Notes on IPv6 Initialization

Hiroki Sato (hrs@FreeBSD.org)

TOKYO TECH
Pursuing Excellence

Targets/Goals At a Glance

- Enable IPv6 by default at userland level.
- Flexible framework that can support host/router model and more.
- No regression for non-IPv6 people.
- No extra burden for IPv6 people, compared to the original KAME implementation.

History(1)

- KAME implemented a script “rc.network6” based on the IPv4 counterpart 10 years ago.
- It was based on “ipv6_”-prefixed variables to separate the namespace. FreeBSD-specific.
- rc.network6 was renamed with rc.d/network_ipv6 when rc.d script was committed.

History(2)

- rc.d scripts for IPv4 was rewritten by mtm@ in 2003. The rc.network was removed and rc.d/netif was added.

- Almost at the same time, devd(8) was added by imp@. After that, thompsa@ committed IFNET devctl notification in 2006.

- The default devd.conf invokes rc.d/netif on arrival/depature of NICs.

History(3)

- rc.d/network_ipv6 was integrated into rc.d/netif by hrs@ in 2009.
  - The "ipv6_" namespace was also integrated.
    - IPv6 was always enabled.
    - IPv6 configuration was also done by devd(8).
- Some default behaviors of IPv6 were changed by dougb@ in 2010.

KAME rc.network6

- ipv6_enable=NO by default while GENERIC contains INET6.
- This heavily depends on host/router model. If ipv6_enable=YES:
  - All of IFs have an LL addr (automatically-assigned).
  - If ip6_forwarding=0, do SLAAC for "one of NICs with no manual configuration" by setting a sysctl accept_rtadv=1 and using rtsol(8).
  - If ip6_forwarding=1, add subnet-router anycast addr.
- ip6addrctl also depends on $ipv6_enable.

KAME rc.network6 vs Goals

- ~~Enable IPv6 by default at userland level.~~
- ~~Flexible framework that can support host/router model and more.~~
- No regression for non-IPv6 people.
- No extra burden for IPv6 people, compared to the original KAME implementation.

Drawbacks of network_ipv6

- Difficult to reinitialize IFs after a boot. Per-IF init/reinit is impossible. (For selection of SLAAC NIC, it enumerates all of interfaces)
- No compatibility with dynamically-added/removed interfaces due to it.
- Auto LLA conf and accepting RA are controlled by global knobs.
  - Multiple RAs from multiple IFs break the FIB.
  - A router cannot receive RAs.

Drawbacks of network_ipv6

- People tends to configure "no LLA + IPv6 GUA" manually when ipv6_enable=NO (default).
- A kitchen sink of IPv6 functionality; IPv6 network options, configuration of stf(4) and faith(4), initialization of routing table, etc... => impossible to enable them by default

rc.d/netif integration

- To enable IPv6 by default:
  - Always do IPv6 global initialization which does not affect IPv4 people.
  - Separate initialization per-IF basis.
  - Convert host/router model to per-IF, too.

rc.d/netif init procedure

- 1. IPv6 global initialization at boot time.
- 2. IPv6 per-IF initialization.
  - dynamically-added/removed interfaces will be handled by 2 on demand.
  - idempotent initialization; multiple passes are possible because of asynchronous initialization by devd(8).

IPv6 global initialization

- Things enough to be done once.
- Add “unconditionally-block” routes to FIB.
- Set IPv6 netoptions like IPV6_IPV6ONLY.
- Set src addr selection policy. IPv4-pref or IPv6.
- RA handling was global one, but moved to per-IF because we cannot handle the host/router model globally.

IPv6 per-IF initialization

- Add IPv6 addresses (LLA, GUA) and per-IF flags.
- host/router model is factored into an interface classification "RA-receiving or not":
  - A host = RA-receiving IF. This means !RA-sending at the same time. Not depending on routing capability.
  - A router = !RA-receiving. It should be RA-sending, but KAME kernel does not have the capability (rtadvd(8) has it).
  - Important: Do not consider "host or router" classification in a system-wide basis. It never becomes consistent.

