What's up in the Linux IPv6 Stack

Hideaki YOSHIFUJI

Keio University
USAGI/WIDE Project

http://www.linux-ipv6.org/materials/200801-LCA2008/

About Me

- Hideaki YOSHIFUJI
  - Core member of **USAGI Project**
  - Co-maintainer of **Networking [IPv4/IPv6]** area
  - Assistant professor, Keio University
    - Networking, especially IPv6, “ubiquitous” computing
    - Linux, open source

    etc...

Table of Contents

Table of Contents

Overview

- Linux IPv6 Stack
  - Kernel, libraries (glibc/uclibc) and tool (iproute2, iputils etc.)
- Kernel part was merged in 1996 (by Pedro Roque).
- USAGI Project was founded in 2000.
  - Universal Playground for IPv6
  - To promote development of Linux IPv6 stack for practical use

About This Talk

- USAGI Project is 7 years old
- This talk aims to...
  - overview highlights of Linux IPv6 stack
    - Features: core, Netfilter, IPsec, Mobile IPv6 and transition mechanisms
    - Quality

and

  - discuss missing pieces and future directions

Table of Contents

Supported Features - Highlights

- Core
- Netfilter
- IPsec
- Mobile IPv6
- Transition Mechanisms

Supported Features - Highlights

- Core
- Netfilter
- IPsec
- Mobile IPv6
- Transition Mechanisms

IPv6 Core

- IPv6 [RFC2460]
  - Deprecation of Routing Header Type 0 [RFC5095]
    - To protect from DoS attack
      - `ping6 r1 r2 r1 r2 .... dest`
- ICMPv6 [RFC4007]
- Neighbor Discovery [RFC4861]
- MLDv1 [RFC2710], MLDv2 [RFC3810]

IPv6 Core (cont'ed): Address Configuration (1)

- Stateless address auto-configuration [RFC4862]
  - Optimistic DAD [RFC4429]
    - Allow to use a likely-unique address before DAD has been completed
  - ND option notification through netlink
    - For RDNSS [RFC5006]
  - Daemons (for sending RAs)
    - **radvd**, maintained by Pekka Savola (litech.org)
    - quagga

IPv6 Core (cont'ed): Address Configuration (2)

