NetConf - IPv6 Developments

USAGI/WIDE Project / Keio University

Hideaki YOSHIFUJI
<yoshfuji@linux-ipv6.org>
July 14th, 2004
Table of Contents

- USAGI Project
  - Overview
  - Current Status
- Planned Patches / Features
- Current Development Details
- Future Plans
USAGI Project

Universal Playground for IPv6 / Rabbit

- Since fall, 2000
- Sponsored by WIDE Project
- Core members from research institutes and companies
- Collaborating with KAME, TAHI and Nautilus
  - KAME (turtle) IPv6, IPsec,... for BSDs
  - TAHI: Verification technology
  - Nautilus: Network Mobility (NEMO)

Goal
- To provide high quality IPv6 stack based on Linux
USAGI Project’s Target Areas

Target Areas (Past - Present)
- IPv6 API
- IPv6 core protocols
- IPsec
- Routing
- Packet filtering (Netfilter)
- Mobile IP
Current Status (1)

- IPv6 API
  - New RFCs available
    - Basic API (RFC3493 aka RFC2553bis)
      - probably done
    - Advanced API (RFC3542 aka RFC2292bis)
      - not yet
Current Status (2)

- IPv6 Core Protocols
  - USAGI Linux 2.6 (snapshot on Jan 19, 2004) got "IPv6 Ready Logo" (tm) from IPv6 forum
    - http://www.ipv6ready.org
  - We see no grave issues
Current Status (3)

- **IPsec**
  - basically done
  - Random fixes and improvements
  - Racoon v2
  - new specs are coming
    - ESPv2, IKEv2

- **Routing**
  - Router Selection / Load Sharing
    - halfly done
  - Policy Routing
    - done (by ville) but check required
  - Multicast Routing
    - not yet
Current Status (4)

- Packet filtering (nf_conntrack)
  - Protocol independent netfilter infrastructure
  - basically done
  - do we need NAT? :-)

- Mobile IPv6
  - under development
  - will be available in this fall
    - before ESTI in October
  - Nautilus Project (another project of WIDE Project) are going to develop NEMO on Linux
Planned Changes

IPv6 Core Protocol
- NDP
  - Accuracy of timing
    - run timer in NUD_REACHABLE
    - eliminate neigh_sync()
  - Neighbor state transition does not conform to the spec.
- Fragmentation / MTU
  - amount of "fragment header" (8 bytes) are always eaten in case fragmentation is required.

It’s time to remove "EXPERIMENTAL" and say Y!
Planning Changes (cont.)

Routing
- **Router Selection / Load Sharing**
  - select preferred route from routes of same metric
- **Policy Routing**
  - rule table
  - default source address selection
  - source address determination when looking up route
Fragmentation / MTU Issue

"Fragment header" (8 bytes) is always reserved.
Fragmentation / MTU Fix

When the packet size are reaching MTU, move tail of current fragment to new one

MTU

reserved

IPv6 payload

IPv6 payload

mtu

IPv6

payload

IPv6

payload

IPv6

payload

IPv6

payload

IPv6

payload

IPv6

payload

IPv6

payload

IPv6

payload

ip6_append_data()
Router Selection

**Issue**
- select one route from multiple routes of same metric
- rt6_dflt_pointer is too static and only for default routes

**Solution**
- round-robin routes of same metric
- use "highest" preferred route
Remaining Issue

- standard specifies hash-based selection
  - how to select an entry in the list?
- Probably we always need to create host route for stable route
Policy Routing

Discussed with HUT GO/Core Project

- search first rule what the request conforms to.
- (*) if rule not found, route not found. (end)
- lookup route in the table which is specified by the rule
- if returned route conforms to the rule, use it. (end)
- otherwise, search next rule what the request conforms to. (repeat from *)
IPsec

- Add icmp type/code to selector
- Fix AH calculation w/ routing header
- Reply window seems strange
- Parse flow when sending messages via raw socket
## Current / Future Items

- **Mobile IPv6**
- **Multicasting**
  - Copy "ipv4/ipmr.c" is not good, I think.
- **Advanced API**
  - new API overrides the definition...
    > probably we allocate new sockopt and provide old sockopt for compatibility
- **Introduce u64 counters**
  - update unsigned long internally, and update u64 periodically
- **everything-over-ipv{4,6} tunnel**
  - ipv{4,6} over ipv4 (tunl), replaces sitX (and greX?)
  - ipv{4,6} over ipv6 (ip6tnl)
Current / Future Items (random)

