NFD - Bug #2492

Packet processing delays in NFD

02/08/2015 11:08 AM - Jeff Burke

Status:	Closed	**Start date:**	
Priority:	High	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Core	**Estimated time:**	0.00 hour
Target version:	v0.5		

Description

ndnrtc application development has identified repeatable packet processing delays in NFD of 250-500ms occurring nominally every 2-3 minutes, for reasonable CPU load, bandwidths in the 150-1000 kbps range, and in the presence of no other network traffic on a switched gigabit LAN. These stop all packet flow and cause audio/video interruptions that are unacceptable for end-user QoE.

More details can be found in recent mailing list threads and conversations with Davide. (I will leave it to others to post them here.)

Debugging the current installation where this problem is occurring is requested.

History

#1 - 02/08/2015 12:11 PM - Davide Pesavento

- Start date deleted (02/08/2015)

Comparing the pcap and NFD logs, Peter and I discovered that the "delayed" packets are actually received by the kernel without any delays. However NFD does not read the packets from the socket until much later (~200-400ms later). We conjectured that during that time NFD was busy doing something else and didn't re-enter the io_service event loop, therefore it was not notified of pending packets. (@Peter, if you still have a pic of the timing diagram you drew, please attach it here)

Unfortunately all logging except for UdpFace and Forwarder was turned off, thus we don't know what was going on exactly. Alex suspects some data signing operations caused by the notification stream or other management operations. It was suggested to use the "file" TPM backend in order to prove or disprove this theory. It would also be beneficial to set the default log level to INFO (at least), unless it causes performance problems.

Another possibility is that the problem might be in the kernel or in Boost.Asio, but this seems less likely.

#2 - 02/08/2015 12:17 PM - Jeff Burke

We will attempt to test the TPM setting change prior to the 1P NFD conference call on Monday.

#3 - 02/08/2015 12:55 PM - Junxiao Shi

According to #1589 note-1, each signing operation using OSX KeyChain takes less than 20ms.

A 250ms delay, if caused by OSX KeyChain signing, means signing more than 12 packets.
I can't think of why NFD Management would need to sign 12 packets in a row.

Can we add a logger call just before io_service::run call, so that we can know whether NFD entered the IO loop?

#4 - 02/08/2015 01:19 PM - Davide Pesavento

Junxiao Shi wrote:

> According to #1589 note-1, each signing operation using OSX KeyChain takes less than 20ms.
>
> A 250ms delay, if caused by OSX KeyChain signing, means signing more than 12 packets.
> I can't think of why NFD Management would need to sign 12 packets in a row.

I know, but it's the only semi-plausible idea we had so far. We thought about the CS too, but that also doesn't really make sense.

> Can we add a logger call just before io_service::run call, so that we can know whether NFD entered the IO loop?

Not sure what you mean here... The event loop is re-entered whenever a callback finishes its execution, so there isn't a single entry point.

#5 - 02/08/2015 01:30 PM - Junxiao Shi

> > According to #1589 note-1, each signing operation using OSX KeyChain takes less than 20ms.
> >
> > A 250ms delay, if caused by OSX KeyChain signing, means signing more than 12 packets.
> > I can't think of why NFD Management would need to sign 12 packets in a row.
>
> I know, but it's the only semi-plausible idea we had so far. We thought about the CS too, but that also doesn't really make sense.

After NFD Management signs some packet, it will send the packet into forwarding pipelines.

With 'Forwarder DEBUG' log level, the Name of that packet will appear in the log.

Do you see any management generated packet Name (starts with /localhost) near the log line of problematic NdnCon packets?

> > Can we add a logger call just before io_service::run call, so that we can know whether NFD entered the IO loop?
>
> Not sure what you mean here... The event loop is re-entered whenever a callback finishes its execution, so there isn't a single entry point.

I was thinking about tracing the io.run() call in main function, but now I know it won't work.

#6 - 02/09/2015 11:01 AM - Peter Gusev

- File IMG_5708.JPG.jpeg added

Here is the diagram we draw with Davide. Also, log level was set to NONE, except UdpFace (TRACE) and Forwarder (DEBUG).
Can anyone point me to the documentation on how to set a file tpm in nfd.conf? Can't find it in the conf file.

#7 - 02/09/2015 11:07 AM - Peter Gusev

Peter Gusev wrote:

> Here is the diagram we draw with Davide. Also, log level was set to NONE, except UdpFace (TRACE) and Forwarder (DEBUG).
> Can anyone point me to the documentation on how to set a file tpm in nfd.conf? Can't find it in the conf file.

Ok, figured this out (/usr/local/etc/ndn/client.conf).

