Intro to IPv6 (nextgen)

Jim Binkley
jrb@cs.pdx.edu
http://www.cs.pdx.edu/~jrb/tcpip.html
IPng - history

- early 90s IETF decided to accept proposals to replace IPv4, three possibilities:
  - SIP(P), Simple IP Plus (SIP + PIP = SIPP)
  - CATNIP, based on ISO CLNP addresses
  - ISO CLNP - variable length addresses
- SIPP chosen, now IPv6 or IP next gen
- SIP advocated 64-bit addresses, IAB settled on 128 addresses for IP src/dst

Jim Binkley
reminder - scalability problems

- exhaustion of IP host addresses/IP networks. IPv6 can **address** this
  - humble apologies for inate pun
- DNS (or .com) growth. NOPE
- routing/network address scalability. CIDR addresses this, not IPv6. NOPE again.
- bottom line: if IPv6 prospers, it prospers under a CIDR administration
- put another way: **unicast allocation is important**

Jim Binkley
IPv6 header (version 3?)

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>version:4</td>
<td>4</td>
</tr>
<tr>
<td>priority:4</td>
<td>4</td>
</tr>
<tr>
<td>flow label:24</td>
<td></td>
</tr>
<tr>
<td>payload length:16</td>
<td>16</td>
</tr>
<tr>
<td>next hdr:8</td>
<td></td>
</tr>
<tr>
<td>hop limit:8</td>
<td>8</td>
</tr>
<tr>
<td>ipNG source address:128</td>
<td></td>
</tr>
<tr>
<td>ipNG destination address:128</td>
<td></td>
</tr>
</tbody>
</table>

40 byte fixed length header, no checksum, options replaced by routing extension headers

Jim Binkley
IPv6 address obviously long!

in hex notation: could be:
1234:ABCD:4321:DCBA:01FE:1212:DEAD:BEEF

8 16 bit segments in hex

note: possibility of mapping in other address spaces (IPv4, IPX, ISO, Social Security Number)

makes DHCP server (IP/MAC binding), and DNS server name/IP binding) a requirement

Jim Binkley
addressing, a few details

- in theory, 1500 or so addresses per square meter of earth’s surface (2 **128 is big number)
- don’t write leading zeros, compress with ::,  
  – must write trailing zeroes
- use HEX, except allow dotted decimal IPv4 at end in one case
address high-level architecture

- FP, format prefix at FRONT is variable-length

- allocation reserved address-space-slice

- reserved 00000000 1/256

- unicast 001 1/8

- unique local unicast FC00/7
  - expected to be globally unique (next 40 bits)

- link-local unicast 1111 1110 10 (FE8) 1/1024

- multicast 1111 1111(FF00) 1/256

Jim Binkley
reserved addresses

• starts with 0x00, note that 0011-111X (except multicast) must have EUI-64 (MAC) bits at end
• unspecified address (all 0’s):
  – 0000:0000:0000:0000:0000:0000 or ::
  – can be src during boot phase, not destination
• ::1 - loopback address
• ::10.0.0.1, ipv4-compatible ipv6 addr
• :: - 0 meaning “me”
local addresses

- link-local used on single link (0xfe)
  1111111010 | 0 (54 zeroes total) | if ID (64 bits)
  - auto-address configuration
  - neighbor discovery
  - no routers present

- unique local unicast (FC00::/7) - unique across subnets
anycast idea

- ipv6 addresses are anycast, unicast, multicast
- “no” broadcast - subsumed by multicast
- anycast: unicast address assigned to more than 1 interface (probably router?)
- some TBD routing technology must route packet to “nearest” interface

Jim Binkley
aggregatable global unicast addr

- **in theory**

<table>
<thead>
<tr>
<th>FP</th>
<th>TLA ID</th>
<th>RES</th>
<th>NLA ID</th>
<th>SLA ID</th>
<th>interface ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>13</td>
<td>8</td>
<td>24</td>
<td>16</td>
<td>64 bits in field</td>
</tr>
</tbody>
</table>

FP = 001
TLA - top-level aggregation identifier, 8k worth, assigned in parts to registry (RIPE, APNIC, ARIN)
NLA - next-level aggregation, to ISPs, /48 public bits
SLA - site-level aggregation, subnets within site
acc. to www.arin.net

- 2001:04AB:0000:0000:0000:0000:0000:0000/35
  as a TLA/NLA allocation example

<table>
<thead>
<tr>
<th>FP</th>
<th>TLA ID</th>
<th>sub-TLA</th>
<th>Res</th>
<th>NLA ID</th>
<th>site local bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>13</td>
<td>13</td>
<td>6</td>
<td>13</td>
<td>80 (sla + if id)</td>
</tr>
</tbody>
</table>

e.g., arin allocates /35 to “big pipe inc” who allocate from NLA space to Enormous State University (ESU) aggregation is important goal, arin wants 8k TLA routes max

