appliances,…)

- For always-on access  
  (cable, xDSL, ethernet-to-the-home,…)

- For applications that are difficult, expensive, or impossible to operate through NATs  
  (IP telephony, peer-to-peer gaming, home servers,…)

- To phase out NATs to improve robustness, security, performance, and manageability of the Internet
Disadvantages of NAT

• performance hit -> rewriting src addresses, recalculating transport checksums, searching for and rewriting application layer addresses (eg. FTP)
  • end - 2 - end significance of IP address is lost

• doesn’t work if higher layer protocol is carrying encrypted address(es)

• inhibits using more than 1 site entry/exit point

• complicates problem diagnosis ( where is a pkt going to / coming from)
Why a New IP?

only compelling reason:

• for billions of new devices, 
  e.g., cell phones, PDAs, appliances, cars, etc.

for billions of new users, 
  e.g., in China, India, etc.

for “always-on” access technologies, 
  e.g., xDSL, cable, ethernet-to-the-home, etc.
But Isn’t There Still Lots of IPv4 Address Space Left?

- ~ half the IPv4 space is unallocated
  if size of Internet is doubling each year, does this mean only one year’s worth?!
- no, because today we deny unique IPv4 addresses to most new hosts
  we make them use methods like NAT, PPP, etc. to share addresses
- but new types of applications and new types of access need unique addresses!
IP Address Allocation History

1981 - IPv4 protocol published
1985 ~ 1/16 of total space
1990 ~ 1/8 of total space
1995 ~ 1/4 of total space
2000 ~ 1/2 of total space

• This despite increasingly intense conservation efforts
  PPP / DHCP address sharing
  CIDR (classless inter-domain routing)
  NAT (network address translation)
  plus some address reclamation

• Theoretical limit of 32-bit space: ~4 billion devices
  Practical limit of 32-bit space: ~250 million devices
  (see draft-durand-huijtema-h-density-ratio)
Incidental Benefits of Bigger Addresses

- easy address auto-configuration
- easier address management/delegation
- room for more levels of hierarchy, for route aggregation
- ability to do end-to-end IPsec (because NATs not needed)
Incidental Benefits of New Deployment

- chance to eliminate some complexity, e.g., in IP header
- chance to upgrade functionality, e.g., multicast, QoS, mobility
- chance to include new enabling features, e.g., binding updates
Summary of Main IPv6 Benefits

• expanded addressing capabilities
• server-less autoconfiguration ("plug-n-play") and reconfiguration
• more efficient and robust mobility mechanisms
• built-in, strong IP-layer encryption and authentication
• streamlined header format and flow identification
• improved support for options / extensions
Other Benefits of IPv6

- Server-less plug-and-play possible
- End-to-end, IP-layer authentication & encryption possible
- Elimination of “triangle routing” for mobile IP
- Other minor improvements

**NON-Benefits:**
- Quality-of-service (same QoS capabilities as IPv4)
  Flow label field in IPv6 header may enable more efficient flow classification by routers, but adds no new capability
- Routing (same routing protocols as IPv4)
  except larger address allows more levels of hierarchy
  except customer multihoming is defeating hierarchy
Why Was 128 Bits Chosen as the IPv6 Address Size?

some wanted fixed-length, 64-bit addresses

- easily good for $10^{12}$ sites, $10^{15}$ nodes, at .0001 allocation efficiency (3 orders of mag. more than IPng requirement)
- minimizes growth of per-packet header overhead
- efficient for software processing

some wanted variable-length, up to 160 bits

- compatible with OSI NSAP addressing plans
- big enough for autoconfiguration using IEEE 802 addresses
- could start with addresses shorter than 64 bits & grow later

settled on fixed-length, 128-bit addresses

($340,282,366,920,938,463,463,374,607,431,768,211,456$ in all!)
## What Ever Happened to IPv5?

<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0–3</td>
<td>unassigned</td>
</tr>
<tr>
<td>4</td>
<td>IPv4 (today’s widespread version of IP)</td>
</tr>
<tr>
<td>5</td>
<td>ST (Stream Protocol, not a new IP)</td>
</tr>
<tr>
<td>6</td>
<td>IPv6 (formerly SIP, SIPP)</td>
</tr>
<tr>
<td>7</td>
<td>CATNIP (formerly IPv7, TP/IX; deprecated)</td>
</tr>
<tr>
<td>8</td>
<td>Pip (deprecated)</td>
</tr>
<tr>
<td>9</td>
<td>TUBA (deprecated)</td>
</tr>
<tr>
<td>10-15</td>
<td>unassigned</td>
</tr>
</tbody>
</table>
Header Formats
The IPv6 Header

<table>
<thead>
<tr>
<th>Version</th>
<th>Traffic Class</th>
<th>Flow Label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Payload Length</th>
<th>Next Header</th>
<th>Hop Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Source Address

Destination Address

32 bits
The IPv4 Header

<table>
<thead>
<tr>
<th>Version</th>
<th>Hdr Len</th>
<th>Prec</th>
<th>TOS</th>
<th>Total Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Identification</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Flags</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Fragment Offset</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Protocol</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Header Checksum</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Source Address</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Destination Address</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Options</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Padding</td>
</tr>
</tbody>
</table>

shaded fields are absent from IPv6 header

32 bits
IPv6 Header compared to IPv4 Header

IPv6 header is twice as long (40 bytes) as IPv4 header without options (20 bytes)
Summary of Header Changes from IPv4

- addresses increased from 32 to 128 bits
- fragmentation fields moved out of base header
- IP options moved out of base header
- Header Checksum eliminated
- Header Length field eliminated
- Flow Label field added
- Time to Live —> Hop Limit
- Protocol —> Next Header
- Precedence & TOS —> Traffic Class
- Length field excludes IPv6 header
- alignment changed from 32 to 64 bits
IPv6 Header Options (RFC 2460)

- Processed only by node identified in IPv6 Destination Address field => much lower overhead than IPv4 options
  - exception: Hop-by-Hop Options header
- Eliminated IPv4’s 40-octet limit on options
  - in IPv6, limit is total packet size, or Path MTU in some cases
### Routing Header

<table>
<thead>
<tr>
<th>Next Header</th>
<th>Hdr Ext Len</th>
<th>Routing Type</th>
<th>Segments Left</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Address[0]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Address[1]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example of Using the Routing Header

Diagram showing a network with nodes S, A, B, and D connected by lines, representing the routing header in a network.
Example of Using the Routing Header

Diagram showing a network with nodes S, A, B, and D, illustrating the routing header.
Example of Using the Routing Header
Example of Using the Routing Header
Extension Headers (cont.)

- processed only by node identified in IPv6 Destination Address field => much lower overhead than IPv4 options
  - exception: Hop-by-Hop Options header
- eliminated IPv4’s 40-byte limit on options
  - in IPv6, limit is total packet size, or Path MTU in some cases
- currently defined extension headers:
  - Hop-by-Hop Options, Routing, Fragment, Authentication, Encryption, Destination Options
are containers for variable-length options:
Option Type Encoding

<table>
<thead>
<tr>
<th>AIU</th>
<th>C</th>
<th>Option ID</th>
</tr>
</thead>
</table>

AIU — action if unrecognized:

- 00 — skip over option
- 01 — discard packet
- 10 — discard packet & send ICMP Unrecognized Type to source
- 11 — discard packet & send ICMP Unrecognized Type to source only if destination was not multicast

C — set if Option Data changes en-route (Hop-by-Hop Options only)
Option Alignment and Padding

two padding options:

Pad1 0 <— special case: no Length or Data fields

PadN 1 N - 2 N-2 zero octets...