#8 - 02/09/2015 11:12 AM - Peter Gusev

Getting "ERROR: private key doesn't exist" after trying to register a prefix using nfdc (nfdc register /ndn/edu/ucla/remap udp://131.179.141.33).

#9 - 02/09/2015 11:35 AM - Jeff Burke

How are you installing NDN (source or macports)?

I don't know where the PIB (ndnsec-public-info.db) is in your installation... in mine it is in .ndn and I think needs to be deleted, and then re-run the default certificate generation steps.

Can the nfd folks advise?

#10 - 02/09/2015 11:39 AM - Peter Gusev

NFD was installed from sources.

#11 - 02/09/2015 11:43 AM - Peter Gusev

Ok, figured out.
Simply had to re-run these steps:

```
; 1. Generate and install a self-signed identity certificate:
;
;      ndnsec-keygen /whoami | ndnsec-install-cert -
;
; Note that the argument to ndnsec-key will be the identity name of the
; new key (in this case, /your-username). Identities are hierarchical NDN
; names and may have multiple components (e.g. /ndn/ucla/edu/alice).
; You may create additional keys and identities as you see fit.
;
; 2. Dump the NDN certificate to a file:
;
;      sudo mkdir -p /usr/local/etc/ndn/keys/
;      ndnsec-cert-dump -i /whoami >  default.ndncert
;      sudo mv default.ndncert /usr/local/etc/ndn/keys/default.ndncert
```

#12 - 02/09/2015 11:44 AM - Alex Afanasyev

If from sources, then ndnsec-public-info.db is in your home directory ~/.ndn.   For brew, nrd and nfd have their own homes, which are in /usr/local/var/lib/ndn/*. Similar to macports, but different path: /opt/local/var/lib/ndn/*. In both macports and brew, nfd and nrd already use file-based tpm.

#13 - 02/09/2015 12:48 PM - Peter Gusev

Got tested with file TPM.

Still getting similar results.

Tested same bitrate over ~10min period. Got 4 cases of intermediate NFD delays.

Full logs/dumps can be found here.

#14 - 02/09/2015 01:48 PM - Davide Pesavento

Can you set the default log level to DEBUG (or INFO) or does it become too slow? It's really hard to figure out what was going on without the rest of the log output.

#15 - 02/09/2015 02:29 PM - Junxiao Shi

- Target version changed from v0.2 to v0.3

20150209 conference call decides that @Peter should do the following two experiments:

- upgrade the router node to NFD git-HEAD version and keep using Macbook, and determine whether the bug exists
    - keep existing consumer and producer nodes, since @Davide has determined that the problem isn't with them
- replace the router node with a linux machine running NFD git-HEAD version, and determine whether the bug exists
    - keep existing consumer and producer nodes, since @Davide has determined that the problem isn't with them

If either experiment turns out to have the bug, we'll try to reproduce the problem in our environment.

If both experiments do not have the bug, and you are still curious why it was happening, please use this procedure to find out which commit fixed the bug, and we'll try to explain.

#16 - 02/09/2015 02:32 PM - Jeff Burke

We can do option one (NFD v0.3 on MacOS X).

Can you provide a Linux box for option two that we can temporarily use at the hub?

Again, we strongly request your help debugging this *in situ*, not reproducing this in your environments.

#17 - 02/09/2015 02:33 PM - Jeff Burke

Just so it's clear, we're not "curious" about the bug source. This is a *block* on deployment of ndnrtc and we are asking for help to determine the root cause, not solving the symptom.

#18 - 02/09/2015 02:37 PM - Junxiao Shi

> Can you provide a Linux box for option two that we can temporarily use at the hub?

Results from both experiments are needed.

You may use the testbed gateway router. Not being in the same subnet shouldn't affect the result.

> Again, we strongly request your help debugging this *in situ*, not reproducing this in your environments.

If the problem isn't reproducible in any other machine or with any other application, it's not a NFD bug.

#19 - 02/09/2015 02:44 PM - Jeff Burke

Just because it is not reproducible in your environment does not disqualify the bug. (This is the whole point of doing deployments.)

We are testing some different log settings first, then will do NFD v0.3 on MacOS X.   Testing on a testbed hub is not equivalent (changing too many variables) - we will come up with a Linux box.

#20 - 02/09/2015 03:12 PM - Alex Afanasyev

> Just because it is not reproducible in your environment does not disqualify the bug. (This is the whole point of doing deployments.)

Even though it does not disqualify the bug, it points out a possibility that the cause for the bug is external to NFD. I'm not saying it is, just explaining why it is important to have a reproducible environment to dig out the error.