Jim Binkley
whois -h rs.arin.net 2001::/21

- produces
  APNIC-001  2001:0200:0*/23
  ESNET-V6  2001:0400:0*/35
  ARIN-001  2001:0400:0*/23
  RIPE-001  2001:0600:0*/23

- whois -h rs.arin.net ARIN-001 will produce full registration info

- ESNET-V6 is the 1st recipient of IPv6 address space from ARIN

Jim Binkley
EUI-64 in a nutshell (IPv6)

- take 48 bit MAC, divide into 2 24-bit parts
- first 24 bits to the front (of the 64 bit space),
- last 24 bits to the end
- put FFFE in the middle (now 64)
- change from left bit 7 to a 1
example:

- **IPv6 address:**
  

- **MAC address:** 00:50:04:76:0f:cf

- so put 00:50:04 in the front

- 76:0f:cf in the back

- ff:fe in the middle

- change 00 to 02 for 7th bit
“transition” strategy with IPv4

- none or minimized flag days
- hosts have dual-stacks, IPv6 and IPv4
- tunnels: IPv6 internets can tunnel IPv6 packets over IPv4 networks, “short-term”
  - IPv4 | IPv6 datagram (IPv6 header + rest)
- if and when more IPv6, then IPv4 tunneled over IPv6
  - IPv6 | IPv4 datagram
- transition likely to be a very long time

Jim Binkley
some features/details

- flow-labels for QOS
- routing extension headers
- multicast addressing
- auto-configuration
- arp replaced by ICMP neighbor discovery and solicitation messages using multicast (no further slides on that subject)
flow-label

- flow informally defined as “associated packets between two ES, or multicast src and all dst); e.g., audio stream, video stream, web transaction
- at IP-level, (ip src, ip dst, priority field, flow id), flow id is src generated
- flow tuple to be used in routers for QOS scheduling
router-extension headers

- features taken OUT of ip header; e.g., ip fragmentation
- encapsulated in additional headers that follow ip header, precede TCP/UDP level
- include: hop by hop, routing, fragment, destination options, security
- security (IPSEC): Authentication Header (AH), Encapsulating Security Payload (ESP) (encryption + optional authentication)
- recommended ordering exists for above; e.g., hop by hop first, ESP near end
fragmentation example

- IPv6 packets if too large for PATH MTU, all require router ICMP error back to sender
  - router error: path MTU here is N bytes
- sender IP must fragment

<table>
<thead>
<tr>
<th>ip headers</th>
<th>fragment hdr</th>
<th>frag 1</th>
</tr>
</thead>
</table>

followed by frag 2, frag 3, ... frag the last

<table>
<thead>
<tr>
<th>next header</th>
<th>reserved</th>
<th>offset (13)</th>
<th>res</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip identification for ip datagram</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

M = 0 on last fragment, else 1

Jim Binkley
multicast addressing

0xFF | flg(4) | scope(4) | group id (112)

flag field has 000T, where T bit if 0, means IANA assigned, else not permanently assigned

scope bits limit multicast scope (better than current IP ttl) to (e.g.,) link local/site local/organization local/global

routers may presumably enforce these distinctions
multicast address examples

- prefixes FF00..FF0F: followed by zero reserved
- FF01:<6 * 0000>: 0001 - node local scope
- FF02:<6 * 0000>: 0002 - link local scope
- FF01:<6 * 0000>: 0002 - node local/all IPv6 routers
- FF02:<6 * 0000>: 0002 - link local/all IPv6 routers
- range FF02:0000:0000:0000:0000:0001:FF00:0000 to FF02:0000:0000:0000:0000:0001:FFFF:FFFF
  - used for neighbor discovery process
- FF02:0:0:0:0:0:0: 5 and 6 used by OSPF
IEEE has extended 48-bit MAC to be 64 bits

- e.g., 48 bit MAC becomes EUI-64 by setting bit 7 to 1
  - `cccccc1gcccccccccccccccccc` - OUI (org. unique id) in 24 bits +
  - `0xFF 0xFE` (16 bits) + (insert two fixed pad bytes)
  - 24 bits of manufacturer bits

site local address (subnet 1) hypothetical example:

- `FEC0:0000:0000:0001:020A:0AFF:FE01:0203`
stateless auto-configuration

- multicast-capable (broadcast) i/f like ethernet at boot can generate host-id portion
  - subject to duplicate address detection check
- router periodically sends router advertisement with net bytes acc. to local subnet prefix
  - flag bits indicate stateful/stateless auto-config
  - host may send router solicitation if impatient
- multicast addresses used to send these packets
- bottom-line some/all addresses can be dynamically configured
crystal-ball and final whines

- my crystal-ball is broken, unclear when IPv6 “will take over”
- not thrilled about hype believed by NAIVE folks:
  - “makes inet secure”, “mobility not possible with IPv4” “gives us Quality of Service” (sigh)
- some think didn’t go far enough for the amount of pain it will cause
- allocation all-crucial, and due to CIDR plus organizational experience, not IPv6
- not SIMPLE IP any more ...