• used to align options so multi-byte data fields fall on natural boundaries

• used to pad out containing header to an integer multiple of 8 bytes
## IPv6 Technology Scope

<table>
<thead>
<tr>
<th>IP Service</th>
<th>IPv4 Solution</th>
<th>IPv6 Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addressing Range</td>
<td>32-bit, Network Address Translation</td>
<td>128-bit, NAT-PT</td>
</tr>
<tr>
<td>Autoconfiguration</td>
<td>DHCP, ZeroConf</td>
<td>Serverless, ZeroConf, Reconfiguration, DHCP</td>
</tr>
<tr>
<td>Security</td>
<td>IPSec</td>
<td>IPSec Mandated, works End-to-End</td>
</tr>
<tr>
<td>Mobility</td>
<td>Mobile IP</td>
<td>Mobile IP with Direct Routing</td>
</tr>
<tr>
<td>IP Multicast</td>
<td>IGMP/PIM/Multicast BGP</td>
<td>MLD/PIM/Multicast BGP, Scope Identifier</td>
</tr>
</tbody>
</table>
IPv6 Deployment
IPv6 @ Cisco

- Co-chair of IETF IPv6 Working Group and NGtrans Working Group
- Well Known Cisco 6Bone router
  ~ 70 tunnels with other companies
  acts as 6to4 Relay
  Official Cisco IPv6 prefix registered to ARIN (2001:0420::/35)
- ‘Founding Member’ of the IPv6 Forum
- Official CCO IPv6 page is www.cisco.com/ipv6
  Cisco IPv6 Statement of Direction published last June
  Cisco IOS IPv6 EFT available for free over 3 years
  ~around 500 sites running Worldwide
- Cisco IOS 12.2(2)T offers official IPv6 support
  including Cisco IOS IPv6 training & Worldwide TAC
IPv4-IPv6 Transition / Co-Existence

A wide range of techniques have been identified and implemented, basically falling into three categories:

(1) **Dual-stack** techniques, to allow IPv4 and IPv6 to co-exist in the same devices and networks

(2) **Tunneling** techniques, to avoid order dependencies when upgrading hosts, routers, or regions

(3) **Translation** techniques, to allow IPv6-only devices to communicate with IPv4-only devices

Expect all of these to be used, in combination
Deployment

• Experimental infrastructure: **the 6bone**
  for testing and debugging IPv6 protocols and operations
  (see www.6bone.net)

• Production infrastructure in support of education and research: **the 6ren**
  CAIRN, Canarie, CERNET, Chunahwa Telecom, Dante, ESnet, Internet 2, IPFNET, NTT, Renater, Singren, Sprint, SURFnet, vBNS, WIDE,…
  (see www.6ren.net, www.6tap.net)

• Commercial infrastructure
  a few ISPs (IIJ, NTT,...) are providing commercial IPv6 service today, others are conducting trials
Deployment (cont.)

- IPv6 address allocation
  6bone procedure for test address space
  Regional IP address registries (APNIC, ARIN, RIPE-NCC) for production address space

- Deployment advocacy
  (a.k.a. marketing)
  IPv6 Forum: www.ipv6forum.com
IPv6 Timeline
(A pragmatic projection)

- Early adopter

- Application porting <= Duration 3+ years =>

- ISP adoption <= Duration 3+ years =>

- Consumer adoption <= Duration 5+ years =>

- Enterprise adoption <= Duration 3+ years =>
### IPv6 Timeline
(A pragmatic projection)

#### Asia

<table>
<thead>
<tr>
<th>Year</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2001</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2002</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2003</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2004</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2006</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2007</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Early adopter**
- **Application porting** <= Duration 3+ years =>
- **ISP adoption** <= Duration 3+ years >
- **Consumer adoption** <= Duration 5+ years =>
- **Enterprise adoption** <= Duration 3+ years =>
IPv6 Timeline
(A pragmatic projection)

Europe

<table>
<thead>
<tr>
<th>Year</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>Q1</td>
<td>Q2</td>
<td>Q3</td>
<td>Q4</td>
</tr>
<tr>
<td>2001</td>
<td>Q1</td>
<td>Q2</td>
<td>Q3</td>
<td>Q4</td>
</tr>
<tr>
<td>2002</td>
<td>Q1</td>
<td>Q2</td>
<td>Q3</td>
<td>Q4</td>
</tr>
<tr>
<td>2003</td>
<td>Q1</td>
<td>Q2</td>
<td>Q3</td>
<td>Q4</td>
</tr>
<tr>
<td>2004</td>
<td>Q1</td>
<td>Q2</td>
<td>Q3</td>
<td>Q4</td>
</tr>
<tr>
<td>2005</td>
<td>Q1</td>
<td>Q2</td>
<td>Q3</td>
<td>Q4</td>
</tr>
<tr>
<td>2006</td>
<td>Q1</td>
<td>Q2</td>
<td>Q3</td>
<td>Q4</td>
</tr>
<tr>
<td>2007</td>
<td>Q1</td>
<td>Q2</td>
<td>Q3</td>
<td>Q4</td>
</tr>
</tbody>
</table>

- Early adopter
- Application porting $\leq$ Duration 3+ yr $\Rightarrow$
- ISP adoption $\leq$ Duration 3+ years $\Rightarrow$
- Consumer adoption $\leq$ Duration 5+ yr $\Rightarrow$
- Enterprise adoption $\leq$ Duration 3+ years $\Rightarrow$

Consumer adoption $\leq$ Duration 5+ yrs $\Rightarrow$

Early adopter $\Rightarrow$ Duration 3+ yrs

Application porting $\leq$ Duration 3+ years $\Rightarrow$

ISP adoption $\leq$ Duration 3+ years $\Rightarrow$

Consumer adoption $\leq$ Duration 5+ years $\Rightarrow$

Enterprise adoption $\leq$ Duration 3+ years $\Rightarrow$
IPv6 Timeline
(A pragmatic projection)

Americas

- Early adopter
- Application porting <= Duration 3+ yrs =>
- ISP adoption <= Duration 3+ years >
- Consumer adoption <= Duration 5+ yrs =>
- Enterprise adoption <= Duration 3+ years =>

2000 2001 2002 2003 2004 2005 2006 2007
Q 1 Q 1 Q 2 Q 3 Q 4 Q 1 Q 2 Q 3 Q 4 Q 1 Q 2 Q 3 Q 4 Q 1 Q 2 Q 3 Q 4 Q 1 Q 2 Q 3 Q 4
## Cisco IOS IPv6 Roadmap

<table>
<thead>
<tr>
<th>IOS Release</th>
<th>Market Target</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Phase I</strong></td>
<td>Early Adopter Deployment</td>
</tr>
<tr>
<td>IOS 12.2(2)T</td>
<td></td>
</tr>
<tr>
<td>Done</td>
<td></td>
</tr>
<tr>
<td><strong>Phase II</strong></td>
<td>Production Backbone Deployment</td>
</tr>
<tr>
<td>H2 CY 2001</td>
<td></td>
</tr>
<tr>
<td><strong>Phase III</strong></td>
<td>Enhanced IPv6 Services</td>
</tr>
<tr>
<td>CY 2002</td>
<td></td>
</tr>
</tbody>
</table>

**Cisco IOS IPv6 Roadmap**

**IOS upgrade = Free IPv6**
## Cisco IOS IPv6 Roadmap

<table>
<thead>
<tr>
<th>IOS Release</th>
<th>IPv6 Features Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Phase I</strong></td>
<td>Basic IPv6 specifications support</td>
</tr>
<tr>
<td>IOS 12.2(2)T</td>
<td>Multi-protocol Extensions for BGP4, RIPv6</td>
</tr>
<tr>
<td>H2 CY2001</td>
<td>Manual, Automatic &amp; 6to4 Tunnel Support</td>
</tr>
<tr>
<td></td>
<td>Tools such as Ping, Traceroute, etc</td>
</tr>
<tr>
<td><strong>Phase II</strong></td>
<td>Enhanced Performance (CEFv6/dCEFv6),</td>
</tr>
<tr>
<td>CY 2002</td>
<td>Link State IGP (I/IS-ISv6), IPv6 Edge router (6PE) over MPLS, Dial, NAT-PT, Enhanced</td>
</tr>
<tr>
<td></td>
<td>tools (SSH, DNS client, MIB, etc)</td>
</tr>
<tr>
<td><strong>Phase III</strong></td>
<td>Hardware Acceleration, OSPFv3, Mobility,</td>
</tr>
<tr>
<td>CY 2002</td>
<td>Multicast, Security, QoS…</td>
</tr>
</tbody>
</table>
IPv6 - Addressing
Some Terminology

node         a protocol module that implements IPv6
router       a node that forwards IPv6 packets not explicitly addressed to itself
host         any node that is not a router
link         a communication facility or medium over which nodes can communicate at the link layer, i.e., the layer immediately below IPv6
neighbors    nodes attached to the same link
interface    a node’s attachment to a link
address      an IPv6-layer identifier for an interface or a set of interfaces
Text Representation of Addresses

“preferred” form: 1080:0:FF:0:8:800:200C:417A

compressed form: FF01:0:0:0:0:0:0:43
becomes FF01::43

IPv4-compatible: 0:0:0:0:0:0:13.1.68.3
or ::13.1.68.3
IPv6 - Addressing Model

- Addresses are assigned to interfaces
  - No change from IPv4 Model
- Interface ‘expected’ to have multiple addresses

- Addresses have scope
  - Link Local
  - Site Local
  - Global

- Addresses have lifetime
  - Valid and Preferred lifetime
Address Types

unicast (one-to-one)

- global
- link-local
- site-local
- IPv4-compatible

multicast (one-to-many)
Address of a set of interfaces
Delivery to all interfaces in the set

anycast (one-to-nearest)

Unicast Address of a set of interfaces
Subnet router anycast address: subnet prefix::/n
Delivery to a single interface in the set