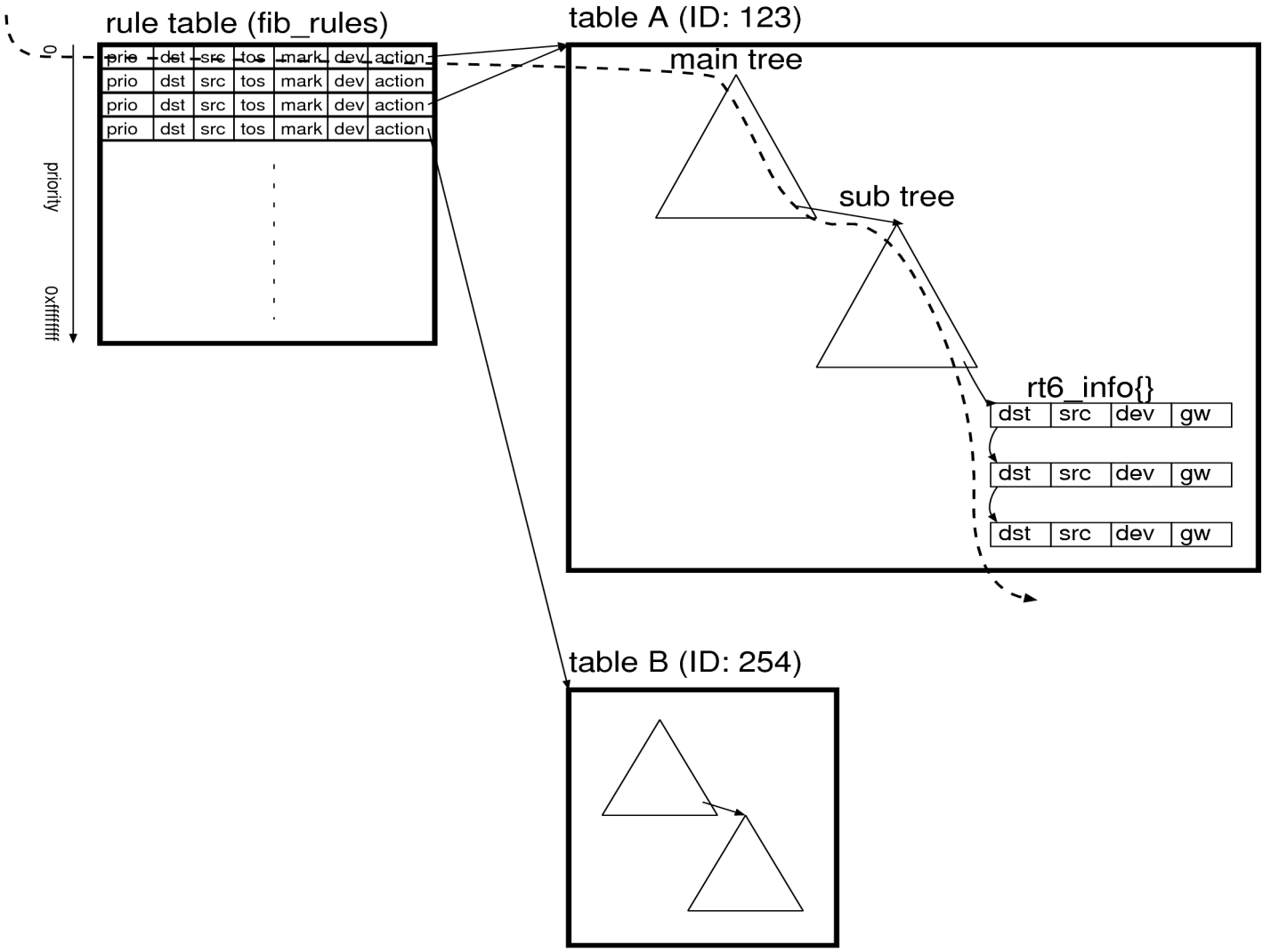
- Stateful address configuration
  - DHCPv6 [RFC3315]
    - **wide-dhcpv6**
      - http://sourceforge.net/projects/wide-dhcpv6/
    - dhcpv6
      - https://hosted.fedoraproject.org/dhcpv6/
    - Dibbler
      - http://klub.com.pl/dhcpv6/

IPv6 Core (cont'ed): Address and Route selection:

- Policy routing: “fib_rule” subsystem
- Default Router Preference and More-Specific Routes [RFC4191]
  - Radvd also supports
- Default Address Selection [RFC3484]
  - Default extended: ULA etc.
  - 2.6.25 or later: configurable labels
    - ip addrlabel subcommand
  - glibc 2.5 or later (preferably 2.7 or later)

IPv6 Core (cont'ed): Policy Routing

- Built on top of fib_rules infra-structure
- Configura ble via “ip rule” sub-command



IPv6 Core (cont'ed):
Policy Routing vs Address Selection

- A policy may depend on source address.
- Source address selection is decided after routing decision (especially interface decision), if upper layer has not specified one.
- FIB_RULE_FIND_SADDR flag
  - Ignore the source address in the rule and find a best route.
  - Decide source address and check if it is okay for the rule used.

IPv6 Core (cont'ed): Management

- Statistics [RFC4293,...]
  - Per-interface statistics (for IPv6)
  - Net-snmp: patches available
- ICMP Node Information Queries [RFC4620]
  - Server: implemented in a daemon: “ninfod”
  - Client: implemented in ping6 utility
    - usagi-tools
    - iputils (not yet)

IPv6 Core (cont'ed): Tunnels

- Tunnels (will be discussed later)
  - “sit” (IPv6 in IPv4)
  - “ip6tnl” (ip6_tunnel): IPv{4,6} in IPv6
  - managed by iproute2 tool (tunnel subcommand)

IPv6 Core (cont'ed): Socket APIs

- Basic Socket API [RFC3493]
  - Core functions
    - socket(2), connect(2), bind(2), ...
  - Name resolution functions
    - getaddrinfo(3), getnameinfo(3), ...
- Advanced Socket API [RFC2292,3542]
  - Raw socket etc.