#21 - 02/09/2015 03:58 PM - Davide Pesavento

Alex Afanasyev wrote:

> Even though it does not disqualify the bug, it points out a possibility that the cause for the bug is external to NFD. I'm not saying it is, just explaining why it is important to have a reproducible environment to dig out the error.

The point is, Peter already has an environment where the phenomenon is consistently reproducible. What else do you want?

#22 - 02/09/2015 04:35 PM - Peter Gusev

I've run two more tests on the existing setup before updating intermediate hub to NFD 0.3.0.

This time, I run 1st test with default_level NONE and second test with default_level INFO. The problems seems to *almost* go away - only in the 1st run I have a pattern similar to the problem:

NFD1:

1423522455.985196   DEBUG: [Forwarder] onIncomingInterest face=268
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7072/data/%00%00
1423522455.985518 DEBUG: [Forwarder] onOutgoingInterest face=263
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7072/data/%00%00
^ 0 sec, 522.150 msec
1423522456.507346 DEBUG: [Forwarder] onIncomingInterest face=268
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7072/data/%00%00
^ 0 sec, 14.508 msec
1423522456.521854 DEBUG: [Forwarder] onOutgoingInterest face=263
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7072/data/%00%00
^ 2 sec, 2482.376 msec
1423522459.004230 DEBUG: [Forwarder] onIncomingData face=263
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7072/data/%00%00/2/7316/243/0
1423522459.004545 DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7072/data/%00%00
1423522459.004762 DEBUG: [Forwarder] onOutgoingData face=268
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7072/data/%00%00/2/7316/243/0
^ 0 sec, 65.617 msec
1423522459.070379 DEBUG: [Forwarder] onIncomingData face=263
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7072/data/%00%00/2/7316/243/0
1423522459.070666 DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7072/data/%00%00

NFD2:

no log

NFD3:

1423522472.777791   DEBUG: [Forwarder] onIncomingInterest face=266
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7072/data/%00%00
^ 0 sec, 118.009 msec
1423522472.895800 DEBUG: [Forwarder] onIncomingData face=269
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7072/data/%00%00/2/7316/243/0
1423522472.896227 DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7072/data/%00%00
1423522472.896580 DEBUG: [Forwarder] onOutgoingData face=266
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7072/data/%00%00/2/7316/243/0

Just for comparison, same logs for the previous frame:

NFD1:

1423522455.940876 DEBUG: [Forwarder] onIncomingInterest face=268
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7071/data/%00%00
1423522455.941184 DEBUG: [Forwarder] onOutgoingInterest face=263
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7071/data/%00%00
^ 287 msec
1423522456.229018 DEBUG: [Forwarder] onIncomingData face=263
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7071/data/%00%00/3/7315/243/0
1423522456.242072 DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7071/data/%00%00

```
1423522456.252751 DEBUG: [Forwarder] onOutgoingData face=268
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7071/data/%00%00/3/7315/243/0
^ 185 msec
1423522456.437698 DEBUG: [Forwarder] onIncomingInterest face=268
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7071/data/%00%00
1423522456.448964 DEBUG: [Forwarder] onOutgoingData face=268
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7071/data/%00%00/3/7315/243/0
```

NFD2:

no log

NFD3:

```
1423522472.733178 DEBUG: [Forwarder] onIncomingInterest face=266
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7071/data/%00%00
1423522472.733593 DEBUG: [Forwarder] onOutgoingInterest face=269
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7071/data/%00%00
^ 126 msec
1423522472.859767 DEBUG: [Forwarder] onIncomingData face=269
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7071/data/%00%00/3/7315/243/0
1423522472.860198 DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7071/data/%00%00
1423522472.860492 DEBUG: [Forwarder] onOutgoingData face=266
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7071/data/%00%00/3/7315/243/0
```

What do you think, guys? Should I try more "NONE log level" tests?
I'll update to NFd 0.3.0 now.

#23 - 02/09/2015 04:44 PM - Alex Afanasyev

Excessing logging could have contributed to significant processing overhead, as it involves disk operations (i'm assuming log output was written into file).

#24 - 02/09/2015 05:15 PM - Peter Gusev

Getting errors starting NFD:

```
1423530677.239185 FATAL: [NFD] PK_Signer: key too short for this signature scheme
1423530679.235611 FATAL: [NRD] PK_Signer: key too short for this signature scheme
```

The same error when trying $ ndnsec-keygen /whoami| ndnsec-install-cert -:

```
Error: PK_Signer: key too short for this signature scheme
```

Any suggestions?

#25 - 02/09/2015 05:31 PM - Alex Afanasyev

run just ndnsec-keygen /whoami. It probably throws some error which you don't see because of the piping.