Reserved

No more broadcast address
# Address Type Prefixes

<table>
<thead>
<tr>
<th>address type</th>
<th>binary prefix</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv4-compatible</td>
<td>0000...0 (96 zero bits)</td>
</tr>
<tr>
<td>global unicast</td>
<td>001</td>
</tr>
<tr>
<td>link-local unicast</td>
<td>1111 1110 10</td>
</tr>
<tr>
<td>site-local unicast</td>
<td>1111 1110 11</td>
</tr>
<tr>
<td>multicast</td>
<td>1111 1111</td>
</tr>
</tbody>
</table>

- all other prefixes reserved (approx. 7/8ths of total)
- anycast addresses allocated from unicast prefixes
Global Unicast Addresses (RFC 2374)

<table>
<thead>
<tr>
<th>001</th>
<th>TLA</th>
<th>NLA*</th>
<th>SLA*</th>
<th>interface ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>public topology</td>
<td>(45 bits)</td>
<td>site topology</td>
<td>(16 bits)</td>
</tr>
</tbody>
</table>

- TLA = Top-Level Aggregator
- NLA* = Next-Level Aggregator(s)
- SLA* = Site-Level Aggregator(s)

- All subfields variable-length, non-self-encoding (like CIDR)
- TLAs may be assigned to Providers or Exchanges
Global Unicast Addresses for the 6Bone

- 6Bone: experimental IPv6 network used for testing only
- TLA 1FFE (hex) assigned to the 6Bone
  thus, 6Bone addresses start with 3FFE:
  (binary 001 + 1 1111 1111 1110)
- next 12 bits hold a “pseudo-TLA” (pTLA)
  thus, each 6Bone pseudo-ISP gets a /28 prefix
- not to be used for production IPv6 service
The 6Bone uses the 3ffe::/16 range:

- A pTLA receives a /28 prefix
- A site receives a /48 prefix
- A LAN receives a /64 prefix
Initial Address Allocation

- The allocation process is:
  - IANA allocates 2001::/16 to registries
  - Each registry gets a /23 prefix from IANA
  - Registry allocates a /35 prefix to a new IPv6 ISP
  - Policy is that an ISP allocates a /48 prefix to each customer
Hierarchical Addressing & Aggregation

Larger address space enables:

- Aggregation of prefixes announced in the global routing table.
- Efficient and scalable routing.

But current Multi-Homing schemes break the model.

Customer no 1
2001:0410:0001::/48

Customer no 2
2001:0410:0002::/48

ISP
2001:0410::/35

IPv6 Internet
2001::/16

Only announces the /35 prefix

Only announces the /35 prefix
Global Unicast Addresses for Production Service

- ISPs start with less space than a TLA; must demonstrate need before getting a TLA ("slow-start" procedure)
- TLA 1 assigned for slow-start allocations
  - thus, initial production addresses start with 2001:
    - (binary 001 + 0 0000 0000 0001)
- next 13 bits hold a subTLA
  - thus, each new ISP gets a /29 prefix
    (or even longer, depending on registry policy)
Address Allocation – New Proposal

Immediate need

Site prefix

LAN prefix

IXP prefix (not announced to peering ISPs)

Grow on HD ratio

Initial LIR

HD Ratio = \[
\frac{\log (\text{Number of allocated objects})}{\log (\text{Max number of allocatable objects})}
\]

Desirable HD Ratio = 0.8 .. 0.85

0.8 picked for new proposal
Link-Local & Site-Local Unicast Addresses

link-local addresses for use during auto-configuration and when no routers are present:

\[
\begin{array}{c|c}
111111010 & 0 \\ \hline
\text{interface ID} & \\
\end{array}
\]

site-local addresses for independence from changes of TLA / NLA*:

\[
\begin{array}{c|c|c}
1111111011 & 0 & \text{SLA*} \\ \hline
\text{interface ID} & \\
\end{array}
\]
Interface IDs

the lowest-order 64-bit field of unicast addresses may be assigned in several different ways:

– auto-configured from a 48-bit MAC address (e.g., Ethernet address), expanded into a 64-bit EUI-64

– assigned via DHCP

– manually configured

– auto-generated pseudo-random number (to counter some privacy concerns)

– possibly other methods in the future
Some Special-Purpose Unicast Addresses

- the unspecified address, used as a placeholder when no address is available:
  
  0:0:0:0:0:0:0:0

- the loopback address, for sending packets to self:
  
  0:0:0:0:0:0:0:1
Routing

- uses same “longest-prefix match” routing as IPv4 CIDR

- straightforward changes to existing IPv4 routing protocols to handle bigger addresses
  - unicast: OSPF, RIP-II, IS-IS, BGP4+, …
  - multicast: MOSPF, PIM, …

- can use Routing header with anycast addresses to route packets through particular regions
  - e.g., for provider selection, policy, performance, etc.
Interface Address set

- **Loopback**
  (only assigned to a single virtual interface per node)
- **Link local**
- **Site local**
- **Auto-configured 6to4**
  (if IPv4 public is address available)
- **Auto-configured IPv4 compatible**
  (operationally discouraged)
- **Solicited node Multicast**
- **All node multicast**
- **Global anonymous**
- **Global published**
Interface IDs

Lowest-order 64-bit field of unicast address may be assigned in several different ways:

- auto-configured from a 64-bit EUI-64, or expanded from a 48-bit MAC address (e.g., Ethernet address)
- auto-generated pseudo-random number (to address privacy concerns)
- assigned via DHCP
- manually configured
- possibly other methods in the future
Eui-64

Eui-64 address is formed by inserting "FFFE" and ORing a bit identifying the uniqueness of the MAC address.

-> appendix A : draft-ietf-ipngwg-addr-arch-v3-11.txt
**Multicast Address Format**

<table>
<thead>
<tr>
<th>FP (8bits)</th>
<th>Flags (4bits)</th>
<th>Scope (4bits)</th>
<th>RESERVED (80bits)</th>
<th>Group ID (32bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>11111111</td>
<td>000T</td>
<td>Lcl/Sit/Gbl</td>
<td>MUST be 0</td>
<td>Locally administered</td>
</tr>
</tbody>
</table>

- **flag field**
  
  T=0 indicates permanently-assigned (by IANA) group (=well-known group)
  
  T=1 indicates a transient group
  
  (three other flags reserved)

- **scope field:**
  
  1 – interface-local
  2 - link-local
  3 - site-local
  4 - site-local
  5 - site-local
  6 - site-local
  7 - site-local
  8 - organization-local
  E - global

- **map IPv6 multicast addresses directly into low order 32 bits of the IEEE 802 MAC**

- **Multicast address is never source address**
Pre-defined Multicast Addresses

- FF01:0:0:0:0:0:0:1 : all IPv6 nodes; interface-local
- FF02:0:0:0:0:0:0:1 : all IPv6 nodes; link-local
- FF01:0:0:0:0:0:0:2 : all IPv6 routers; interface-local
- FF02:0:0:0:0:0:0:2 : all IPv6 routers; link-local
- FF05:0:0:0:0:0:0:2 : all IPv6 routers; site-local
A solicited node multicast address is a:
Multicast address with a link-local scope.
Formed by this prefix and the rightmost 24 bits of the unicast/anycast address.
Subnet is /104
Required Addresses

• Node:
  – Link-local address
  – All configured unicast/anycast addresses
  – Loopback address
  – All-nodes multicast address
  – Solicited-node multicast addresses (one for each unicast/anycast address)
  – Multicast addresses for all other groups for this node

• Router:
  – Subnet-router anycast addresses for all interfaces
  – All other anycast addresses
  – All-routers multicast addresses
Multicast Listener Discover – MLD

- MLD is equivalent to IGMP in IPv4
- MLD messages are transported over ICMPv6
- Version number trouble
  
  MLDv1 corresponds to IGMPv2
  
  See RFC 2710

  MLDv2 corresponds to IGMPv3, needed for SSM
  
  See draft-vida-mld-v2-01.txt

- MLD and IGMP are now standardized in the IETF MAGMA working group

  http://www.ietf.org/html.charters/magma-charter.html
IPv6 Multicast Routing

- PIM, SSM, MBGP cover IPv4 and IPv6
  - draft-ietf-pim-sm-v2-new-04.txt,
  - draft-ietf-ssm-overview-02.txt (SSM needs MLDv2)
  - RFC 2858
- Bidir PIM also applicable
- Currently no MSDP work for IPv6
- Strong doubts that BGMP will ever make it as Inter-Domain protocol
- For the time being, it is assumed that SSM solves the Inter-domain IPv6 Multicast problem (?)
Neighbor Discovery - Neighbor Solicitation

ICMP type = 135
Src = A
Dst = Solicited-node multicast of B
Data = link-layer address of A
Query = what is your link address?

