

IPv6 Lesson 1: protocol and implementation status

Jun-ichiro itojun Hagino, Ph.D
KAME project/ijl@lab
itc@junlab.net

Tutorial overview

- Jan 8, 2003
 - IPv6 protocol itself (packet format, ...)
 - Impact/relationship to other internet protocols (such as TCP/HTTP)
 - Implementation status
- Jan 15, 2003
 - Deployment status
 - Operational issues, clues, tips
 - UNIX socket API programming for IPv6

Internet protocol version 6 (IPv6)

Prerequisite

- Knowledge of
 - IPv4, TCP, UDP, routing in the internet
- SOI courses
 - IPv6 tutorial by kazuo/itojun
- Books (not mandatory to read, but are informative)
 - Huitema, "IPv6: the new internet protocol"
 - Stevens, "TCP/IP illustrated vol.1"
 - Stevens, "UNIX network programming"

What is IPv6? (1)

- The answer to IPv4 address space problem
 - IP address changed from 32bit to 128bit
 - 32bit is smaller than worldwide human population
 - IPv4 address shortage is a bondage to new application areas
 - IPv4 addresses will run out around 2010?
 - $4 \times 10^9 < 3.4 \times 10^{38}$
 - 4 billion -> 340 undecillion
- A new starting point
 - IPv4: used for 20 years, minimal implementation lacks almost everything
 - IPv6: new standard features like multicast, PMTU, IPsec, autoconfig
 - Leverage the availability of advanced technologies
- An answer to routing table size issue
 - IPv4: portable address (<-CIDR), result in 50K entries
 - IPv6: full CIDR + well-structured address assignment
 - restricts DFZ routes to 8K

What is IPv6? (2)

- Say a long good-bye to NAT, say hi again to end-to-end model
 - Has been used as temporary cure for address space issue
 - NAT breaks bidirectional communication
 - NAT is a bondage to application protocol designers
 - NAT does not let multicast/Ipsec through
 - NAT does not work with proprietary protocols
 - NAT does not improve security
 - NAT box is a single point of failure
 - If you reboot NAT box, all connection will be gone
 - NAT is a barrier to scalability of the Internet as a whole

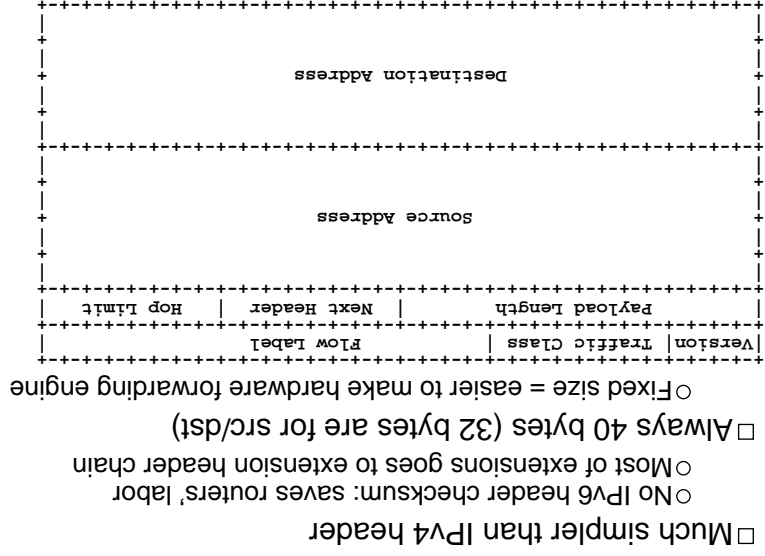
IPv6 INCORRECT rumors

- Fixes all Internet problems
 - How? :-P
 - Noone uses it
 - There are so many users in Japan (ISP IPv6 service is available as well)
 - Asia and Europe are more active, US seems to be more optimistic about IPv4
- QoS problem is solved
 - Fields are reserved for QoS
 - QoS technology is still in its infancy, and needs more time to be stabilized
- Security problem is solved
 - IPsec is mandatory
 - NAT will be gone, so IPsec can be used everywhere
 - However, IPsec itself has very difficult issue
- IPv6 is not necessary since we have NAT
 - BIG NO!
- Need hardware upgrade
 - In most cases, no. Software upgrade is enough

