IPv6 Introduction

by

Harald Welte <laforge@rfc2460.org>

What? Why?

- What is IPv6?
  - Successor of currently used IP Version 4
  - Specified 1995 in RFC 2460
- Why?
  - Address space in IPv4 too small
  - Routing tables too large

Advantages

- Advantages
  - stateless autoconfiguration
  - multicast obligatory
  - IPsec obligatory
  - Mobile IP

  - Address renumbering
  - Multihoming
  - Multiple address scopes
  - smaller routing tables through aggregatable allocation

  - simplified l3 header
    - 64bit aligned
    - no checksum (l4 or l2)
    - no fragmentation at router

Disadvantages

- Disadvantages
  - Not widely deployed yet
  - In most cases access only possible using manual tunnel
  - OS support not ideal in most cases
    - W2k: IPv6 available from MS
    - Windows XP: IPv6 included
    - Linux has support, but not 100% RFC compliant
    - *BSD: full support (KAME)
    - Solaris 8/9/10: full support
  - Application support not ideal in most cases
    - Biggest problem: squid
    - supported: bind8/9, apache, openssh, xinetd, rsync, exim, zmailer, sendmail, qmail, inn-2.4(CVS), zebra, mozilla
  - Conclusion: Circular dependencies
    - no application support without OS support
    - no good OS support without applications
    - no wide deployment without applications
    - no applications without deployment
    - no deployment without applications

Deployment

- Experimental (6bone)
  - Experimental 6bone (3ffe::) has been active since 1995.
  - Uses slightly different Addressing Architecture (RFC2471)
  - Phased out on 06/06/2006
  - No new pTLA assignments starting from 2005
- Production (2001::)
  - Initial TLA's and sub-TLA's assigned in Sept 2000
  - Mostly used in education+research
  - Some commercial ISP's in .de are offering production prefixes
- Why isn't IPv6 widely used yet?
  - No immediate need in Europe / North America
  - Big deployment cost at ISP's (Training, Routers, ..)

Technical: Address Space

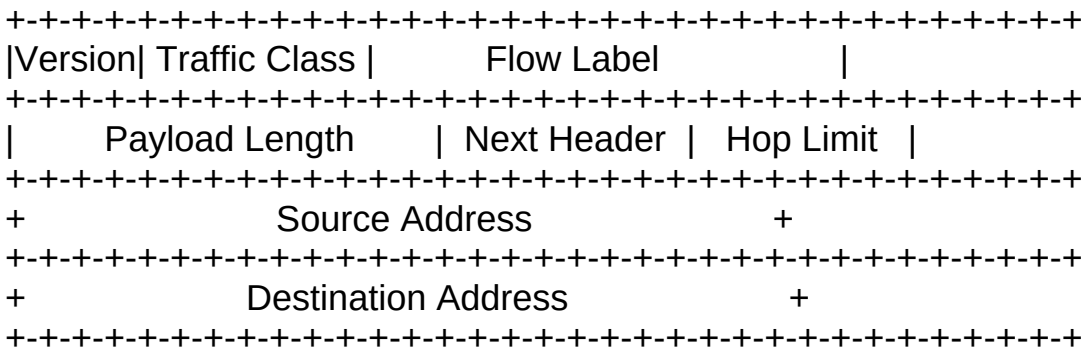
- IP Version 6 Addressing Architecture (RFC2373)
  - Format prefix, variable length
    - 001: RFC2374 addresses, 1/8 of address space
    - 0000 001: Reserved for NSAP (1/128)
    - 0000 010: Reserved for IPX (1/128)
    - 1111 1110 10: link-local unicast addresses (1/1024)
    - 1111 1110 11: site-local unicast addresses (1/1024)
    - 1111 1111 flgs scop: multicast addresses

flgs (0: well-known, 1:transient)
scop (0: reserved, 1: node-local, 2: link-local, 5: site-local, 8: organization-local, e: global scope, f: reserved)

Technical: Address Space

- Aggregatable Global Unicast Address Format (RFC2374)
  - 3bit FP (format prefix = 001)
  - 13bit TLA ID - Top-Level Aggregation ID
  - 13bit Sub-TLA - Sub-TLA Aggergation ID
  - 19bit NLA - Next-Level Aggregation ID
  - 16bit SLA - Site-Level Aggregation ID
  - 64bit Interface ID - derived from 48bit ethernet MAC
- Initial subTLA-Assignments
  - 2001:0000::/29 - 2001:01f8::/29 IANA
  - 2001:0200::/29 - 2001:03f8::/29 APNIC
  - 2001:0400::/29 - 2001:05f8::/29 ARIN
  - 2001:0600::/29 - 2001:07f8::/29 RIPE
- loopback ::1
- unspecified: ::0
- embedded ipv4
  - IPv4-compatible address: 0::xxxx:xxxx
  - IPv6-mapped IPv4 (IPv4 only node): 0::ffff:xxxx:xxxx
- anycast
  - allocated from unicast addresses
  - only subnet-router anycast address predefined (prefix::0000)