ICMP type = 136
Src = B
Dst = A
Data = link-layer address of B

A and B can now exchange packets on this link.
Stateless autoconfiguration

Router solicitation are sent by booting nodes to request RAs for configuring the interfaces.

1. **RS:**
   - ICMP Type = 133
   - Src = ::
   - Dst = All-Routers multicast Address
   - query= please send RA

2. **RA:**
   - ICMP Type = 134
   - Src = Router Link-local Address
   - Dst = All-nodes multicast address
   - Data= options, prefix, lifetime, autoconfig flag

**•** Router solicitation are sent by booting nodes to request RAs for configuring the interfaces.
Packet Size Issues
Minimum MTU

• definitions:
  
  link MTU a link’s maximum transmission unit, i.e., the max IP packet size that can be transmitted over the link

  path MTU the minimum MTU of all the links in a path between a source and a destination

• minimum link MTU for IPv6 is 1280 octets (versus 68 octets for IPv4)

• on links with MTU < 1280, link-specific fragmentation and reassembly must be used
Path MTU Discovery

• implementations are expected to perform path MTU discovery to send packets bigger than 1280 octets:
  
  for each dest., start by assuming MTU of first-hop link

  if a packet reaches a link in which it cannot fit, will invoke ICMP “packet too big” message to source, reporting the link’s MTU; MTU is cached by source for specific destination

  occasionally discard cached MTU to detect possible increase

• minimal implementation can omit path MTU discovery as long as all packets kept = 1280 octets

  e.g., in a boot ROM implementation
Fragment Header

<table>
<thead>
<tr>
<th>Next Header</th>
<th>Reserved</th>
<th>Fragment Offset</th>
<th>0 0 M</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Original Packet Identifier</td>
<td></td>
</tr>
</tbody>
</table>

- though discouraged, can use IPv6 Fragment header to support upper layers that do not (yet) do path MTU discovery
- IPv6 frag. & reasm. is an end-to-end function; routers do not fragment packets en-route if too big—they send ICMP “packet too big” instead
Maximum Packet Size

- base IPv6 header supports payloads of up to 65,535 bytes (not including 40 byte IPv6 header)
- bigger payloads can be carried by setting IPv6 Payload Length field to zero, and adding the “jumbogram” hop-by-hop option:
  
<table>
<thead>
<tr>
<th>Option Type=194</th>
<th>Opt Data Len=4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payload Length</td>
<td></td>
</tr>
</tbody>
</table>

- cannot use Fragment header with jumbograms
Security
IPv6 Security

• all implementations expected to support authentication and encryption headers ("IPsec")
• authentication separate from encryption for use in situations where encryption is prohibited or prohibitively expensive
• key distribution protocols are under development (independent of IP v4/v6)
• support for manual key configuration required
Authentication Header

<table>
<thead>
<tr>
<th>Next Header</th>
<th>Hdr Ext Len</th>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security Parameters Index (SPI)</td>
<td>Sequence Number</td>
<td>Authentication Data</td>
</tr>
</tbody>
</table>

- Destination Address + SPI identifies security association state (key, lifetime, algorithm, etc.)
- provides authentication and data integrity for all fields of IPv6 packet that do not change en-route
- default algorithms is (was?) Keyed MD5
Encapsulating Security Payload (ESP)

- Security Parameters Index (SPI)
- Sequence Number
- Payload
- Padding
- Padding Length
- Next Header
- Authentication Data
Transport Mode ESP (End-to-End)

IPv6 header
[+ ext. headers]
ESP header
e2e ext. headers
transport header
data
ESP trailer

Node 1

Node 2
Tunnel Mode ESP
(End to Security Gateway)
Tunnel Mode ESP
(Gateway to Gateway)
What does IPv6 do for:

- **Security**
  
  Nothing IP4 doesn’t do - IPSec runs in both but IPv6 mandates IPSec

- **QoS**
  
  Nothing IP4 doesn’t do - Differentiated and Integrated Services run in both
  
  So far, Flow label has no real use
Quality of Service
two basic approaches developed by IETF:

- “Integrated Service” (int-serv)
  fine-grain (per-flow), quantitative promises (e.g., x bits per second), uses RSVP signalling

- “Differentiated Service” (diff-serv)
  coarse-grain (per-class), qualitative promises (e.g., higher priority), no explicit signalling
IPv6 Support for Int-Serv

20-bit Flow Label field to identify specific flows needing special QoS

each source chooses its own Flow Label values;
    routers use Source Addr + Flow Label to identify distinct flows

Flow Label value of 0 used when no special QoS requested (the common case today)

this part of IPv6 is not standardized yet, and may well change semantics in the future
8-bit Traffic Class field to identify specific classes of packets needing special QoS

same as new definition of IPv4 Type-of-Service byte

may be initialized by source or by router enroute; may be rewritten by routers enroute

traffic Class value of 0 used when no special QoS requested (the common case today)
Mobility
IPv6 Mobility

- A mobile host has one or more home address(es) relatively stable; associated with host name in DNS.

- When it discovers it is in a foreign subnet (i.e., not its home subnet), it acquires a foreign address.
  - Uses auto-configuration to get the address.
  - Registers the foreign address with a home agent, i.e., a router on its home subnet.

- Packets sent to the mobile’s home address(es) are intercepted by home agent and forwarded to the foreign address, using encapsulation.
Mobile IP (v4 version)

- mobile host
- correspondent host
- foreign agent
- home agent
- home location of mobile host
Mobile IP (v4 version)

- mobile host
- correspondent host
- foreign agent
- home agent
- home location of mobile host
Mobile IP (v4 version)

- mobile host
- correspondent host
- foreign agent
- home agent
- home location of mobile host
Mobile IP (v4 version)

- mobile host
- correspondent host
- foreign agent
- home agent
- home location of mobile host
Mobile IP (v6 version)

- Mobile host
- Correspondent host
- Home agent
- Home location of mobile host
Mobile IP (v6 version)

- mobile host
- correspondent host
- home agent
- home location of mobile host
Mobile IP (v6 version)

- mobile host
- correspondent host
- home agent
- home location of mobile host
Mobile IP (v6 version)

- mobile host
- correspondent host
- home agent
- home location of mobile host
Mobile IP (v6 version)

- Mobile host
- Correspondent host
- Home agent
- Home location of mobile host
ICMP / Neighbor Discovery
ICMP Error Messages

common format:

<table>
<thead>
<tr>
<th>Type</th>
<th>Code</th>
<th>Checksum</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>As much of the invoking packet as will fit without the ICMP packet exceeding 1280 octets</td>
</tr>
</tbody>
</table>

(code and parameter are type-specific)
ICMP Error Message Types

- destination unreachable
  - no route
  - administratively prohibited
  - address unreachable
  - port unreachable
- packet too big
- time exceeded
- parameter problem
  - erroneous header field
  - unrecognized next header type
  - unrecognized option
ICMP Informational Messages

- echo request & reply (same as IPv4)
- multicast listener discovery messages:
  query, report, done (like IGMP for IPv4):

<table>
<thead>
<tr>
<th>Type</th>
<th>Code</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Maximum Response Delay</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Multicast Address</td>
</tr>
</tbody>
</table>
Neighbor Discovery

ICMP message types:
  router solicitation
  router advertisement
  neighbor solicitation
  neighbor advertisement
  redirect

functions performed:
  router discovery
  prefix discovery
  autoconfiguration of address & other parameters
  duplicate address detection (DAD)
  neighbor unreachable detection (NUD)
  link-layer address resolution
  first-hop redirect
Router Advertisements

- periodically multicast by router to all-nodes multicast address (link scope)
- contents:
  - “I am a router” (implied)
  - lifetime as default (1 sec – 18 hr) » prefix
  - “get addresses from DHCP” flag » prefix length
  - “get other stuff from DHCP” flag » valid lifetime
  - router’s link-layer address » preferred lifetime
  - link MTU » on-link flag
  - suggested hop limit » autoconfig OK flag
- not sent frequently enough for unreachability detection
Other Neighbor Discovery Messages

- **router solicitations**
  - sent only at host start-up, to solicit immediate router advert.
  - sent to all-routers multicast address (link scope)

- **neighbor solicitations**
  - for address resolution: sent to “solicited node” multicast addr.
  - for unreachability detection: sent to neighbor’s unicast addr.