IPv6 features

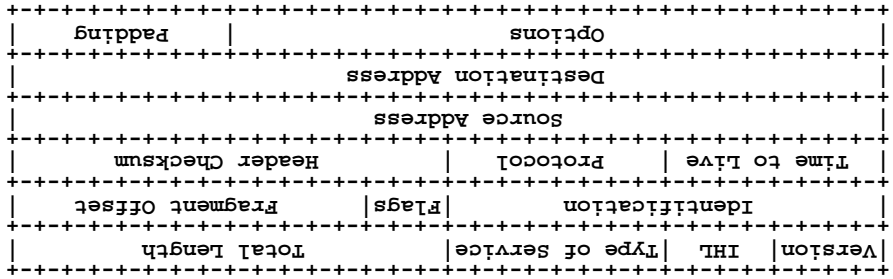
- Can coexist with IPv4 network
 - There's no flag day, you can use existing devices
 - "Dual stack" - devices that can use IPv4/IPv6 at the same time
- Only needs software update in most cases
 - You do not need to buy new routers
- Vast amount of address space
 - 2^{v32} or 2^{v13} route entries in ISP
 - 2^{v64} nodes on a subnet
 - 2^{v16} subnets to a site
- Aggregation friendly
 - Core routes are limited to 8192 (2^{v13}) by addressing architecture
 - No portable address available: multihoming?
- Designed for less operation cost in routers
 - Mandates PMTUD, no fragmentation in intermediate routers
- Autoconfiguration works beautiful
 - No server necessary, no wacky state management
 - Autoconfig available everywhere: IPv4 autoconfig required DHCP server

IPv6 header



- Much simpler than IPv4 header
- No IPv6 header checksum: saves routers' labor
- Most of extensions goes to extension header chain
- Always 40 bytes (32 bytes are for src/dst)
- Fixed size = easier to make hardware forwarding engine

IPv4 header



- Many fields
- TOS: left unused in most cases
- IP header checksum: high cost to maintain, cost/benefit?
- IPv4 options: not really used
- Usually 20 bytes (8 bytes are for src/dst)

Upper layer protocols

- Upper layer
 - ICMPv6: ICMP for IPv6
 - Very similar
 - TCP, UDP: same as IPv4
 - Only pseudo-header checksum is different
 - Protocols on TCP/UDP: same
 - HTTP, FTP will work just like it was on IPv4
 - If we embed IP address into upper-layer protocol header, we need to modify it (FTP)

How to write IPv6 addresses

- 16 groups of 4-digit hexadecimal
 - 3fe:0507:0000:0001:0200:86ff:fe05:80fa
 - 0000:0000:0000:0000:0000:0000:0000:0001
- You can omit starting "0" in each of the groups
 - 3fe:507:0:1:200:86ff:fe05:80fa
 - 0:0:0:0:0:0:1
- In only one place, you can use "::" to denote continuous "0"
 - 3fe:507::1:200:86ff:fe05:80fa
 - ::1
- Usually you do not write numeric IPv6 addresses, use DNS names

Modifications to upper layer protocols

- FTP
- PORT, PASV passes IPv4 address, and does not support other types
- EPSV, EPRM command - RFC2428
- HTTP
- host:port form is ambiguous if host is an IPv6 address
- [host]:port notation - RFC2732

IPv4 packet example

TCP packet toward port 22, in IPv4

```
16:09:25.113204 202.232.15.102.63472 > 202.232.15.98.22: P 20:40(20) ack 21
win 17520 <nop,nop,timestamp 245605 38398> (ttl 64, id 13795)
4500 0048 35e3 0000 4006 9034 cae8 0f66
cae8 0f62 f7f0 0016 8520 7612 1672 89e4
8018 4470 4060 0000 0101 080a 0003 bf65
0000 95fe 0000 000a 0eeF be30 a72a 41f1
7be1 2f9b 3ccd b5b0
16:09:25.113364 202.232.15.102.63472 > 202.232.15.98.22: P 20:40(20) ack 21
win 17520 <nop,nop,timestamp 245605 38398> (ttl 64, id 13795)
4500 0048 35e3 0000 4006 9034 cae8 0f66
cae8 0f62 f7f0 0016 8520 7612 1672 89e4
8018 4470 4060 0000 0101 080a 0003 bf65
0000 95fe 0000 000a 0eeF be30 a72a 41f1
7be1 2f9b 3ccd b5b0
```

