NFD - Feature #4528

Management thread

03/02/2018 02:43 PM - Junxiao Shi

Status:	In Progress	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Davide Pesavento	**% Done:**	0%
Category:	Management	**Estimated time:**	12.00 hours
Target version:			

Description

NFD currently implements its management protocol in the same thread as packet forwarding. As part of the management protocol implementation, the main thread is doing computationally intensive work, such as packet signing and signature verification, causing significant forwarding delays.

This issue moves NFD management to a separate thread, using the following approach:

- **Data structure locking**: Add a lightweight lock to NFD's data structures that are accessed by both forwarding and management. This includes FaceTable, FIB, StrategyChoice, and ContentStore.
- **Management thread separation**: Create a management thread. Connect management thread to forwarding through a Unix stream face.

Related issues:

Related to NFD - Feature #4530: More efficient communication between threads	**In Progress**
Blocks NFD - Feature #4529: Merge NFD-RIB into management thread	**New**
Blocked by NFD - Task #3565: Investigate management thread	**Closed**

History

#1 - 03/02/2018 02:45 PM - Junxiao Shi

- Blocks Feature #4529: Merge NFD-RIB into management thread added

#2 - 03/02/2018 02:47 PM - Junxiao Shi

- Related to Feature #4530: More efficient communication between threads added

#3 - 03/02/2018 02:53 PM - Junxiao Shi

- Blocked by Task #3565: Investigate management thread added

#4 - 03/17/2018 06:26 PM - Davide Pesavento

- Status changed from New to In Progress

Junxiao Shi wrote:

> Connect management thread to forwarding through a Unix stream face.

I will not do this. It's faster to simply implement #4530 directly.

#5 - 03/18/2018 02:58 AM - Junxiao Shi

> > Connect management thread to forwarding through a Unix stream face.
>
> I will not do this. It's faster to simply implement #4530 directly.

That's fine. I wrote "Unix stream face" to reduce blocking relations at the cost of increasing total work amount.

#6 - 03/18/2018 04:27 PM - Davide Pesavento

With this change (and even more with #4529) the current source-level split into the three core, daemon, and rib subdirs doesn't make sense anymore.

daemon/mgmt and rib should be together, and they should be separate from the rest of the modules that run on the main (forwarding) thread. Feel free to bikeshed how exactly this should look like.

Management unit tests should also be split from unit-tests-daemon. In fact, I propose to go one step further and split unit-tests-daemon into unit-tests-face, unit-tests-fw, unit-tests-table, and unit-tests-mgmt. After #4529, unit-tests-rib can either be merged into unit-tests-mgmt or be kept separate.

#7 - 02/20/2019 11:48 AM - Junxiao Shi

Regarding codebase structure, I would say:

- dp/face: data plane, face system
- dp/table: data plane, tables
- dp/fw: data plane, forwarding
- cp/mgmt: control plane, management protocol
- cp/rib: control plane, RIB service

#8 - 02/20/2019 12:02 PM - Davide Pesavento

There is an implicit desire to minimize the total number of changes, especially unnecessary ones. The current daemon/{face,fw,table} subdir structure is perfectly fine, there is no reason to change it, so I will not touch it. I'm only going to change rib/, daemon/mgmt (if necessary), and probably some files in core/. So your proposal does not work for me.

Also, what should happen to RibManager? It's currently in rib/ but it's technically part of the management protocol.

#9 - 02/20/2019 01:20 PM - Junxiao Shi

> There is an implicit desire to minimize the total number of changes, especially unnecessary ones.

Minimizing diff size is explicitly a non-goal. See https://gerrit.named-data.net/#/c/NFD/+/332/1/daemon/fw/strategy.cpp@39

> The current daemon/{face,fw,table} subdir structure is perfectly fine, there is no reason to change it, so I will not touch it.

Either way is fine. daemon means "data plane". Just put that in daemon/README.
Similarly, you may rename cp to something better, but I'd suggest avoiding rib or mgmt, because control plane does not equal RIB or management.

> Also, what should happen to RibManager? It's currently in rib/ but it's technically part of the management protocol.

It eventually should be cp/mgmt/rib-manager.hpp.
This would require some object ownership changes in #4731.

#10 - 02/21/2019 08:40 PM - Junxiao Shi

> I don't understand what the problem is. You said "daemon/" is fine. Be more specific in your comments.

daemon is fine as an alternative to dp.
mgmt and rib belong to control plane and they need a separate folder other than daemon.

#11 - 02/21/2019 08:48 PM - Davide Pesavento

Not gonna do that. "daemon" means daemon, not dataplane. All code of nfd binary goes in daemon/ subdir. As I already said, I will not change that.

#12 - 02/22/2019 03:36 AM - Junxiao Shi

Due to this dispute, this issue needs to be discussed in next NFD call.

#13 - 02/23/2019 01:00 PM - Alex Afanasyev

I completely agree with Davide. Whatever control not control plane is the business of the daemon and not source code organization. The structure is already non-trivial to understand, I see absolutely no reason to make more changes than minimally necessary.

#14 - 03/17/2019 05:50 PM - Junxiao Shi

In https://gerrit.named-data.net/5328 several components in the RIB thread start to use getScheduler() global function instead of the m_scheduler member variable. I remember that 6th-ndn-hackathon report indicates that globals should be avoid in favor of passing io_service and schedulers around. What has changed since then, that leads to a different design?

#15 - 03/17/2019 10:44 PM - Davide Pesavento

I may have found a way to do it without passing io_service and Scheduler around. Of course getting rid of all "globals" is still possible, but I'd rather not do it unless strictly necessary, I tried with Scheduler and you need to change really *a lot* of code just to pass the reference around to the classes that need it.

#16 - 03/25/2019 10:36 AM - Davide Pesavento

And more importantly, self-learning required adding the getMainIoService and getRibIoService helpers, which are another way to do this. We did investigate this approach during that hackathon, but ultimately abandoned it because we thought it was not particularly elegant and expected it to be rejected by reviewers.

#17 - 08/27/2019 08:43 PM - Davide Pesavento

- Target version deleted (v0.7)

#18 - 09/12/2020 02:25 PM - Junxiao Shi

> how much performance improvement would you expect from separating mgmt out -- 20%? 50%?

Benefits depend on frequency of management commands.
It will most likely be less than 20%.