- **neighbor advertisements**
  - for address resolution: sent to unicast address of solicitor
  - for link-layer address change: sent to all-nodes multicast addr.
  - usable for proxy responses (detectable)
  - includes router/host flag
Serverless Autoconfiguration ("Plug-n-Play")

- hosts can construct their own addresses:
  - subnet prefix(es) learned from periodic multicast advertisements from neighboring router(s)
  - interface IDs generated locally, e.g., using MAC addresses
- other IP-layer parameters also learned from router adverts (e.g., router addresses, recommended hop limit, etc.)
- higher-layer info (e.g., DNS server and NTP server addresses) discovered by multicast / anycast-based service-location protocol [details still to be decided]
- DHCP also available for those who want more control
Effects on Higher Layers
So Much for the Modularity of Layering!

- changes TCP/UDP checksum “pseudo-header”
- affects anything that reads/writes/stores/passes IP addresses (just about every higher protocol)
- packet lifetime no longer limited by IP layer (it never was, anyway!)
- bigger IP header must be taken into account when computing max payload sizes
- new DNS record type: AAAA and (new) A6
- ...
IPv6 Transition
Transition / Co-Existence Techniques

A wide range of techniques have been identified and implemented, basically falling into three categories:

1. Techniques, to allow IPv4 and IPv6 to co-exist in the same devices and networks.

2. Techniques, to avoid order dependencies when upgrading hosts, routers, or regions.

3. Techniques, to allow IPv6-only devices to communicate with IPv4-only devices.

Expect all of these to be used, in combination.
IPv6 Transition – Tasks & Methods

- Connect IPv6 Islands/Nodes over existing Infrastructure with IPv6 Nodes
  
  Tunneling: Manually or automagically configured – 6to4, ISATAP, ...
  
  IPv6 over dedicated Link-Layer: ATM/FR/SDH/WDM or AToM/L2TPv3

  Dual-Stacked Network

  IPv6 over MPLS: 6PE
Tunnels to Get Through IPv6-Ignorant Routers

• encapsulate IPv6 packets inside IPv4 packets (or MPLS frames)
• many methods exist for establishing tunnels:
  manual configuration
  “tunnel brokers” (using web-based service to create a tunnel)
  “6-over-4” (intra-domain, using IPv4 multicast as virtual LAN)
  “6-to-4” (inter-domain, using IPv4 addr as IPv6 site prefix)
• can view this as:
  IPv6 using IPv4 as a virtual link-layer, or
  an IPv6 VPN (virtual public network), over the IPv4 Internet (becoming “less virtual” over time, we hope)
Overview

- Configured tunnels
- 6to4 tunnels
- Automatic tunnels
- BGP tunnels
- (6over4, NAT-PT)
- 6PE
IPv6 over IPv4 Tunnels

- Several Tunnelling mechanisms defined by IETF
  - GRE, Configured Tunnels, Automatic Tunnels using IPv4 compatible IPv6 Address, 6to4
  - All of the above are supported on Cisco IOS 12.2T
- Apply to ISP and Enterprise WAN networks
- Leverages 6Bone experience
- No impact on Core infrastructure
  - Either IPv4 or MPLS
IPv6 over IPv4 Tunnels
Case Study

ISP scenario
- Configured Tunnels between IPv6 Core Routers
- Configured Tunnels to IPv6 Customers
- MP-BGP4 Peering with other 6Bone users
- Connection to an IPv6 IX
- 6to4 tunnels to IPv6 Customers
- 6to4 relay service

Enterprise scenario
- 6to4 tunnels between sites
- Configured tunnels between sites or to 6Bone users
Configured tunnels

- IPv4: 130.67.0.1, 148.122.0.1
- IPv6: 3ffe:c00:1::/48, 3ffe:c00:2::/48

IPv4 protocol type = 41

<table>
<thead>
<tr>
<th>IPv4 header</th>
<th>IPv6 header</th>
<th>IPv6 payload</th>
</tr>
</thead>
</table>
• Configured tunnels require:
  
  Dual stack end points
  
  Both IPv4 and IPv6 addresses configured at each end
# Configured tunnels II

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>As point to point links</td>
<td>Has to be configured and managed</td>
</tr>
<tr>
<td>Multicast</td>
<td>Inefficient traffic patterns</td>
</tr>
<tr>
<td>Real addresses</td>
<td>No keepalive mechanism, interface is always up</td>
</tr>
</tbody>
</table>
interface Tunnel0
  description ACME networks
  ipv6 address 3FFE:C00:E:0:1::C/64
  tunnel source Fddi0
  tunnel destination 199.185.115.246
  tunnel mode ipv6ip
!
interface Fddi0
  ip address 192.31.7.104 255.255.255.224
!
show ipv6 route
L 3FFE:C00:E:0:1::C/128 [0/0]
  via ::, Tunnel61, 5d12h/never
C 3FFE:C00:E:0:1::C/64 [0/0]
  via ::, Tunnel61, 5d12h/never
ipv6-router#sh ipv6 int tunnel0
Tunnel0 is up, line protocol is up
IPv6 is enabled, link-local address is FE80::C01F:768
Description: ACME networks
Global unicast address(es):
   3FFE:C00:E:0:1::C, subnet is 3FFE:C00:E:0:1::C/64
Joined group address(es):
   FF02::1
   FF02::2
   FF02::1:FF70:47
   FF02::1:FF00:C
   FF02::1:FF1F:768
MTU is 1480 bytes
ICMP error messages limited to one every 500 milliseconds
ND reachable time is 30000 milliseconds
Hosts use stateless autoconfig for addresses.
ipv6-router#
6to4 tunnels

6to4 prefix is 2002::/16 + IPv4 address.
2002:a.b.c.d::/48

IPv6 Internet

6to4 relay
2002:B00:1::1
Announces 2002::/16 to the IPv6 Internet
6to4 Tunnel

• 6to4 Tunnel:
  Is an automatic tunnel method
  Gives a prefix to the attached IPv6 network
6to4 Tunnel (Cont.)

IPv6 Network 6to4 Router IPv4 IPv6 Network

IPv6 6to4 Router IPv4 IPv6

Type: native IPv6 Dst: 2002:c0a8:1e01::1

Type: IPv6 in IPv4 Dst: 192.168.30.1

2002:c0a8:1e01::1

192.168.30.1
6to4 Relay

- 6to4 relay:
  Is a gateway to the rest of the IPv6 Internet
  Default router
### 6to4 tunnels II

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimal configuration</td>
<td>All issues that NMBA networks have.</td>
</tr>
<tr>
<td>Only site border router needs to know about 6to4</td>
<td>Requires relay router to reach native IPv6 Internet</td>
</tr>
<tr>
<td></td>
<td>Has to use 6to4 addresses, not native.</td>
</tr>
</tbody>
</table>

NB: there is a draft describing how to use IPv4 anycast to reach the relay router. (This is already supported, by our implementation...)
6to4 tunnels III
relay

interface Tunnel2002
  no ip address
  ipv6 address 2002:C01F:768::1/128
  tunnel source Fddi0
  tunnel mode ipv6ip 6to4
!
ipv6 route 2002::/16 Tunnel2002
end

ipv6-router#sh ipv6 route
IPv6 Routing Table - 416 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
Timers: Uptime/Expires

LC 2002:C01F:768::1/128 [0/0]
  via ::, Tunnel2002, 4w3d/never
S 2002::/16 [1/0]
  via ::, Tunnel2002, 4w3d/never
6to4 tunnels IV site

interface Tunnel0
  ipv6 address 2002:A31:BCE9::1/64
  tunnel source Ethernet0
  tunnel mode ipv6ip 6to4
!
interface Ethernet0
  ip address 10.49.188.233 255.255.255.248
  ipv6 address 3FFE:C15:C000:8::1/64
!
ipv6 route 2002::/16 Tunnel0
ipv6 route ::/0 2002:C687:1B4::1
end
Automatic tunnels

Connects dual stacked nodes
Quite obsolete

IPv6 Internet

130.67.0.1 ::130.67.0.1

148.122.0.1 ::148.122.0.1
### Automatic tunnels II

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obsolete</td>
<td>Difficult to reach the native IPv6 Internet, without injecting IPv4 routing information in the IPv6 routing table</td>
</tr>
<tr>
<td>Useful for some other mechanisms, like BGP tunnels</td>
<td>Has to use IPv4 compatible addresses</td>
</tr>
</tbody>
</table>
Automatic tunnels III

interface Tunnel0
  tunnel source FastEthernet0/0
  tunnel mode ipv6ip auto-tunnel
!
interface FastEthernet0/0
  ip address 130.67.0.2 255.255.255.0
!

brum-72b# sh ipv6 route
IPv6 Routing Table - 12 entries
Codes: C - Connected, L - Local
Timers: Uptime/Expires
  L   ::130.67.0.2/128 [0/0]
      via ::, Tunnel0, 1d22h/never
  C   ::/96 [0/0]
      via ::, Tunnel0, 1d23h/never
BGP tunnels

Useful for connecting IPv6 PE devices over an IPv4 only core.