Implementation

- Add “auto_linklocal” and “accept_rtadv” per-IF options.
- Still covers old host/router model, and more: (IF1=host, IF2=router):
  ifconfig_IF1_ipv6=”inet6 accept_rtadv” (SLAAC only)
  ifconfig_IF1_ipv6=”inet6 2001:db8:1::1/64 accept_rtadv” (manual + SLAAC)
  ifconfig_IF2_ipv6=”inet6 2001:db8:2::1/64” (manual conf)
  ifconfig_IF2_ipv6_alias0=”inet6 2001:db8:2::/64 anycast” (subnet-router anyc)
- Remember: KAME’s accept_rtadv is global, and RS msg was sent one of IFs which we cannot control.
- This classification based on “RA-receiving or not” makes things much simpler and extensive.

Default settings(1)

- System-wide defaults can be controlled by something like (some are not implemented yet):
  - ifconfig_DEFAULT_ipv6=”inet6 accept_rtadv”
  - net.inet6.ip6.accept_rtadv=1
    in /etc/sysctl.conf
  - net.inet6.ip6.auto_linklocal=0
    in /etc/sysctl.conf

Default settings(2): RA

- "RA-receiving by default" is too dangerous for the system default while IPv6 enthusiasts say it is an IPv6-way.
- configuration via RA does not work with multiple NICs (including dynamically-added pseudo IFs). Mainly because of FIB manipulation.
- We cannot go back to network_ipv6 (pick up one IF to send RS).
- L3 auto address configuration must be disabled by default. This is our secteam's decision, anyway.

RA handling scenarios

- Simple SLAAC:
  Mark an IF as "RA-receiving". Should work even if ip6_forwarding=1 in the future.

- O-flag, RDNSS options:
  Mark as RA-receiving, and then the kernel will pass information to a userland daemon. DHCPv6, /etc/resolv.conf (not implemented yet).

- "RA-receiving or not" does not always imply a simple behavior. Magic keywords like "RTADV" does not make sense (DHCPv6 can actually be triggered by RA).

Default settings(3): LLA

- LLA is essential for IPv6. However, KAME does not force it.
- LLA is annoying for IPv4 people who are not familiar with IPv6. Even if it is not harmful to IPv4 functionality itself.
- Default settings were:
  - 1) No LLA other than lo0.
  - 2) Assign an LLA only if the rc.conf has $ifconfig_IF_ipv6 line.
  - 3) Add a seatbelt for manual IPv6 configuration.

Default settings(3): LLA

- LLA is usually assigned when an IF becomes UP (auto_linklocal per-IF flag)
- If no LLA is assigned, IPv4 people is likely to create “GUA + no LLA” situation when they try a manual configuration of IPv6.
- “GUA + no LLA” works in some degree, but most of functionality

Default settings(3): LLA

- So, the actual implementation was:
  - a) Set ifdisabled flag to all interfaces. This prevents automatic LLA assignment.
  - b) Clear it if the interface has ifconfig_IF_ipv6 line.

Default settings(3): LLA

- Why not simply remove “auto_linklocal”?
  - removing “auto_linklocal” also prevents automatic LLA assignment.
- However, after initialization at boot time, people can configure IPv6 GUA by ifconfig (8).
- It leads to “no LLA + an IPv6 GUA” situation easily. I would like to prevent this.

Default settings(3): LLA

- A) Using “ifdisabled” (+ifdisabled by default)
  - ifconfig fxp0 inet6 2001:db8:1::1/64
    => assign a GUA but communication blocked.
    ifconfig fxp0 inet6 -ifdisabled
    => assign an LLA.
- B) Using “auto_linklocal” (-auto_linklocal by default)
  - ifconfig fxp0 inet6 2001:db8:1::1/64
    => assign a GUA only and communication works.