IPv6 packet example

TCP packet toward port 22, in IPv6

```
15:43:17.093542 3ffe:507:0:1:200:86ff:fe05:80fa.49226 >
3ffe:507:0:1:260:97ff:fe07:69ea.22: . ack 1093 win 17519 <nop,nop,timestamp
242469 35276> [flowlabel 0x62073] (len 32, hlim 64)
6006 2073 0020 0640 3ffe 0507 0000 0001
0200 86ff fe05 80fa 3ffe 0507 0000 0001
0260 97ff fe07 69ea c04a 0016 3ee4 92b2
29d4 2455 8010 446f 80da 0000 0101 080a
0003 b325 0000 89cc
15:43:17.093648 3ffe:507:0:1:200:86ff:fe05:80fa.49226 >
3ffe:507:0:1:260:97ff:fe07:69ea.22: . ack 1093 win 17519 <nop,nop,timestamp
242469 35276> [flowlabel 0x62073] (len 32, hlim 64)
6006 2073 0020 0640 3ffe 0507 0000 0001
0200 86ff fe05 80fa 3ffe 0507 0000 0001
0260 97ff fe07 69ea c04a 0016 3ee4 92b2
29d4 2455 8010 446f 80da 0000 0101 080a
0003 b325 0000 89cc
```

IPv6 extension headers

- Hop-by-Hop Options
 - Routers will look at it
 - Routing
 - Source routes
 - Fragment
 - Fragmentation, ONLY at originating node
 - Destination Options
 - Pass info to final destination
 - IPsec (AH, ESP)
 - Header chain
 - Header parsing code becomes somewhat different from IPv4
- IPv6(nxt=TCP) TCP payload
IPv6(nxt=routing) Routing(nxt=TCP) TCP payload
IPv6(nxt=fragment) Fragment(nxt=TCP) TCP payload

IPv6 extension headers example (1)

IPv6 echo request/reply, without AH

```
16:14:31.750102 ::1 > ::1: icmp6: echo request (len 16, hlim 64)
6000 0000 0010 3a40 0000 0000 0000 0000
0000 0000 0001 0000 0000 0000 0000 0000
57ea e137 d471 0b00
16:14:31.750155 ::1 > ::1: icmp6: echo reply (len 16, hlim 64)
6000 0000 0010 3a40 0000 0000 0000 0000
0000 0000 0001 0000 0000 0000 0000 0000
57ea e137 d471 0b00
```

IPv6 extension headers example (2)

IPv6 echo request/reply, with AH

```
16:14:57.060181 ::1 > ::1: AH(spi=9999, sumlen=16, seq=0x7): icmp6: echo
request (len 40, hlim 64)
6000 0000 0028 3340 0000 0000 0000 0000
0000 0000 0001 0000 0000 0000 0000 0000
0000 0000 0000 0000 0001 3a04 0000 0000
270f 0000 0007 5a3c 2d0d efbf 3370 d029 1ed5
8000 c3a3 dd02 0300 71ea e137 88ea 0000
16:14:57.060299 ::1 > ::1: icmp6: echo reply (len 16, hlim 64)
6000 0000 0010 3a40 0000 0000 0000 0000
0000 0000 0001 0000 0000 0000 0000 0000
71ea e137 88ea 0000
```

Lower layer

- Ethernet protocol type: different from IPv4
 - IPv4: 0x0800
 - IPv6: 0x86dd
- Hardware address resolution
 - IPv4: medium dependent, ARP for ethernet
 - Simple protocol
 - IPv6: medium independent, NDP on ICMPv6
 - Manages more state than ARP, more complex
 - Implements DAD, Duplicate Address Detection
- Multicast address mappings
 - Take lowermost 4 bytes from IPv6 address, and embed to MAC addr
 - 33:33:xx:yy:zz:uu
 - No use of broadcast
 - Link-layer broadcast will wake everyone up on the medium

Ethernet encapsulation example (1)