- Introducing expiration list for purging entries
  - sorted by expiration time
  - e.g. routing

- Introduce "long term" timer
  - timer in HZ precision often overflows

- Restructuring ip directory
Mobile IP

- Mobile IPv6 is now RFC
  - RFC3775 "Mobility Support in IPv6"
  - RFC3776 "Using IPsec to Protect Mobile IPv6 Signaling Between Mobile Nodes and Home Agents"

- Packet Delivery Framework
  - Bidirectional Tunneling
  - Route Optimization
Mobile IP (cont.)

- MH (Mobility Header)
  - signaling
  - extension header but nexthdr = NONE
- HoA option (in (special) destination header)
  - for source HoA; source is MN
- Routing header option of type 2
  - for destination HoA; destination is MN
Basic Design

Designed by

- USAGI and HUT (Helsinki Univ. of Tech.)

Packet modifications, such as Bi-Tunnel, RO and IPsec, are done inside kernel

- XFRM framework
  - Build XFRM state respectively
    - it manages packet mangling.
    - like Binding Cache, but it is not the same.

- Standard IPv6-IPv6 tunnel
  - for link-local protocol

Signaling is handled in userspace daemon

- manages binding cache and XFRM policy/state
<table>
<thead>
<tr>
<th>Kernel User API</th>
</tr>
</thead>
<tbody>
<tr>
<td>□ XFRM</td>
</tr>
<tr>
<td>○ ~1500 lines</td>
</tr>
<tr>
<td>□ PF_MOBILITY(?)</td>
</tr>
<tr>
<td>○ under discussion w/KAME</td>
</tr>
</tbody>
</table>
XFRM State Management API

- current keys: (family, daddr, spi, proto)
- not sufficient (especially for mobile ip)
  - userland daemon need to add/delete with specific source address

```c
struct xfrm_usersa_id {
    xfrm_address_t daddr;
    __u32 spi;
    __u16 family;
    __u8 proto;
    xfrm_address_t saddr; // NEW
};
```

- This is probably good for xfrm6_tunnel management, too.
  - We see "hashed" spi for xfrm6_tunnel.
- Mobile IP is a kind of tunnel, anyway.
Binding Error Notification

- Binding Error will be passed to the userspace using new XFRM_MSG_MIP6NOTIFY message
  - Unexpected set of CoA and HoA
  - unknown MH type
    - this can be handled in userspace, directly
Remaining issue

- HA shall not accept Home Registration without IPsec while HA (Home Agent) may receive BU from MN as if HA is CN, which is valid, without IPsec
  - XFRM Selector extension, which allow us to use H bit in BU as a selector
    - pros: easy to implement
    - cons: MH "flag" is very local to BU (is a type of MH); a kind of layer violation
  - Refer sec_path[] at in-kernel MH receiver
    - pros: easy to implement
    - cons: still needs in-kernel MH receiver
If we had IPsec information (such as protocol and algorithm) notification mechanism, we could do everything in userspace

- **Pros:** simpler and generic features in kernel
- **Cons:** no such standards
Remaining issue (cont.)

- IPsec and Mobile IP co-existence
  - How to allow coexistence of IPsec and Mobile IP for same destination?
    - Combining IPsec / Mobile IP Policies
    - Allow multiple type of templates
    - Merge them according to "meta-template"
    - still under discussion
Restructuring ip directory

- net/ip/ipv4
  - tcp.c, tcp_diag.c, tcp_input.c, tcp_minisocks.c, tcp_output.c,
    tcp_timer.c + sctp_ipv4.c

- net/ip/ipv6
  + sctp_ipv6.c

- net/ip/tunnel

- net/ip/tcp
  + tcp.c, tcp_diag.c, tcp_input.c, tcp_minisocks.c, tcp_output.c,
    tcp_timer.c

- net/ip/sctp
  - ipv6.c
Request to Other Maintainers

Please, please keep IPv6 in your mind.

- Expect extension headers
- Please do not make things depend on seeing inner "things" (including headers)

When you define API and/or see API, keep the viewpoint of "protocol independency"

- use protocol independent address structures
  - pointer to sockaddr{}
  - sockaddr_storage{}
- sockaddr_in{}? hmm...
- u32? in_addr? What is it? :-)