#26 - 02/09/2015 05:33 PM - Peter Gusev

still the same:

```
$ ndnsec-keygen /whoami
Error: PK_Signer: key too short for this signature scheme
```

#27 - 02/09/2015 05:48 PM - Alex Afanasyev

I can only guess. Either some permission problem or there is a mixup of ndn-cxx versions...

#28 - 02/09/2015 05:49 PM - Peter Gusev

I used git-HEAD of ndn-cxx and git-HEAD of NFD, if it helps...

#29 - 02/09/2015 05:53 PM - Yingdi Yu

This error may happen when keychain fails to load the private key. In this particular case, I suspect that the private key is not generated as expected. If you have switched the tpm setting, you probably want to delete all the .ndn/ndnsec-* files/dirs, otherwise the inconsistency between pib and tpm may cause such a problem.

#30 - 02/09/2015 05:55 PM - Alex Afanasyev

I understand that. In the lab I had several problems which were caused by multiple installation of ndn-cxx (in /usr/local, in /opt/local/, in just /usr). Can you double check.

If nothing works, remove ~/.ndn folder. If nothing works again, remote /usr/local/etc/ndn/client.conf and /usr/local/etc/ndn/nfd.conf (and copy defaults).

If nothing works again, create /usr/local/etc/ndn/client.conf from sample and set file-based TPM (i forgot the correct name for that).

#31 - 02/09/2015 06:12 PM - Peter Gusev

Nothing helped. Please, see detailed logs below.

Checking ndn-cxx:

```
$ ls /usr/
X11     bin     lib     local       share
X11R6       include     libexec     sbin        standalone
$ ls /usr/local/include/
ndn     ndn-cpp     ndn-cxx     ndn-tlv     ndnboost
$ ls /usr/local
bin etc include lib man share
$ ls /opt/local/
Library bin etc include lib libexec man sbin    share   var
$ ls /opt/local/include/
autosprintf.h       cursesw.h       gettext-po.h        ncurses.h       tic.h
boost           cursslk.h       histedit.h      ncurses_dll.h       unctrl.h
bzlib.h         db46            iconv.h         openssl         unicode
cryptopp        editline        layout          panel.h         zconf.h
curses.h        eti.h           libcharset.h        sqlite3.h       zlib.h
cursesapp.h     etip.h          libintl.h       sqlite3ext.h
cursesf.h       expat.h         localcharset.h      term.h
cursesm.h       expat_external.h    menu.h          term_entry.h
cursesp.h       form.h          nc_tparm.h      termcap.h
```

Seems not a problem of conflicting ndn-cxx versions.

Checking removing .ndn folder (nfd has TRACE log level):

```
$ rm -Rf ~/.ndn
$ nfd-start
Error: PK_Signer: key too short for this signature scheme
Password:
1423533979.696405 TRACE: [NameTree] lookup /
1423533979.696482 TRACE: [NameTree] insert /
1423533979.696544 TRACE: [NameTree] Name / hash value = 0  location = 0
1423533979.696647 TRACE: [NameTree] Did not find /, need to insert it to the table
1423533979.696759 INFO: [StrategyChoice] setDefaultStrategy /localhost/nfd/strategy/best-route/%FD%02
1423533979.697251 INFO: [InternalFace] registering callback for /localhost/nfd/fib
1423533979.697386 INFO: [InternalFace] registering callback for /localhost/nfd/faces
1423533979.697523 INFO: [InternalFace] registering callback for /localhost/nfd/strategy-choice
1423533979.697649 INFO: [InternalFace] registering callback for /localhost/nfd/status
1423533979.697765 INFO: [FaceTable] Added face id=1 remote=internal:// local=internal://
1423533979.793466 FATAL: [NFD] PK_Signer: key too short for this signature scheme
1423533981.692590 INFO: [RemoteRegistrator] Load remote_register section in rib section
1423533981.693111 INFO: [RemoteRegistrator] Load remote_register section in rib section
1423533981.693287 INFO: [RibManager] Listening on: /localhost/nfd/rib
1423533981.796775 FATAL: [NRD] PK_Signer: key too short for this signature scheme
```