BGP next-hop is ::130.67.0.1
Router is configured for automatic tunneling
### BGP tunnels II

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real addresses</td>
<td>Multicast issues</td>
</tr>
<tr>
<td>Simple configuration</td>
<td>BGP convergence times</td>
</tr>
</tbody>
</table>

Where to use: Within one AS! Where it is hard to upgrade the core
router bgp 100
  no bgp default ipv4-unicast
  neighbor ::130.67.0.1 remote-as 100
!
  address-family ipv6
  neighbor ::130.67.0.1 activate
  neighbor ::130.67.0.1 next-hop-self
  network DEAD::/64
  exit-address-family
!
brum-72b#sh bgp ipv6
BGP table version is 4, local router ID is 161.0.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete

<table>
<thead>
<tr>
<th>Network</th>
<th>Next Hop</th>
<th>Metric</th>
<th>LocPrf</th>
<th>Weight</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>*&gt; CAFE::/64</td>
<td>::130.67.0.2</td>
<td>100</td>
<td></td>
<td></td>
<td>i</td>
</tr>
<tr>
<td>*&gt; DEAD::/64</td>
<td>::</td>
<td>32768</td>
<td></td>
<td></td>
<td>i</td>
</tr>
</tbody>
</table>
Tunneling issues

- IPv4 fragmentation. We do not translate IPv4 Path MTU messages to IPv6
- Translating IPv4 ICMP messages and pass back to IPv6 originator
- Some hardware platforms cannot do tunneling well. ->GSR
- End up with an inefficient topology
- Tunnel interface is always up. Use routing protocol to determine link failures or keepalives.
# Tunnel troubleshooting

```
ipv6-router#sh ipv6 tunnel
Tun Route LastInp Packets Description
  0 RIPng never       0 "WIDE (JP) IPv6 Tunnel"
  1 - 00:00:17 417990 ESnet - <nitzan@es.net>
  2 - never       0 NUS-IRDU - <roland@irdu.nus.sg>
  3 - 4w6d       891 Sun - Carl Williams <carlw@Eng.Sun.COM>
  4 - never       0 "NETLAG IPv6 tunnel"
  5 - 00:00:10 106099 UCLA - Yixin Jin <yjin@cs.ucla.edu>
```

ddebug tunnel
ddebug ipv6 packet
ddebug ip packet
ping/traceroute
NAT-PT

NAT-Prefix: prefix::/96 announced by NAT-PT

1. A sends out a DNS request for B
2. NAT-PT intercepts the DNS reply
3. NAT-PT translates from A to AAAA, uses prefix:a.b.c.d as the IPv6 address
4. NAT-PT creates translation slot
5. Communication can begin
Translation

• NAT-PT (Network Address Translation - Protocol Translation) - RFC 2766

• may prefer to use IPv6-IPv4 protocol translation for:
  new kinds of Internet devices (e.g., cell phones, cars, appliances)
  benefits of shedding IPv4 stack (e.g., serverless autoconfig)

• this is a simple extension to NAT techniques, to translate header format as well as addresses

  IPv6 nodes behind a translator get full IPv6 functionality when talking to other IPv6 nodes located anywhere
  they get the normal (i.e., degraded) NAT functionality when talking to IPv4 devices
Cisco IOS NAT-PT features

- NAT-PT support is in 12.2(4th)T
- IP Header and Address translation
- Support for ICMP and DNS embedded translation
- Auto-aliasing of NAT-PT IPv4 Pool Addresses
- Future developments will add ALGs support
  - 1st implementation does not support FTP ALG
6over4 (not implemented)

Useful within one organisation
Uses Neighbor Discovery over IPv4 multicast to reach neighbors
The IPv4 multicast cloud becomes one flat IPv6 Virtual Ethernet
IPv6 over MPLS Infrastructure

- Service Providers have already deployed MPLS in their IPv4 backbone for various reasons
  - MPLS/VPN, MPLS/QoS, MPLS/TE, ATM + IP switching
- Several IPv6 over MPLS scenarios
  - IPv6 over AToM (no impact on IPv6)
  - IPv6 Provider Edge Router (6PE) over MPLS (no impact on MPLS core) -> only PE routers are IPv6 aware
  - Native IPv6 MPLS (require full network upgrade)
6PE

Useful for connecting IPv6 PE devices over an MPLS only core.

BGP session over IPv4. BGP next-hop is ::130.67.0.1 + label
Similar to MPLS VPNs. Two labels, an inner IPv6 and an outer IPv4 label

IPv6
130.67.0.1

IPv4
IPv6
148.122.0.1

iBGP connections
IPv6 over AToM (same for UTI/L2TPv3)

- No impact on existing IPv4 or MPLS Core (v6 unaware)
- Edge MPLS Routers need to support AToM
- Mesh of PE-to-PE connections
- PE routers can be regular IPv6 Routers (V6 over ATM, v6 over FR, v6 over Ethernet,…) or forward just the L2 VC (e.g. Ethernet) to the IPv6 router
6PE Overview

- P routers (LSRs) in the core of the MPLS cloud are not IPv6 aware and just use IPv4 MPLS Control Plane
- 6PE routers are dual stack and use IPv4 MPLS Control Plane with the core, Native IPv6 with IPv6 routers, Native IPv4 with IPv4 routers
- P and 6PE routers share a common IPv4 IGP
- 6PE routers are MP-BGP4 capable, fully or partially meshed
- MPLS dual labels stack is used
### 6PE Overview

**IPv4**
- 192.76.10.0
- 145.95.0.0

**IPv6**
- 2001:0420::
- 2001:0421::
- 2001:0620::
- 2001:0621::
- 192.254.10.0

---

**MP-BGP sessions**
- v4
- v6

**Dual Stack**
- V4: IGP/BGP
- V6: IGP/BGP

**IPv6 unaware**
- No core upgrade
6PE Routing

- IGPv6 or MP-BGP advertising 2001:0421::
- MP-BGP advertises 2001:0421:::
  and binds a (2nd level) label
  IPv6 Next Hop is an IPv4 compatible IPv6 address built from 192.254.10.17

Translation of v6 BGP Next Hop into v4 address
Recursion of this address via IGPv4

- LDPv4 binds label to 192.254.10.17

- IGPv4 advertises reachability of 192.254.10.17

- IPv6 Next Hop is an IPv4 compatible IPv6 address built from 192.254.10.17

- Translation of v6 BGP Next Hop into v4 address
  Recursion of this address via IGPv4
6PE Forwarding

IPv6 packet to 2001:0421::

192.72.170.13

192.254.10.17

2001:0420::

2001:0421::
IPv6 Forwarding and Label Imposition:
• 6PE-1 receives an IPv6 packet
• Lookup is done on IPv6 prefix
• Result is:
  Labelz binded by MP-BGP to 2001:0421::
  Label1 binded by LDP/IGPv4 to the IPv4 address of BGP Next Hop (6PE-2)
IPv6-UNaware MPLS Label Switching:
- P1 receives an MPLS packet
- Lookup is done on Label1
- Result is Label2
IPv6-UNaware MPLS Label Switching:
- P2 receives an MPLS packet
- Lookup is done on Label2
- Result includes Pop label (PHP)
MPLS Label Pop and IPv6 Forwarding:
- 6PE-2 receives an MPLS packet
- Lookup is done on Labelz
- Result is: Pop the label & do IPv6 lookup on IPv6 destination
6PE Standardisation

• See <draft-ietf-ngtrans-bgp-tunnel-04.txt> : “BGP Tunnelling”
• Co-authored by Cisco
• Generic solution for transport of IPv6 over any tunnelling technique (including MPLS) using MP-BGP
• IETF Working Group document
• 6PE is Cisco IOS implementation of “BGP Tunnelling” over MPLS
Summary

- Configured tunnels: just like IP in IP tunnels. Point to point.
- Automatic tunnels: IPv4 compatible addresses, point to multipoint.
- 6to4: 2002::/16. point to multipoint.
- 6over4: Uses IPv4 as an virtual Ethernet.
- NAT-PT: Translation between IPv4 and IPv6
### IPv6 Deployment Phases

<table>
<thead>
<tr>
<th>Phases</th>
<th>Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv6 Tunnels over IPv4</td>
<td>Low cost, low risk to offer IPv6 services. No infrastructure change. Has to evolve when many IPv6 clients get connected.</td>
</tr>
<tr>
<td>Dedicated Data Link layers for Native IPv6</td>
<td>Natural evolution when connecting many IPv6 customers. Require a physical infrastructure to share between IPv4 and IPv6 but allow separate operations.</td>
</tr>
<tr>
<td>MPLS 6PE</td>
<td>Low cost, low risk, it requires MPLS and MP-BGP4. No need to upgrade the Core devices, keep all MPLS features (TE, IPv4-VPN).</td>
</tr>
<tr>
<td>Dual stack</td>
<td>Requires a major upgrade. Valid on Campus or Access networks as IPv6 hosts may be located anywhere.</td>
</tr>
<tr>
<td>IPv6-Only</td>
<td>Requires upgrading all devices. Valid when IPv6 traffic will become preponderant.</td>
</tr>
</tbody>
</table>
ISATAP