IPv4 broadcast icmp echo request and icmp echo reply

```
15:46:30.651062 0:0:f8:5:86:5:80:fa ff:ff:ff:ff:ff:ff 0800 98: 202.232.15.102 >
202.232.15.103: icmp: echo request (ttl 255, id 9961)
4500 0054 26e9 0000 ff01 e021 cae8 0f66
cae8 0f67 0800 b14d 02a5 0000 c6e3 e137
a7ee 0900 0809 0a0b 0c0d 0e0f 1011 1213
1415 1617 1819 1a1b 1c1d 1e1f 2021 2223
2425 2627 2829 2a2b 2c2d 2e2f 3031 3233
3435 3637

15:46:30.651377 0:0:f8:5:69:42 0:0:86:5:80:fa 0800 98: 202.232.15.100 <
202.232.15.102: icmp: echo reply (ttl 255, id 2369)
4500 0054 0941 0000 ff01 fdcc cae8 0f64
cae8 0f66 0000 b94d 02a5 0000 c6e3 e137
a7ee 0900 0809 0a0b 0c0d 0e0f 1011 1213
1415 1617 1819 1a1b 1c1d 1e1f 2021 2223
2425 2627 2829 2a2b 2c2d 2e2f 3031 3233
3435 3637
```


IPv6 autoconfiguration

- Stateless address autoconfiguration
 - No resource management thanks to address architecture
 - Routers advertise info about the subnet
 - Hosts receive the information and configures itself
- DHCPv6
 - Stateful, needs server everywhere
- Stateless address autoconfiguration is much easier, and will be available everywhere

IPv6 addressing architecture (2)

- Scoped addresses
 - Unique only in the "scope"
- Link-local: fe80::/10
 - Addresses used for address resolution and bootstrap
 - Unique only on single link
- Site-local: fec0::/10
 - Similar to 10.0.0.0/8, for use within sites
 - Unique only in single site
 - Usage model/validity is still under discussion
- Multicast: ff00::/8
 - Multicast addresses are scoped as well
 - IPv4 - controlled by TTL, a whole mess
 - ff02::/16 - link, ff05:: - site

IPv6 autoconfig details

- End host transmits router solicitation
- "I want to know the configuration for the subnet"
- Router transmits router advertisement
- "Okay, here's information on subnet"
- End host configures itself

```
15:40:16.590444 fe80::200:86ff:fe05:80fa > ff02::2: icmp6: router
solicitation (src lladdr: 0:0:86:5:80:fa) (len 16, hlim 255)
6000 0000 0010 3aff fe80 0000 0000 0000
0200 86ff fe05 80fa ff02 0000 0000 0000
0000 0000 0000 0002 8500 6d2e 0000 0000
0101 0000 8605 80fa
15:40:16.819647 fe80::260:97ff:fe07:69ea < ff02::1: icmp6: router
advertisement (chlim=64, router_ltime=1800, reachable_time=30000,
retrans_time=1000) (src lladdr: 0:60:97:7:69:ea) (mtu: mtu=1500) (prefix
info: IA valid_ltime=3600000, preferred_ltime=3600000,
prefix=3ffe:507:0:1::/64) (len 64, hlim 255)
6000 0000 0040 3aff fe80 0000 0000 0000
0260 97ff fe07 69ea ff02 0000 0000 0000
0000 0000 0000 0001 8600 4625 4000 0708
0000 7530 0000 03e8 0101 0060 9707 69ea
0501 0000 0000 05dc 0304 40c0 0036 ee80
0036 ee80 0000 0000 3ffe 0507 0000 0001
```

IPv6 and DNS

- IPv6 in DNS payload
 - AAAA record: similar to A record
 - PTR record: use "ip6.arpa." tree
 - "ip6.int" tree has been used, transition ongoing
- IPv6 in DNS transport
 - BIND8 supports queries via IPv4 TCP/UDP only
 - BIND9 supports queries over IPv6 TCP/UDP as well

IPv6 cloud and IPv4 ocean

- At this moment, internet infrastructure uses IPv4
 - How IPv6 sites communicate?
 - -> tunneling
- RFC2893 defines IPv6-over-IPv4 tunneling
 - Can be used just like IPv6 p2p link
 - We run IPv6 routing protocol on top of it

IPv6 in real life network

- IPv6-capable node needs to be able to talk with IPv4-only node
 - Need to click www.yahoo.com...
- IPv4/v6 dual stack
 - Ethernet protocol type is different - they can coexist
 - IPv4/v6 dual stack node can speak with IPv4-only node
 - You can even use NAT for IPv4 communication
 - NAT network for IPv4, global network access for IPv6
- Translator (for IPv6 only nodes)
 - Application layer: web proxy, fwtk, sendmail
 - TCP layer: socks, fwtk
 - IP layer: header translation
- Has same problem with NAT (not end-to-end, single point of failure)