Technical: Header



- 4bit Version: 6
- 8bit Traffic Class
- 20bit Flow Label
- 16bit Payload Length (incl. extension hdrs)
- 8bit next header (same values like IPv4, RFC1700 et seq.)
- 8bit hop limit (TTL)
- 128bit source address
- 128bit dest address
- extension headers:
  - hop-by-hop options
  - routing
  - fragment
  - destination options
  - IPsec (AH/ESP)

Technical: Layer 2 <-> Address mapping

- Ethernet: No more ARP, everything within ICMPv6
- No Broadcast, everything built using multicast.

- all-nodes multicast address ff02::1
- all-routers multicast address ff02::2

Technical: Address Configuration

- router discovery
  - routers periodically send router advertisements
  - hosts can send router solicitation to explicitly request RADV
- prefix discovery
  - router includes prefix(es) in ICMPv6 router advertisements
  - other nodes receive prefix advertisements and derive their final address from prefix + EUI64 of MAC address
- neighbour discovery
  - machines can discover it’s neighbours without advertising router

How to get connected

- In case of static IPv4 address
  - SIT (ipv6-in-ipv4) tunnel possible
  - http://www.join.uni-muenster.de/
- In case of dynamic IPv4 address
  - ppp (ipv6 over ppp) tunnel (pptp, l2tp) possible
  - sitctrl (linux <-> linux)
  - atncp (*NIX), http://www.dhis.org/atncp/

Stateless Autoconfiguration

- Address space is split in two 64bit halves
  - Upper 64bit ’2001:780:44:1100:’ used to specify a network segment (/64)
  - Lower 64bit ’204:61ff:fe5c:74b9’ used to specify node within segment
  - Lower 64bit are generated from 48bit mac address with ’fffe’ in the middle
- Potential Problem: Privacy
- IETF Solution: RFC3041 "Privacy Extension"
  - uses additional ’alias’ IPv6 adresses that are created randomly and only valid for hours/days

DNS and IPv6

- Forward resolval (hostname->address)
  - IPv4 uses "IN A" record
  - IPv6 uses "IN AAAA" record
  - A particular hostname can have both A and AAAA

- Reverse resolval (address->hostname)
  - Uses ".ip6.arpa." suffix
  - Uses hexadecimal instead of decimal notation
    - 4.4.0.0.0.0.0.0.0.8.7.0.1.0.0.2.ip6.arpa.

BSD Sockets API and IPv6

- new structures
  - in_addr has become in6_addr
  - sockaddr_in has become sockaddr_in6
- new API's like getaddrinfo are compatible with ipv6 and ipv4
- portable applications use sockaddrr_storage and don't make assumptions about it's size

Configuration under Linux

- Router/Gateway
  - Runs radvd or zebra for for sending router advertisements
- Client
  - Just has to load "ipv6" module and configure interface up
  - Receives prefix-advertisements(s) and autoconfigures address

IPv6 option headers

- New concept of option header
  - Any number of option headers between l3 and l4 header
  - With one exception only processed ad sender and receiver

- Defined option headers
  - Hop-by-hop options (processed by every node)
  - Destination options
  - Routing header
  - Fragment header
  - Authentication (AH)
  - Encapsulating Security Payload (ESP)

IPv6 specific security issues

- hop-by-hop options header
  - should be filtered out at typical internet gateway
- routing header
  - should be filtered out like IPv4 loose source / record route
- ICMPv6
  - has to be allowed for neighbour discovery to work

IPv6 specific security issues

- iptables -> ip6tables changes
  - matching of ah/esp
    - not by -p !
  - matching of fragments
    - not by -f !
  - no connection tracking in mainline kernel yet
    - existing ip6_conntrack patchces (deprecated)
      - code duplication
      - no interaction between ip_conntrack/ip6_conntrack
    - existing nf_conntrack patches
      - one code base to rule them all
      - ipv4 and ipv6 plugins
      - l3 independent tcp and udp modules independent
      - l3 independent helpers
  - BUT: no NAT as of now :(

Further Reading

- http://www.ipv6-net.org/ (deutsches IPv6 forum)
- http://www.6bone.net/ (ipv6 testing backbone)
- http://www.freenet6.net/ (free tunnel broker)
- http://hs247.com/ (list of tunnel brokers)

- http://www.bieringer.de/ (ipv6 for linux)
- http://www.linux-ipv6.org/ (improved ipv6 for linux)
- http://www.kame.net/ (ipv6 for *BDS)
- http://www.join.uni-muenster.de/ (ipv6 at DFN/WiN)

- http://www.gnumonks.org/ (slides of this presentation)

- And of course, all relevant RFC’s