IPv6 Core (cont'ed): Basic Socket API

- Discussion about bind(2) and packet delivery semantics
- Question
  - Is it okay to allow a socket to bind on a port which bound by another socket?
  - To which socket should we deliver a packet if we have multiple sockets on the same port?
    - IPv4 vs IPv4 (specific vs any)
    - IPv4 vs IPv6 (specific vs any, any vs any)
    - IPv6 vs IPv6 (specific vs any)

IPv6 Core (cont'ed): Basic Socket API: Port Space

- Co-existence of IPv4 and IPv6 socket on the same port is not allowed by default.
  - Port space is shared between IPv4 and IPv6.
  - With IPV6_V6ONLY socket option set, port space is partially split.
- On some systems, port space is shared but automatically allows IPv4 socket to bind on the same port.
- On other systems, port space is split.

IPv6 Core (cont'ed): Advanced Socket API

- Advanced Socket API [RFC2292,3542]
  - 3542 options have different semantics
    - IPV6_HOPOPTS (2292)
    - IPV6_HOPOPTS and IPV6_RECVHOPOPTS (3542)
  - 2292 options renamed
    - IPV6_2292HOPOPTS

IPv6 Core (cont'ed): Advanced Socket API Example

To receive hop-by-hop options:

```
    if (
#ifdef IPV6_RECVHOPOPTS /* Check if we have new API */
        setsockopt(s, IPPROTO_IPV6, IPV6_RECVHOPOPTS,
                   &on, sizeof(on)) < 0
#ifdef IPV6_2292HOPOPTS /* Try Old API for old kernel */
        && setsockopt(s, IPPROTO_IPV6, IPV6_2292HOPOPTS,
                      &on, sizeof(on)) < 0
#endif
#else /* If new API unavailable, try old one */
        setsockopt(s, IPPROTO_IPV6, IPV6_HOPOPTS,
                   &on, sizeof(on)) < 0)
#endif
    )

        perror("setsockopt(HOPOPTS)");
```

Otherwise, you will suck...

Supported Features - Highlights

- Core
- Netfilter
- IPsec
- Mobile IPv6
- Transition Mechanisms

Netfilter

- nf_conntrack subsystem
  - General framework for connection tracking
    - for stateful filtering and NAT
  - Superseded ip_conntrack for IPv4
- ip_tables / ip6_tables abstraction layer (x_tables)
- Extension module API (for IPv4 and IPv6)
  - Got easier to add new IPv6 matches / targets
  - More 14 modules newly supports IPv6 (1.4.0)

Netfilter (cont'ed)

- Enhancements
  - Fragment handling
  - Elimination of skb_linearize()
  - Introduced a single method to find a specific header (ipv6_find_hdr())

Supported Features - Highlights

- Core
- Netfilter
- IPsec
- Mobile IPv6
- Transition Mechanisms

IPsec: Features

- 2.6 supports IPsec [RFC2401]
  - “XFRM” / stackable destination architecture
    - Supports both IPv4 and IPv6
  - Many crypto algorithms (cryptoapi)
  - Inter-family IPsec
    - Helps IPv6 deployment
  - BEET (Bound End-to-End Tunnel) mode
- Many methods for key exchange available
  - IKEv1, IKEv2, KINK

IPsec: Key Exchange

- IKEv1 [RFC2409]
  - **Racoon** (ipsec-tools)
  - Pluto (strongSwan/Openswan)
  - Racoon2 (Racoon2 Project)
- IKEv2 [RFC4306]
  - OpenIKEv2 (OpenIKEv2 Project)
  - **Racoon2** (Racoon2 Project)
  - Charon (strongSwan)
  - Ikev2 (IKEv2 Project)

IPsec: Key Exchange (cont'ed)

- KINK [RFC4430] (Kerberos)
  - Racoon2 (Racoon2 Project)

- Basic key exchange features are likely supported by those IKE applications.