Deleting nfd.conf and client.conf:

```
$ sudo rm /usr/local/etc/ndn/nfd.conf /usr/local/etc/ndn/client.conf
$ sudo cp /usr/local/etc/ndn/nfd.conf.sample /usr/local/etc/ndn/nfd.conf
$ nfd-start
ERROR: TPM locator supplied does not match TPM locator in PIB: tpm-file: != tpm-osxkeychain:
libc++abi.dylib: terminating with uncaught exception of type ndn::KeyChain::MismatchError: TPM locator supplied does not match TPM locator in PIB:
tpm-file: != tpm-osxkeychain:
libc++abi.dylib: terminating with uncaught exception of type ndn::KeyChain::MismatchError: TPM locator supplied does not match TPM locator in PIB:
tpm-file: != tpm-osxkeychain:
$ sudo cp /usr/local/etc/ndn/client.conf.sample /usr/local/etc/ndn/client.conf
$ nfd-start
ERROR: TPM locator supplied does not match TPM locator in PIB: tpm-file: != tpm-osxkeychain:
libc++abi.dylib: terminating with uncaught exception of type ndn::KeyChain::MismatchError: TPM locator supplied does not match TPM locator in PIB:
tpm-file: != tpm-osxkeychain:
libc++abi.dylib: terminating with uncaught exception of type ndn::KeyChain::MismatchError: TPM locator supplied does not match TPM locator in PIB:
tpm-file: != tpm-osxkeychain:
/usr/local/bin/nfd-start: line 55:   554 Abort trap: 6           nrd
```

That's something new... Anyways, continue with file TPM:

Trying file TPM:

```
$ tail /usr/local/etc/ndn/client.conf
;   sqlite3
; pib=sqlite3

; "tpm" determines which Trusted Platform Module (TPM) should used by default in applications.
; If "tpm" is not specified, the default TPM will be used.
; Note that default TPM could be different on different system.
; If "tpm" is specified, it may have a value of:
;   file
;   osx-keychain
tpm=file
$ nfd-start
Error: PK_Signer: key too short for this signature scheme
1423534243.949454 INFO: [StrategyChoice] setDefaultStrategy /localhost/nfd/strategy/best-route/%FD%02
1423534243.949865 INFO: [InternalFace] registering callback for /localhost/nfd/fib
1423534243.949975 INFO: [InternalFace] registering callback for /localhost/nfd/faces
1423534243.950052 INFO: [InternalFace] registering callback for /localhost/nfd/strategy-choice
1423534243.950134 INFO: [InternalFace] registering callback for /localhost/nfd/status
1423534243.950211 INFO: [FaceTable] Added face id=1 remote=internal:// local=internal://
1423534244.096951 FATAL: [NFD] PK_Signer: key too short for this signature scheme
1423534245.944809 INFO: [RemoteRegistrator] Load remote_register section in rib section
1423534245.945287 INFO: [RemoteRegistrator] Load remote_register section in rib section
1423534245.945404 INFO: [RibManager] Listening on: /localhost/nfd/rib
1423534246.087423 FATAL: [NRD] PK_Signer: key too short for this signature scheme
```

#32 - 02/09/2015 06:35 PM - Alex Afanasyev

Whenever you change settings in client.conf, you need to remove ~/.ndn.

Btw. Is your home folder on some non-standard file system? There are known issues of sqlite3 and NFS.

#33 - 02/09/2015 10:19 PM - Junxiao Shi

> Excessing logging could have contributed to significant processing overhead, as it involves disk operations (i'm assuming log output was written into file).

I agree that disk operations for logging is a probable cause, because logger is implemented using std::clog <<, and this operation is synchronous.

To eliminate this cause, keep DEBUG log level, but redirect stderr to a file on ramdisk (eg. /dev/shm/nfd.log) instead of hard drive.

#34 - 02/10/2015 06:26 AM - Jeff Burke

> Whenever you change settings in client.conf, you need to remove ~/.ndn.
>
> Btw. Is your home folder on some non-standard file system? There are known issues of sqlite3 and NFS.

It's a typical Mac OS X filesystem. (hpfs, I suppose)

#35 - 02/10/2015 02:39 PM - Peter Gusev

Removing ~/.ndn folder and using default nfd.conf and client.conf helped.

I've run tests for the configuration when the bug was revealed (Forwarder DEBUG and UdpFace TRACE on intermediate hub) and observed two similar cases (again, test run was about ~10 min like usual).
For the configuration with logging disabled, I didn't observe this bug.

I'll keep you updated if I'll find out anything new this week.

#36 - 02/10/2015 09:09 PM - Junxiao Shi

> I've run tests for the configuration when the bug was revealed (Forwarder DEBUG and UdpFace TRACE on intermediate hub) and observed two similar cases (again, test run was about ~10 min like usual).
> For the configuration with logging disabled, I didn't observe this bug.

This is compatible with my guess in note-33.

Please run another experiment using the same system as note-35:

1. keep logging on (Forwarder DEBUG UdpFace TRACE)
2. redirect logs to /dev/shm/nfd.log (or another path that is mapped to memdisk), instead of hard drive

If problem disappears, log writing is the bottleneck.