Jim Binkley
IPv6 at PSU - reality check

- Internal allocation for CECS inside of PSU: 2001:0468:1f04:0200::/56
- Allocation for netlab within CECS: 2001:0468:1f04:02f0::/60
- Welcome to IPv6 and CIDR ...
- Remember there are 64 bits of IP
Jim Binkley

netlab IPv6 topology

OIT PSU router

OIT-netlab ipv6/ipv4 tunnel

ipv6/ipv4 abilene tunnel/s

netlab Ethernet (net 0/vlan 1)

dfulton.cs.pdx.edu
netlab ipv6 router
FreeBSD host

hahaha.cs.pdx.edu
ipv6 host
FreeBSD host
there are 5 tasks

1. turn IPv6 “on” and enable router function

2. setup a gif0 tunnel to the OIT router

3. manually allocate an IPv6 address for the one interface used here

4. create a manual IPv6 default route thru the tunnel

5. run a router advert daemon so that auto-config will work for local subnet hosts

Jim Binkley
Jim Binkley

**dfulton - router setup in /etc/rc.conf**

- ipv6 on: in /etc/rc.conf
  - ipv6_enable="YES"
- enable router functionality
  - ipv6_defaultrouter="YES"
  - ipv6_gateway_enable="YES"
  - ipv6_router_enable="YES"
- rtadvert daemon
  - rtadvd_enable="YES"
  - rtadvd_interfaces="xl0"
dfulton - router setup in /etc/rc.conf

- bind ip address to xl0
  - ipv6_ifconfig_xl0_alias0="2001:468:1f04:2f0:201:2ff:fe48:9659 prefixlen 64"

- in /etc/rc.local add tunnel setup
  - ifconfig gif0 create
  - ifconfig gif0 tunnel 131.252.215.3 131.252.2.66
  - ifconfig gif0 inet6 alias 2001:468:1F04:2::2 prefixlen 64

- default route for ipv6 thru tunnel
  - route add -inet6 default -interface gif0
host setup on hahaha.cs.pdx.edu

- all we need to do is to turn ipv6 on
- however we could add commands
  - 1. rtsold <interface>
  - 2. rtsol <interface>
- for router solicitation messages
- rtsol is done at boot anyway for auto-config
ifconfig on dfulton

# ifconfig xl0
xl0: flags=8943<UP, BROADCAST, ...>
...
inet 131.252.215.3 netmask 0xfffffffffe0 broadcast 131.252.215.31
inet6 fe80::201:2ff:f348:9659%xl0 prefixlen 64 scopeid 0x
inet6 2001:468:1f04:2f0:201:2ff:fe48:9659 prefixlen 64 ether 00:01:02:48:96:59
ifconfig on hahaha.cs.pdx.edu

- ifconfig xl0
  inet 131.252.215.15 ...
  inet6 fe80::250:4ff:fe76:fcf%xl0 ...
  inet6 2001:486:1f04:2f0:250:4ff:fe76:fcf
  prefixlen 64 autoconf
  ether 00:50:04:76:0f:cf
note tools on freebsd

- ping6
- traceroute6
- is there a telnet6 ? (no ...)
  - very important news on the DNS front ...
DNS revisited

- **goal:** support both ipv6/ipv4 lookup in the same application
- all apps need to be rewritten, but it’s not difficult
- `getaddrinfo(3)` replaces `gethostbyname(3)` and `getservbyname(3)` - protocol independent
- `getnameinfo(3)` replaces `gethostbyaddr(3)` and `getservbyport(3)`
look at handouts

- 1. Inet6 traceroute: *ipv6.traceroute6.txt*
- 2. netstat -a from a host: *ipv6.netstat.txt*
- 3. ndp -a from a host: *ipv6.ndp.txt*
- 4. look at C src example of getaddrinfo(3) *tcpclient.c*
- 5. look at C src example: *tcpserver.c*
bottom-line: so what’s important?

◆ cut and paste!!!
  – all those long addresses
◆ auto-configuration
◆ tunnels (ipv4 over ipv6) “for now” (forever)
◆ getaddrinfo(3)