Tunneling example

□ RFC2893 tunneling

```
16:20:56.718074 210.163.17.131 > 202.232.15.98:
3ffe:507:102:0:260:8cfe:b43b.22 >
3ffe:50a:fff:100:210:4bfe:fe0a:b89.1021: p 249:357(108) ack 380 win 17080
[Flowlabel 0x2c8df] (len 128, hlim 64) (ttl 20, id 8872)
4500 00bc 22a8 0000 1429 c500 d2a3 1183
ca88 0f62 6002 c8df 0080 0640 3ffe 0507
0102 0000 0260 8cfe fe0a b43b 3ffe 050a
ffff 0100 0210 4bfe fe0a 0b89 0016 03fd
e7a9 375e 1fee a61c 5018 42b8 6270 0000
0000 0062 8d0a 2d88 6d8b 9ee9 4fdd a123
36a2 5b2d fef2 a3ec 882e 543b 4cbb 5fe8
8d31 8257 a02d 345b cbf6 491f 476e df02
1526 91fa 5e59 52fb f970 209a 24cc 4a45
f4d6 5e9e 81df 0c38 9514 27a5 2de5 b55c
2d41 c98f 06d2 2e06 4830 6990 dff2 d99b
3630 9a69 e0c7 476a 67be 0a9e
```

Tiny demo

```
$ ifconfig sml
sml: flags=8863<UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST> mtu 1500
media: Ethernet 10baseT
inet 0.0.0.0 netmask 0xffff0000 broadcast 255.255.255.255
inet6 fe80:1::200:86ff:fe05:80fa prefixlen 64
inet6 3fe:501:4819:2000:200:86ff:fe05:80fa prefixlen 64
$ ifconfig lo0
lo0: flags=8009<UP,LOOPBACK,MULTICAST> mtu 32976
inet 127.0.0.1 netmask 0xffff0000
inet6 fe80::2::1 prefixlen 64
inet6 ::1 prefixlen 128
$ ping6 ::1
PING6(56=40+8+8 bytes) ::1 --> ::1
16 bytes from ::1, icmp_seq=0 hlim=64 time=0.137 ms
16 bytes from ::1, icmp_seq=1 hlim=64 time=0.112 ms
$ telnet localhost
Trying ::1...
Connected to localhost.
Escape character is '^['.
FreeBSD (banana.kame.net) (tty)
login:
```

IPv6 Implementations

IPv6 implementations (1)

□ Layer 2

- Nothing tricky, you just need normal switch/hub
- Some of ethernet card/driver have problem with multicast
- Multicast MUST work properly

□ Routers (like Cisco)

- Many vendors are shipping IPv6 in their firmware
 - shipping: Cisco, Juniper, Hitachi, Fujitsu, NEC, Yamaha, IJ, Allied-Telesys
 - testing: Extreme, Foundary, ...
- Dialup servers/DSL devices are behind the schedule

IPv6 implementations (2)

- End node
 - Linux: kernel is IPv6 ready (2.1 or later)
 - Need to reinstall most of the userlands
 - Status depends on the distribution you are using
 - BSD: all *BSDs are ready
 - Windows NT: MS Research release
 - Windows 95/98: Hitachi Toolnet6, Trumpet Fantare
 - Windows XP: disabled by default (for developers), near-future service pack will enable it
 - Solaris: Solaris 8 is ready
 - MacOS X: 10.2 is IPv6 ready (for developers), 10.3 will be fully IPv6-ready
 - DEC/Compaq, IBM AIX
- Many of the implementation can become routers
- IPsec on IPv6
 - KAME, NRL, MSR IPv6, IBM AIX

IPv6 implementations (3)

- Applications: anything you want!
 - apache bind8 squid mozilla lynx graal
 - kaffe(java interpreter) perl ruby python
 - icecast ...
- Routing daemons
 - Similar to IPv4
 - As we have limited set of algorithms
 - RIPng: similar to RIP
 - BGP4+: multiprotocol version of BGP
 - OSPFv6 (OSPFv3): similar to OSPF
- Advanced features
 - There are inherently difficult problem remains
 - Experimental code is available
 - diffserv, IPsec policy control, mobile-ipv6, multihoming