If problem persists, log generation is the bottleneck.

Please also run experiments using a Linux box as router.

#37 - 02/12/2015 03:12 PM - Peter Gusev

I've run test on MacOS with writing logs to a file on a ramdisk (in-memory). The problem persists:

Case 1

NFD1

```
1423781206.076750   DEBUG: [Forwarder] onIncomingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00
1423781206.077225   DEBUG: [Forwarder] onOutgoingInterest face=259
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00
^ .18 sec, 181.510000 msec
1423781206.258735   DEBUG: [Forwarder] onIncomingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00
1423781206.259210   DEBUG: [Forwarder] onOutgoingInterest face=259
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00
^ .56 sec, 565.279000 msec
1423781206.824489   DEBUG: [Forwarder] onIncomingData face=259
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00/3/15678/522/0
1423781206.824841   DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00
1423781206.825089   DEBUG: [Forwarder] onOutgoingData face=265
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00/3/15678/522/0
```

NFD2

```
1423781206.647413   DEBUG: [Forwarder] onIncomingInterest face=266
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00
1423781206.647768   DEBUG: [Forwarder] onOutgoingInterest face=264
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00
^ .01 sec, 10.660000 msec
1423781206.658428   DEBUG: [Forwarder] onIncomingInterest face=266
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00
1423781206.660769   DEBUG: [Forwarder] onIncomingData face=264
```

```
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00/3/15678/522/0
1423781206.661062   DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00
1423781206.661313   DEBUG: [Forwarder] onOutgoingData face=266
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00/3/15678/522/0
^ .10 sec, 100.718000 msec
1423781206.762031   DEBUG: [Forwarder] onInterestFinalize
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00 satisfied
```

NFD3

```
1423781208.310953   DEBUG: [Forwarder] onIncomingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00
1423781208.311478   DEBUG: [Forwarder] onOutgoingInterest face=268
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00
1423781208.313922   DEBUG: [Forwarder] onIncomingData face=268
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00/3/15678/522/0
1423781208.314448   DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00
1423781208.314822   DEBUG: [Forwarder] onOutgoingData face=265
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15155/data/%00%00/3/15678/522/0
```

Case 2

NFD1

```
1423781224.225285   DEBUG: [Forwarder] onIncomingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
1423781224.225677   DEBUG: [Forwarder] onOutgoingInterest face=259
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
^ .16 sec, 168.895000 msec
1423781224.394572   DEBUG: [Forwarder] onIncomingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
1423781224.395050   DEBUG: [Forwarder] onOutgoingInterest face=259
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
^ .19 sec, 194.732000 msec
1423781224.589782   DEBUG: [Forwarder] onIncomingData face=259
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00/2/16213/540/0
1423781224.590154   DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
1423781224.590448   DEBUG: [Forwarder] onOutgoingData face=265
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00/2/16213/540/0
```

NFD2

```
1423781224.067487   DEBUG: [Forwarder] onIncomingInterest face=266
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
1423781224.068338   DEBUG: [Forwarder] onOutgoingInterest face=264
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
^ .16 sec, 168.644000 msec
1423781224.236982   DEBUG: [Forwarder] onIncomingInterest face=266
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
1423781224.237556   DEBUG: [Forwarder] onOutgoingInterest face=264
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
^ .19 sec, 191.210000 msec
1423781224.428766   DEBUG: [Forwarder] onIncomingData face=264
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00/2/16213/540/0
1423781224.429251   DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
1423781224.429647   DEBUG: [Forwarder] onOutgoingData face=266
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00/2/16213/540/0
^ .10 sec, 102.126000 msec
1423781224.531773   DEBUG: [Forwarder] onInterestFinalize
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00 satisfied
```

NFD3

```
1423781226.078766   DEBUG: [Forwarder] onIncomingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
1423781226.079160   DEBUG: [Forwarder] onOutgoingInterest face=268
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
1423781226.084523   DEBUG: [Forwarder] onIncomingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
1423781226.088801   DEBUG: [Forwarder] onIncomingData face=268
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00/2/16213/540/0
```

```
1423781226.089190   DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00
1423781226.089482   DEBUG: [Forwarder] onOutgoingData face=265
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15672/data/%00%00/2/16213/540/0
```

#38 - 02/13/2015 10:04 AM - Junxiao Shi

I don't think log generation could be causing this Bug, because every packet should generate the same amount of log lines, which is inconsistent with the pattern of "occurring nominally every 2-3 minutes".

Writing logs to ramdisk eliminates the bottleneck of disk access. However, NFD process still does synchronous syscalls to the kernel's filesystem module.

I suggest disabling logging for now, and reenable logging after #2513 completes.