• Intra-Site Automatic Tunnel Addressing Protocol
• Connect IPv6 nodes to IPv6 routers within a predominantly IPv4 environment
• Ideal for sparse distribution of IPv6 nodes
• E.g. Campus Networks with IPv4-only L3-Switches
• Incremental deployment
• See draft-ietf-ngtrans-isatap-12.txt
• In 12.2(13)T and 12.2(14)S
ISATAP Details

- Use IANA’s OUI 00-00-5E and encode IPv4 address as part of EUI-64

- Automatic discovery of ISATAP routers
  DNS (isatapgw well known service name)
  Anycast – assignment from 192.88.99.0/24

- Automatic deprecation when end system receives native IPv6 router advertisements
Dual-Stack Approach

• when adding IPv6 to a system, do not delete IPv4
  this multi-protocol approach is familiar and well-understood (e.g., for AppleTalk, IPX, etc.)
  note: in most cases, IPv6 will be bundled with new OS releases, not an extra-cost add-on
• applications (or libraries) choose IP version to use
  when initiating, based on DNS response:
    if (dest has AAAA or A6 record) use
  when responding, based on version of initiating packet
• this allows indefinite co-existence of IPv4 and IPv6, and gradual app-by-app upgrades to IPv6 usage
Dual Stack IPv4-IPv6 Infrastructure

• More appropriate to Campus or Access networks
• On WAN, is generally a long term goal, when IPv6 traffic and users will be rapidly increasing
• Can be configured on Cisco IOS 12.2(2)T but have to consider
  - Memory size for IPv4 and IPv6 routing tables
  - IGP options: Integrated versus “Ships in the Night”
  - Full network upgrade
• IPv4 and IPv6 traffic should not impact each other
  - Require more feedback and experiments
For More Information

- http://www.6bone.net/
IPv6 Deployment
Packaging

- Two IPv6 feature sets - with and without BGP.
- the IPv6 feature is included in Provider, Plus and Home Office images.
- full IPv6 support in -p-, -is-, -ds-, -js, -ys-
- without BGP in -y-
- IPv6 support in boot images exists but not built for the 12.2(1)T release
CLI

- Generally new top level IPv6 exec and configuration commands
  
  e.g. \{show, clear, debug\} ipv6 <>

- Hooked into some IPv4 commands
  
  ip as-path, ip name-server

- Routing protocols are different
  
  RIPng has new IPv6 cli
  BGP4+ shared config with other address-families
Processes

• IPv6 Input

High priority process, whose main task is to handle the ipv6_packetQ.

IPv6 background tasks:

- garbage collecting the MTU cache
- reassemble fragmented datagrams
- static routing, loop detection
- handle interface up/down events.
Processes

- IPv6 ND

Normal priority process, handles:

- Neighbour Discovery events.
- Sending periodic Router Advertisements
- Maintaining the Neighbour & Router table
- (Duplicate Address Detection)
IPv6 data structures - RIB

- Note that as opposed to IPv4 we store our own interface addresses as host routes in the RIB.

- CLI
  
  show ipv6 route
  clear ipv6 route
  debug ipv6 routing
IPv6 data structures
Neighbor table

A global data structure, organised as a hash. Currently with 256 buckets.

Used for storing the link layer address of a neighbour and the neighbour's reachability state.

Partly similar to IPv4's ARP table.

• CLI

  show ipv6 neighbors
  clear ipv6 neighbors
  debug ipv6 nd
Scopes

- Conceptual routing table per zone
- Forwarding of packets with link-local destination is not supported
- Site-locals are treated as globals, ie we do not support a multi-sited router, or do scope checks on output
- Limited application support for scopes. e.g.% syntax. (ping fe80::1%Ethernet0)
- No VPNs
IPv6 Configuration
Forwarding

- Forwarding is off by default! Enable with “ipv6 unicast-routing”
- Routing table also in host mode. Uses received Router Advertisements to pick default router.
- No ip default-gateway or ip default-network.
- Per-packet load balancing.
IPv6 addresses

- An IPv6 interface has many addresses:
  - link-local e.g. fe80::/10
  - site-local e.g. fec0::/10
  - multiple global addresses: e.g. 2001::/64
  - anycast addresses

- Belongs to multiple multicast groups:
  - all-hosts ff02::1
  - all-routers ff02::2
  - solicited node multicast address, e.g. ff02::1:ff18:c00 (ff02:0:0:0:0:1:ff::/104)

- Subnet-router anycast address e.g. 2001::/128
IPv6 addresses II

brum-72a#show ipv6 interface
FastEthernet0/0 is up, line protocol is up
IPv6 is enabled, link-local address is FE80::250:A2FF:FE18:C00
Global unicast address(es):
   DEAD:BEEF:CAFE:0:250:A2FF:FE18:C00, subnet is DEAD:BEEF:CAFE::/64
   3FFE:C00::1, subnet is 3FFE:C00::/64
Joined group address(es):
   FF02::1
   FF02::2
   FF02::1:FF18:C00
   FF02::1:FF00:1
MTU is 1500 bytes
ICMP error messages limited to one every 500 milliseconds
ND DAD is enabled, number of DAD attempts: 1
ND reachable time is 30000 milliseconds
ND advertised reachable time is 0 milliseconds
ND advertised retransmit interval is 0 milliseconds
ND router advertisements are sent every 200 seconds
ND router advertisements live for 1800 seconds
Hosts use stateless autoconfig for addresses.
IPv6 addresses - commands

Configures addresses on an interface:
  [no] ipv6 address <prefix>/64 eui-64
  [no] ipv6 address <prefix>
  [no] ipv6 address <link-local> link-local

Enable ipv6 processing on an interface:
  [no] ipv6 enable

Use global address from some other interface:
  [no] ipv6 unnumbered <interface>

Link-local address is automatically configured. Can be overridden. Multicast memberships are automatically configured.
## Default source address selection

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Precedence</th>
<th>Label</th>
<th>MatchSrcLabel</th>
</tr>
</thead>
<tbody>
<tr>
<td>fe80::/10</td>
<td>40</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>fec0::/10</td>
<td>30</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>::/0</td>
<td>20</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2002::/16</td>
<td>10</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>::/96</td>
<td>10</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

- Picks source of same scope or type as the destination address
- Longest common prefix length
- Different rules for address selection for: ICMP error messages, ND, Routing protocols
Neighbor Discovery

- Address resolution
- Address autoconfiguration
- Prefix Discovery
- Parameter Discovery
- Neighbor Unreachability Detection
- Duplicate Address Detection
- Redirect
- Router Discovery
Neighbor Discovery Messages

- Neighbor Solicitation
- Neighbor Advertisement
- Router Solicitation
- Router Advertisement
- Redirect
Neighbor Solicitation message

<table>
<thead>
<tr>
<th>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>+----------------------------------------------------------+-</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>+----------------------------------------------------------+-</td>
</tr>
<tr>
<td>+----------------------------------------------------------+-</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>+----------------------------------------------------------+-</td>
</tr>
<tr>
<td>+----------------------------------------------------------+-</td>
</tr>
<tr>
<td>+----------------------------------------------------------+-</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>+----------------------------------------------------------+-</td>
</tr>
<tr>
<td>+----------------------------------------------------------+-</td>
</tr>
<tr>
<td>+----------------------------------------------------------+-</td>
</tr>
<tr>
<td>+----------------------------------------------------------+-</td>
</tr>
<tr>
<td>+----------------------------------------------------------+-</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>+----------------------------------------------------------+-</td>
</tr>
</tbody>
</table>
# Neighbor Advertisement message

<table>
<thead>
<tr>
<th>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>+---------------------------------------------------------------+</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>+---------------------------------------------------------------+</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>+---------------------------------------------------------------+</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>+---------------------------------------------------------------+</td>
</tr>
<tr>
<td>+---------------------------------------------------------------+</td>
</tr>
<tr>
<td>+---------------------------------------------------------------+</td>
</tr>
<tr>
<td>+---------------------------------------------------------------+</td>
</tr>
<tr>
<td>+---------------------------------------------------------------+</td>
</tr>
<tr>
<td>+---------------------------------------------------------------+</td>
</tr>
</tbody>
</table>

© 2001, Cisco Systems, Inc. All rights reserved.
## Router Solicitation message

<table>
<thead>
<tr>
<th>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>+----------------------------------------------------------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>+----------------------------------------------------------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>+----------------------------------------------------------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>+----------------------------------------------------------</td>
</tr>
</tbody>
</table>
# Router Advertisement message