IPsec: Missing pieces

- AHv2 [RFC4302], ESPv3[RFC4303]
  - Extended sequence number
- PFP flag [RFC4301]

Supported Features - Highlights

- Core
- Netfilter
- IPsec
- Mobile IPv6
- Transition Mechanisms

Mobile IPv6

- Mobility support for IPv6 [RFC3775,...]
- MIPL (Mobile IPv6 for Linux) 2
  - Jointly developed by Helsinki University of Technology (HUT) and USAGI Project
  - Kernel + Daemon
    - Kernel: packet processing
    - Daemon: signaling processing
  - Avoiding patchy and intrusive implementation
    - Kernel changes required is relatively small

Mobile IPv6 (cont'ed): UMIP (USAGI-Patched MIPL2)

- Enhancements / patches for MIPL2 release to support latest kernel and features
  - Mobile IPv6 [RFC3775]
  - MIGRATE [draft-sugimoto-mip6-pfkey-migrate]
- Most of changes are ready in main-line kernel.
- http://www.linux-ipv6.org/memo/mipv6/

Mobile IPv6 with IPsec

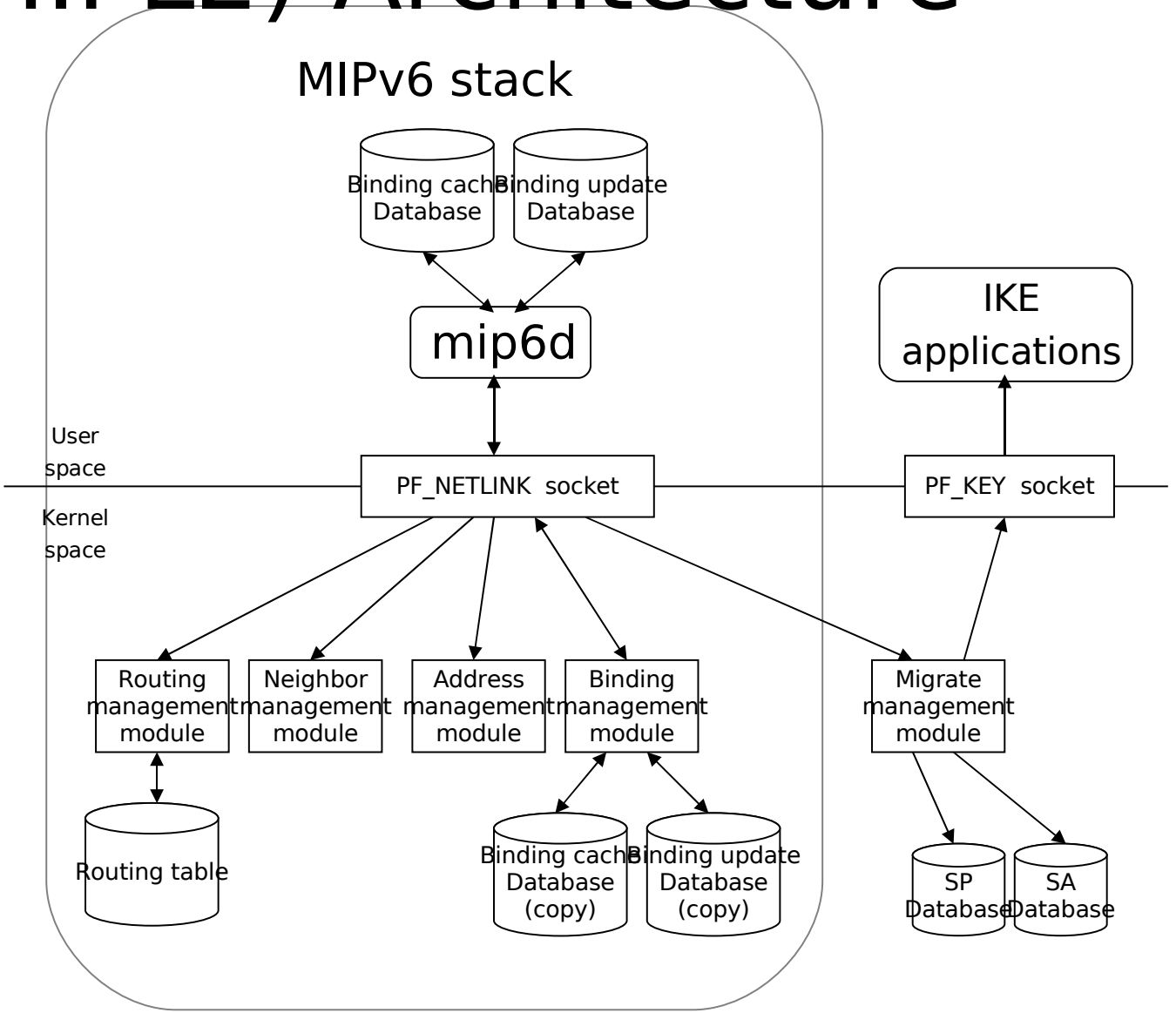
- RFC3776,4877
- draft-sugimoto-mip6-pfkey-migrate
  - Interface between Mobile IPv6 subsystem and IPsec/IKE subsystem
    - MIGRATE
      - 2.6.21 or later
      - Patches for Racoon (IKEv1) and Racoon2 (IKEv2) available
    - Packet extension
      - implementation on Linux cannot be straight-forward...