#39 - 02/14/2015 02:08 PM - John DeHart

I have been going through some of the sets of log files that Peter gave me and have
come across something that might need explaining.

This is a case where the consumer seems to be the one that gets jammed up. The Hub and Producer
both seem to process and send Interests and Data in a timely way.
Eventually the Consumer APP resends the Interests.

Here is the Consumer NFD log messages for the original Interests:

```
1423189526.462412 DEBUG: [Forwarder] onIncomingInterest face=270
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%00
1423189526.462737 DEBUG: [Forwarder] onOutgoingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%00
1423189526.463005 DEBUG: [BestRouteStrategy2]
/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%00?ndn.MustBeFresh=1&ndn.InterestLifetime=5000&ndn.Nonce=30
5112104 from=270 newPitEntry-to=265
1423189526.463366 DEBUG: [Forwarder] onIncomingInterest face=270
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%01
1423189526.463653 DEBUG: [Forwarder] onOutgoingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%01
1423189526.463902 DEBUG: [BestRouteStrategy2]
/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%01?ndn.MustBeFresh=1&ndn.InterestLifetime=5000&ndn.Nonce=25
22549811 from=270 newPitEntry-to=265
1423189526.464237 DEBUG: [Forwarder] onIncomingInterest face=270
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%02
1423189526.464541 DEBUG: [Forwarder] onOutgoingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%02
1423189526.464798 DEBUG: [BestRouteStrategy2]
/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%02?ndn.MustBeFresh=1&ndn.InterestLifetime=5000&ndn.Nonce=16
6698611 from=270 newPitEntry-to=265
```

Here is the Consumer NFD log messages for the re-send of the Interests 600+ms later:

```
1423189527.095527 DEBUG: [Forwarder] onIncomingInterest face=270
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%00
1423189527.109664 DEBUG: [Forwarder] onOutgoingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%00
1423189527.126382 DEBUG: [BestRouteStrategy2]
/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%00?ndn.MustBeFresh=1&ndn.InterestLifetime=5000&ndn.Nonce=88
08564 from=270 retransmit-retry-to=265
1423189527.146407 DEBUG: [Forwarder] onIncomingInterest face=270
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%01
1423189527.164608 DEBUG: [Forwarder] onOutgoingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%01
1423189527.179107 DEBUG: [BestRouteStrategy2]
/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%01?ndn.MustBeFresh=1&ndn.InterestLifetime=5000&ndn.Nonce=30
30726010 from=270 retransmit-retry-to=265
1423189527.202391 DEBUG: [Forwarder] onIncomingInterest face=270
```

```
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%02
1423189527.216815 DEBUG: [Forwarder] onOutgoingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%02
1423189527.235703 DEBUG: [BestRouteStrategy2]
/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%02?ndn.MustBeFresh=1&ndn.InterestLifetime=5000&ndn.Nonce=43
6933208 from=270 retransmit-retry-to=265
```

What I find interesting is the time between the ‘onIncomingInterest’ message and the onOutgoingInterest.

For the original Interest 0 that is 0.325 ms
For the re-send Interest 0 that is 14.137 ms

Here is an example for an Interest slightly before the resend one above:

```
1423189526.896900 DEBUG: [Forwarder] onIncomingInterest face=270
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7646/data/%00%00
1423189526.912839 DEBUG: [Forwarder] onOutgoingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7646/data/%00%00
```

This one takes 15.939 ms and I believe this is NOT a resend but is the original request for this Interest.

I have not removed any log messages between the onIncomingInterest and the onOutgoingInterest.

What would cause this time to increase from a fraction of a ms to 14+ ms?

#40 - 02/14/2015 08:30 PM - Junxiao Shi

> What would cause this time to increase from a fraction of a ms to 14+ ms?

In BestRouteStrategy2, 'newPitEntry-to' and 'retransmit-retry-to' take different code paths, and it's normal for 'retransmit-retry-to' to be slower than 'newPitEntry-to'.
See BestRouteStrategy2::afterReceiveInterest function impl.

#41 - 02/15/2015 05:03 AM - John DeHart

But I think I have shown two cases, one which would be a new and one a retransmit and they both take over 14 ms.
And, 14 ms? That is a lot of processing time compared to 0.3 ms.

I am continuing to analyze the log files I have. I'll have more details soon but I think there is a period of time
where ALL Interests take this kind of time between inComing and outGoing.