Issues of "ifdisabled" flag

- Must not considered as a "disable IPv6" flag. Because:
  - This simulates DOWN at ND level. L3 handling in the kernel still works.
  - Originally designed for DAD failure. So, clearing it lightly is always an wrong idea.
  - For example, If automatic LLA assignment happens on all IFs even if "ifdisabled" is on them, routes injected into the FIB lead strange behaviors:
    - scope violation
    - communication failure due to invalid routes

Issues of "ifdisabled" flag

- Manual manipulation must be avoided wherever possible if you use IPv6.
  - Must not force IPv6 people to use it.
  - Only used as a seatbelt for IPv4 people (because I designed in that way).
- If we strongly need an enable/disable flag for IPv6, we should implement it in another way.

Impact of default settings for IPv4 people?

- No extra burden for IPv4 people even if IPv6 initialization happened.
  - All of IFs other than lo0 have no LLA.
  - IPv4-preferred src addr selection.
  - ifdisabled on all non-IPv6 IFs to prevent malformed manual IPv6 configuration.
  - Important:
    When people do a manual configuration of IPv6, it must be done in a correct way. To realize that, one extra step “clearing ifdisabled flag” was added.

Planning changes by hrs@

- More fine-grained manipulation of RA. Ignoring defroutes is necessary at least.
- Make more global flags to per-IF ones. tempaddr may be a good target.
- Performance analysis of INET6 stack as we did for INET.
- A unified userland daemon supporting local-link handling like RA and SeND.

Contention happened in Apr/May 2010

- The implementation in last year has been partially reverted.
- Primary issues of the current implementation:
  - a) Abuse of "ifdisabled". IPv6 people who manually configure IFs always need to clear it.
  - b) RTADV keyword in rc.d.
  - c) IPv6 src addr selection by default.
- Other parts are harmless (some are unreasonable changes, though).

a) Abuse of "ifdisabled"

- Technically unreasonable. Must be changed.
- The difference between the previous one and the current one is ifdisabled for non-rc.d configured IFs.

a) Abuse of "ifdisabled"

- Possible workaround:
  - A) Replace it with "-auto_linklocal by default".
    It is safer, and basically the same effect other than seatbelt for IPv4 people. ("no LLA + a GUA" issue is left in that case)
  - B) Or, add net.inet6.ip6_ifdisabled and set to 1 by default. IPv6 people can set it to 0.

a) Abuse of “ifdisabled”

- After discussion: Back out the ifdisabled handling to the previous behavior.
  - “ifdisabled by default on non-IPv6 IFs” is only for IPv4 people. The flag should be cleared for IPv6 people.
- We need a knob to control whether the system is ready for IPv4-only people or IPv6 people.
- The default of the knob must be “IPv4-only”. IPv6 people can choose “I use IPv6” by using the knob.
- The name of the knob is a moot point, but at this moment $ipv6_prefer is enough and works fine. Decoupling “ifdisabled or not” and “IPv6-preferred src addr sel” from it is sensitive and must not be happened in Doug’s way.

b) RTADV keyword

- A kind of style issue.
- IMO it is not reasonable because what RA implies is complex and we already have "accept_rtadv".
- I can ignore it. No one suffers from it.

b) RTADV keyword

- After discussion: Back out this if possible. Even if it does not happen, it is possible that we need to remove it because the meaning of "receiving RA" becomes more complex recently (it can mean both SLAAC and DHCPv6, for example).

- Non-sense to have RTADV. We already have "accept_rtadv". If we need a shorter name, change ifconfig(8), not rc.d scripts.

- We cannot agree with it, but can ignore it if Doug insists this strongly. Impact is small. No one badly suffers from it.

c) IPv6 src addr selection

- Needs a consensus.
- Can we agree with possible performance regression? If so, I don't mind changing the default. No discussion/agreement so far.
- I don't want to hear "IPv6 is the culprit, remove options INET6 and recompile" after a release.

c) IPv6 src addr selection

- After discussion: Must be backed out. IPv4-preferred should be the default at this moment.

- We need more careful investigation of the performance regression before making IPv6-preferred the default.

- No justification here if we have an expected impact to IPv4 people.