Mobile IPv6: UMIP (MIPL2) Architecture

Mobility functions are maintained by a daemon (mip6d).

PF_NETLINK is used for API

MIGRATE (through PF_KEY) is used to update endpoint address of IPsec tunnel when MN moves.



Supported Features - Highlights

- Core
- Netfilter
- IPsec
- Mobile IPv6
- Transition Mechanisms

Transition Mechanisms

- Basic tunnels
  - 6-in-4 tunnel (sit)
  - 4-in-6 tunnel (ip6_tunnel)
- Inter-family IPsec tunnel
  - 6-in-4 / 4-in-6

Transition Mechanisms (cont'ed)

- Teredo (Tunneling IPv6 over UDP through NATs) [RFC4380]
  - Miredo (by Rémi Denis-Courmont)
- ISATAP (Intra-Site Automatic Tunnel Addressing Protocol) [RFC4214]
  - 2.6.24 or later, by Fred L. Templin
  - with daemon's help (still under development)
- NAT-PT [RFC2766,...]
  - Userspace implementation available.

Table of Contents

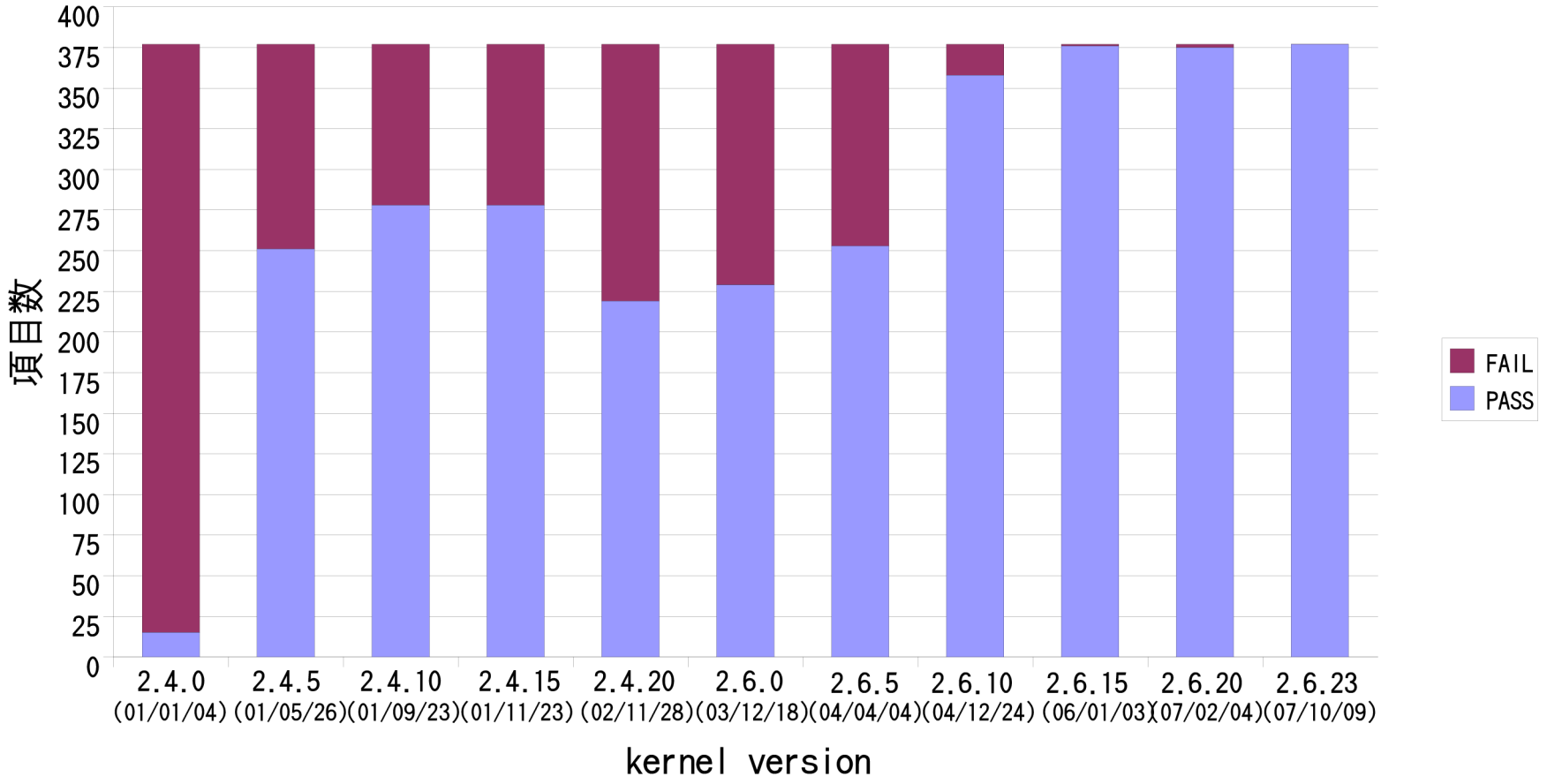
Quality of the Stack

- Quality measurement
  - TAHI Conformance Test
  - IPv6 Ready Logo (http://www.ipv6ready.org)
    - Several sets got certified
      - 2.6.11-rc2 (with USAGI radvd (sV6READYP1-20050121_20050124) for Router)
        - Phase-1: Host
        - Phase-1: Router
      - 2.6.15
        - Phase-2: Core(Host), IPsec(End-Node)
      - 2.6.20 + radvd-1.0
        - Phase-2: Core(Router), IPsec(Security Gateway)

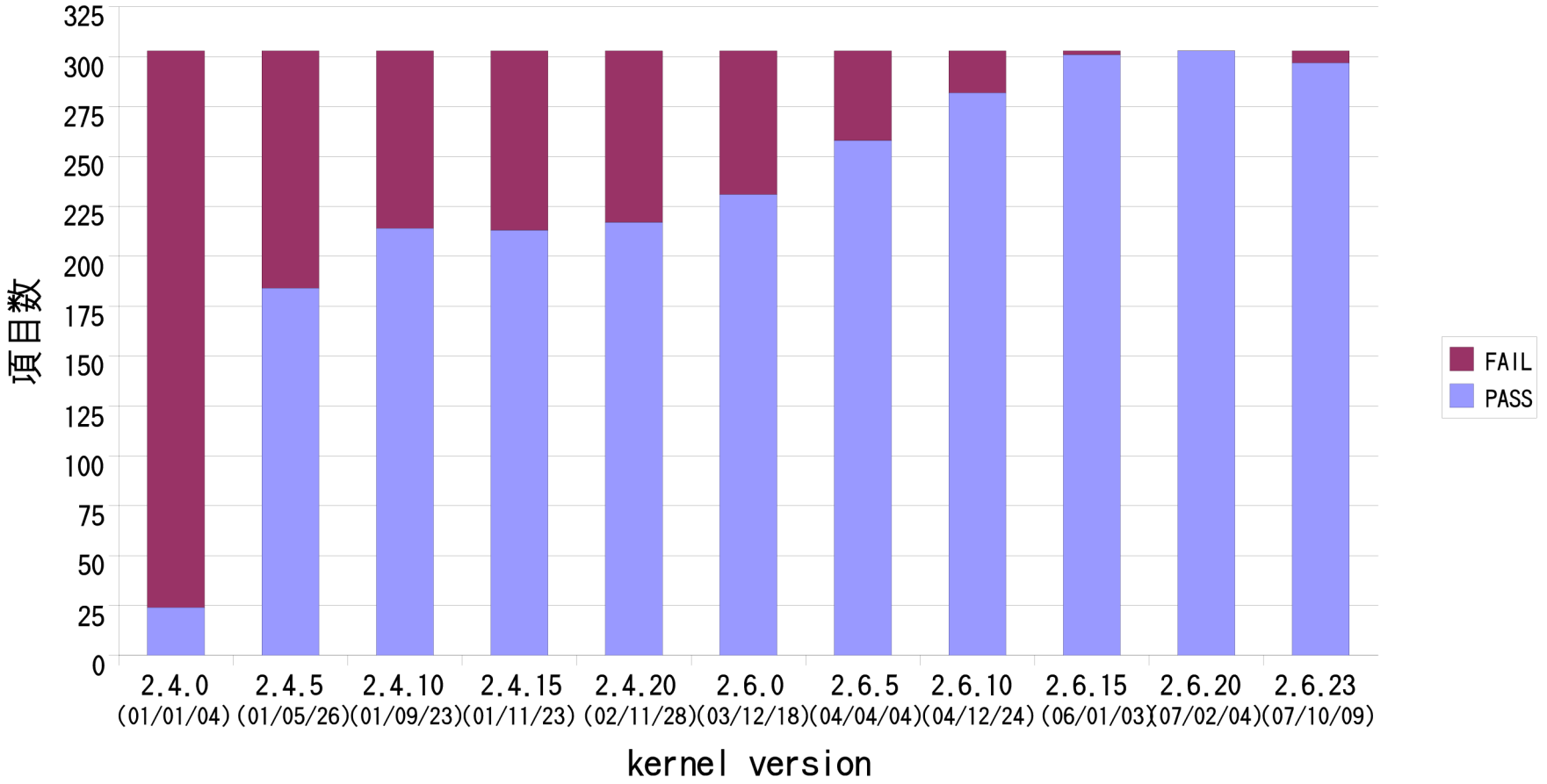
Quality of the Stack (cont'ed): Automatic Test System