<table>
<thead>
<tr>
<th>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Cur Hop Limit</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>reachable Time</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>Retrans Timer</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>Options ...</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
</tr>
</tbody>
</table>
## Redirect message

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 |
| + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| Target Address |
| + |
| + |
| Destination Address |
| + |
| + |
| Options ... |
| + |

<table>
<thead>
<tr>
<th>Type</th>
<th>Code</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

| + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |

| Options ... |
| + |
| + |

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 |
Neighbor Discovery options

| 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 |
|---------------------------------|-----------------------------|
| +---------------------------------+-+-----------------------------|
| |     Type      |    Length     |              ...              |

Fields:

Option Name                      Type
Source Link-Layer Address         1
Target Link-Layer Address         2
Prefix Information                3
Redirected Header                 4
MTU                                5
Neighbor Discovery options II

Source/Target Link-layer Address

<table>
<thead>
<tr>
<th>Type</th>
<th>Length</th>
<th>Link-Layer Address ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9</td>
<td>0 1 2 3 4 5 6 7 8 9</td>
<td></td>
</tr>
</tbody>
</table>

Redirected header

<table>
<thead>
<tr>
<th>Type</th>
<th>Length</th>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7</td>
<td></td>
<td>0 1 2 3 4 5 6 7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7</td>
</tr>
</tbody>
</table>

IP header + data

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7</td>
</tr>
</tbody>
</table>

MTU

<table>
<thead>
<tr>
<th>Type</th>
<th>Length</th>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7</td>
<td></td>
<td>0 1 2 3 4 5 6 7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MTU</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7</td>
</tr>
</tbody>
</table>
Neighbor Discovery options III

Prefix Information:

```
+-----------------+-----------------+-----------------+----+
|  Type  |  Length  | Prefix Length |L|A|
+-----------------+-----------------+-----------------+----+
   |                      Valid Lifetime                         |
+-----------------+-----------------+-----------------+----+
   |                  Preferred Lifetime                        |
+-----------------+-----------------+-----------------+----+
   |                        Reserved2                           |
+-----------------+-----------------+-----------------+----+
   |                                                             |
+-----------------+-----------------+-----------------+----+
   +                              Prefix                             |
```
ND Autoconfiguration, Prefix & Parameter Discovery

1. RS:
   - ICMP Type = 133
   - Src = ::
   - Dst = All-Routers multicast Address
   - query = please send RA

2. RA:
   - ICMP Type = 134
   - Src = Router Link-local Address
   - Dst = All-nodes multicast address
   - Data = options, prefix, lifetime, autoconfig flag

• Router solicitation are sent by booting nodes to request RAs for configuring the interfaces.
ND Address Resolution & Neighbor Unreachability Detection

ICMP type = 135 (NS)
Src = A
Dst = Solicited-node multicast of B
Data = link-layer address of A
Query = what is your link address?

ICMP type = 136 (NA)
Src = B
Dst = A
Data = link-layer address of B

A and B can now exchange packets on this link
ND Redirect

- Redirect is used by a router to signal the reroute of a packet to an onlink host to a better router or to another host on the link.
ND Duplicate Address Detection

ICMP type = 135
Src = 0 (::)
Dst = Solicited-node multicast of A
Data = link-layer address of A
Query = what is your link address?

• Duplicate Address Detection (DAD) uses neighbor solicitation to verify the existence of an address to be configured.
brum-72a#sh ipv6 routers

Router FE80::210:7BFF:FEC2:ACCC on Ethernet2/1, last update 2 min
   Hops 64, Lifetime 1800 sec, AddrFlag=0, OtherFlag=0
   Reachable time 0 msec, Retransmit time 0 msec
   Prefix 3F02:1::/32 onlink autoconfig
   Valid lifetime -1, preferred lifetime -1

Router FE80::290:92FF:FE65:6800 on Ethernet2/0, last update 0 min
   Hops 64, Lifetime 1800 sec, AddrFlag=0, OtherFlag=0
   Reachable time 0 msec, Retransmit time 0 msec

Router FE80::250:A2FF:FE4D:A9 on Ethernet2/1, last update 1 min
   Hops 64, Lifetime 1800 sec, AddrFlag=0, OtherFlag=0
   Reachable time 0 msec, Retransmit time 0 msec
   Prefix 3001:1::/64 onlink autoconfig
   Valid lifetime -1, preferred lifetime -1
   Prefix 2F07:1::/64 onlink autoconfig
   Valid lifetime -1, preferred lifetime -1
### Neighbor Discovery

#### Neighbor cache

```
brum-72a#show ipv6 neighbors
IPv6 Address                        Age Link-layer Addr State Interface
FE80::202:4BFF:FEA3:9DB2             2776 0002.4ba3.9db2 STAEL Ethernet2/0
FE80::290:92FF:FE65:6800            0 0090.9265.6800 REACH Ethernet0/0
FE80::290:92FF:FE65:681D             3079 0090.9265.681d STAEL Ethernet2/1
```

```
brum-72a#debug ipv6 nd
1d02h: ICMPv6-ND: Sending RA to FF02::1 on FastEtherenet0/0
1d02h: ICMPv6-ND: prefix = 5000::1/64 onlink autoconfig
1d02h: ICMPv6-ND: INCMP created: 5000::2
1d02h: ICMPv6-ND: Sending NS for 5000::2 on FastEtherenet0/0
```

NB! the Neighbor cache is only used for multicast capable multi-access links (Ethernet/FDDI). You will not see neighbors on p2p links.
Neighbor Discovery commands

```
brum-72a(config-if)#ipv6 nd ?

dad         Duplicate Address Detection
managed-config-flag Hosts should use DHCP for address config
ns-interval Set advertised NS retransmission interval
other-config-flag Hosts should use DHCP for non-address config
prefix-advertisement Configure IPv6 Routing Prefix Advertisement
ra-interval Set IPv6 Router Advertisement Interval
ra-lifetime Set IPv6 Router Advertisement Lifetime
reachable-time Set advertised reachability time
suppress-ra Suppress IPv6 Router Advertisements
```

```
show ipv6 interface
<snip>
ND reachable time is 30000 milliseconds
ND advertised reachable time is 0 milliseconds
ND advertised retransmit interval is 0 milliseconds
ND router advertisements are sent every 200 seconds
ND router advertisements live for 1800 seconds
Hosts use stateless autoconfig for addresses
```

```
ipv6 nd prefix-advertisement <prefix> <valid-time>
<preferred-time> [onlink | autoconfig]
```
Access lists

- Named standard access lists
- Supports only source/destination address/prefix
- Implicit deny at the end
- Use prefix-lists for filtering prefixes!
- Extended access-lists since 12.2(14)S, 12.2(13)T
Access lists - usage

ipv6 access-list <name> {permit | deny} {<source prefix> | any} {<destination prefix> | any} [priority <prio>]

no ipv6 access-list <name>

Traffic filtering:
ipv6 access-list foo deny 3000::/64 any
ipv6 access-list foo deny ::/96 4000:1:2::1/128
ipv6 access-list foo permit any any

interface Ethernet0
  ipv6 traffic-filter <aclname> [in|out]

To the router itself:
line vty 0 4
  ipv6 access-class <aclname> [in|out]
Access lists - old syntax

```
ipv6 access-list <name> permit [<prefix> [*]| any] {<prefix> [*]| any}
ipv6 access-list <name> deny [<prefix> [*]| any] {<prefix> [*]| any}
no ipv6 access-list <name>
```

Prefix filtering (routing protocols)
```
ipv6 access-list foo deny 3000::/96 *
ipv6 access-list foo deny 3000::/65 *
ipv6 access-list foo deny 4000:1234:1234:1234::/64
ipv6 access-list foo permit any
```

Route-maps:
```
route-map foo permit 10
match ipv6 address foo
```

Please don’t use access lists for route filtering!
IPv6 media specific issues

- **Dialer**
  
  dialer-map and dialer-list commands are extended. e.g
dialer map ipv6 2001:1:1::1 ...

- **ATM PVC**
  
  point to point, or point to multipoint. Multipoint requires
a map-list. Supports both aal5mux and aal5snap.

- **FR PVC**
  
  frame-relay map command extended. Or use point-to-
point.
IPv6 misc commands

- ipv6 hop-limit
- ipv6 host
- ipv6 icmp error-interval
- show ipv6 mtu
IPv6 client applications

• ping [ipv6] {hostname|IPv6-address}
  Extended ping and ping using UDP echo is supported
• traceroute [ipv6] {hostname|IPv6-address}
• telnet {IPv6-address|hostname} /ipv6
• tftp, rsh, rlogin, UDP small servers