- IPv4: BSD UNIX kernel was the reference
- IPv6: documents comes first, no de-facto reference code
 - Interop, availability, education...
- Goals: provide IPv6/IPsec reference code to the world
 - BSD license: can be incorporated into product
 - Incorporate experimental protocols and recent technologies
 - mobile-ip4/6, IP over satellite, traffic shaping/diffserv
- 10 core members from 8 companies, including IJ
- Suborganization of WIDE consortium

BSD implementation - KAME project

- Advanced application needs more IPv6 deployment to start with
- Home appliances
 - Lots of interests, research ongoing, some real products
- Cellphones
 - Nokia/Ericsson are pushing backbone using IPv6, handset using IPv6/VoIP
- Other applications
 - "IP in every taxicab" experiment (Nagoya Japan)
 - IPv6 multicast video streaming events

Home appliances and other devices

Using KAME code

- Most applications are seamless
 - ftp, telnet, rsh, ssh, ...
- IPv6 becomes visible only when you use numeric addresses
 - ping6 :: 1
 - telnet :: 1
- Socket APIs are updated with `AF_INET6` and `sockaddr_in6`
 - Basic applications are trivial
 - Routing socket, `setsocopt`, and other tricky parts need update

KAME status

- IPv6
 - Latest spec + corrections to spec
 - IPsec + IKE, all made in Japan
 - Good note PC support, flow control, ATM PVC
 - Routing daemons
 - IPv6 multicast
 - Bunch of IPv6-ready applications (ports/pkgsrc)
 - Various physical medium
 - Ethernet, ATM PVC, sync ppp, async ppp, satellite
 - mobile-ipv6, queuing framework (diffserv), and other interesting things
- Integrated into NetBSD, OpenBSD, FreeBSD and BSD/OS
 - For normal use, plain NetBSD/OpenBSD/FreeBSD installation is okay
- Available from `ftp.kame.net`
 - Weekly SNAP kit: for those who want more hacking than stability

Homework - Windows

- Make sure to use Windows XP (95, 98, NT are difficult)
- invoke "ipv6 install" from the command line
- various command line tools should become IPv6 capable
- Try to click <http://www.kame.net/>

Homework - Overview

- Configure an IPv6 node and IPv6 network
- Report
 - what you have done
 - what was it like compared to IPv4 operations
 - even if you feel no difference, it's okay, just describe it
 - 2-4 pages
 - plaintext or PDF please - NO microsoft word (*.doc)
- For testing: <http://www.kame.net/> is available over IPv4 and IPv6
 - If you click the page over IPv6, the turtle will dance
- Depends on the implementation you are using
 - If you don't have any of these, install NetBSD 1.6 onto your machine

Homework - Macintosh

- Make sure to use MacOS 10.2
- invoke Terminal (Applications/Utilities)
- try using /usr/sbin/ip6 and /usr/sbin/ip6config
- Install IPv6-capable SSH
 - `ftp://ftp.kame.net/pub/kame/misc/OpensSH-3.4p1-IPv6.dmg`
- restart SSH service ("Remote Login" in "Sharing" system preference)
- see IPv6 service is enabled with "netstat -an"

Homework - Linux

- Enable IPv6 services by `/etc/inetd.conf`
 - "inet :::1" or "ssh :::1" should be possible even without IPv6 external connectivity
 - Configure 6to4 network
 - Install Mozilla and try `http://www.kame.net/`
- <http://www.bieringer.de/linux/IPv6/status/IPv6+Linux-status-distributions.html>

URLs of interest

<http://www.wide.ad.jp/>
<http://www.v6.wide.ad.jp/>
<http://www.kame.net/>
<http://www.6bone.net/>
<http://www.ipv6.org/>

Homework - BSD UNIX

- Use the latest version possible (FreeBSD 4.7, NetBSD 1.6, OpenBSD 3.2)
- Enable IPv6 services by `/etc/inetd.conf`
- "telnet ::1" or "ssh ::1" should be possible even without IPv6 external connectivity
- Configure `6to4` by looking at <http://www.netbsd.org/Documentation/network/ipv6/>
- Install Mozilla and try `http://www.kame.net/`