- USAGI Testlab
  - http://testlab.linux-ipv6.org
- Daily automatic test system to find regressions
  - Basic tool: TAHI Conformance Test
    - IPv6 Core: Host, Router
    - IPsec: End-Node, Security-Gateway
    - Mobile IPv6: CN, HA, MN
  - Target: every *-git kernels

Quality of the Stack (cont'ed)
Host



Quality of the Stack (cont'ed)
Router



- Recent FAILs are due to RH0 deprecation.

Table of Contents

Missing Pieces and Others: IPv6 Multicast Forwarding

- Linux box as a multicast (PIM-SM SSM) router
  - Original work by Mickael Hoerdt
  - Imported into my tree and rebased
    - 2.6.24-dev
  - With several issues fixed
- Target: 2.6.26
- Further optimization is also planned.

Missing Pieces and Others

- rp_filter
  - Check for reverse-path routing
- “inet6peer” infrastructure
  - Randomized fragment-id
  - SYN Cookies
- HC (64bit) counters on 32bit systems
- Some of advanced API
- Several IPsec things

Missing Pieces and Others: in the Wild

- HIP (Host Identity Protocol)
  - HIPL: HIP for Linux (by HIIT (Helsinki Institute for Information Technology) and HUT (Helsinki University of Technology))
  - OpenHIP (or aka Boeing HIP)
- XCAST (Explicit Multicast) [RFC5058]
  - “2.0” is under development
  - Built in user space?

Future Directions

- To make IPv4 and IPv6 cleaner
  - IPv6 module still cannot be unloaded.
- To consolidate more things between IPv4 and IPv6
  - We have more things to share.
  - Split up IP-generic code from net/ipv4

Table of Contents

The Rabbit Is Finally...

- The USAGI Project consortium is planned to be concluded in March 2008.
  - Its mission has been achieved.
    - Enough for practical use
    - Getting more eyes
- We still continue development and deployment of IPv6.
  - We would shift our focus to more advanced topics.

Conclusion

- Linux IPv6 stack has a lot of good, practical features, plus more great things will come soon.
- USAGI Project will be concluded in this March, in order to shift our focus on more advanced research and development items. Maintenance and development will be continued.

Thank you