#42 - 02/15/2015 08:22 AM - John DeHart

So, lets take a look at a larger set of log messages around the retransmit of the
original Interests in question:

```
1423189526.896900 DEBUG: [Forwarder] onIncomingInterest face=270
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7646/data/%00%00
1423189526.912839 DEBUG: [Forwarder] onOutgoingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7646/data/%00%00
1423189526.932740 DEBUG: [BestRouteStrategy2]
/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7646/data/%00%00?ndn.MustBeFresh=1&ndn.InterestLifetime=5000&ndn.Nonce=15
3505309 from=270 newPitEntry-to=265
1423189526.956603 DEBUG: [Forwarder] onIncomingInterest face=270
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7646/data/%00%01
1423189526.970187 DEBUG: [Forwarder] onOutgoingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7646/data/%00%01
1423189526.986366 DEBUG: [BestRouteStrategy2]
/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7646/data/%00%01?ndn.MustBeFresh=1&ndn.InterestLifetime=5000&ndn.Nonce=38
61357429 from=270 newPitEntry-to=265
1423189527.007187 DEBUG: [Forwarder] onIncomingInterest face=270
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7646/data/%00%02
1423189527.024547 DEBUG: [Forwarder] onOutgoingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7646/data/%00%02
1423189527.039788 DEBUG: [BestRouteStrategy2]
/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7646/data/%00%02?ndn.MustBeFresh=1&ndn.InterestLifetime=5000&ndn.Nonce=16
21725226 from=270 newPitEntry-to=265
1423189527.064591 DEBUG: [Forwarder] onIncomingInterest face=270
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7642/data/%00%01
1423189527.078460 DEBUG: [Forwarder] onOutgoingData face=270
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7642/data/%00%01/2/7906/263/0
1423189527.095527 DEBUG: [Forwarder] onIncomingInterest face=270
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%00
1423189527.109664 DEBUG: [Forwarder] onOutgoingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%00
1423189527.126382 DEBUG: [BestRouteStrategy2]
/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%00?ndn.MustBeFresh=1&ndn.InterestLifetime=5000&ndn.Nonce=88
08564 from=270 retransmit-retry-to=265
1423189527.146407 DEBUG: [Forwarder] onIncomingInterest face=270
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%01
1423189527.164608 DEBUG: [Forwarder] onOutgoingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%01
1423189527.179107 DEBUG: [BestRouteStrategy2]
/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%01?ndn.MustBeFresh=1&ndn.InterestLifetime=5000&ndn.Nonce=30
30726010 from=270 retransmit-retry-to=265
1423189527.202391 DEBUG: [Forwarder] onIncomingInterest face=270
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%02
1423189527.216815 DEBUG: [Forwarder] onOutgoingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%02
1423189527.235703 DEBUG: [BestRouteStrategy2]
/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/7643/data/%00%02?ndn.MustBeFresh=1&ndn.InterestLifetime=5000&ndn.Nonce=43
6933208 from=270 retransmit-retry-to=265
```

First we have three Interests for delta 7646. These are all treated by BestRouteStrategy2 as newPitEntry.

Then we have one Interest for delta 7642. This apparently has local data already cached as we see
an immediate (14ms) onOutgoingData.

Then we have the three Interests for delta 7643. These are all treated by BestRouteStrategy2 as retransmit.

If we look at the first of the delta 7646 Interests, the log messages for it show
15.939 ms between inComing and outGoing and 19.901 ms between outgoing and BestRouteStrategy2 messages.
That is 35.840 ms for this one Interest. With that kind of processing time for one Interest
we are down to less than 30 Interests a second.

Lets jump ahead to the Interest for delta 7642 that already has local data cached.
That takes 13.869 ms. Is it taking us that long to search the Content Store?

Am I missing something or is there something very strange going on here?
It could be that there is something going on the OS that is causing this, but whatever
it is we need to understand it.

#43 - 02/15/2015 09:33 AM - Junxiao Shi

First please make sure ndn-cxx and NFD are **compiled in release mode**. If either is compiled with --debug, any benchmark is meaningless.

The analysis below assumes this condition is satisfied. If not, redo the benchmark in release build of git-HEAD version.

In forwarding pipelines,

- PIT lookup happens for every incoming Interests
- if PIT entry already exists,
  - no ContentStore lookup
  - 'retransmit-retry-to'
- if PIT entry does not exist, it's inserted, then ContentStore lookup is performed
  - if HIT, Data is returned, and strategy is not triggered
  - if MISS, 'newPitEntry-to'

According to #2254 note-3 benchmark, a ContentStore lookup takes less than 0.020ms on OSX.

NFD as a userspace application does not have guaranteed CPU time.
It is possible that kernel is giving NFD less CPU time.

This factor can be analyzed by looking at kernel scheduler log, and can be eliminated by decreased NICE value of NFD process.

#44 - 02/15/2015 09:43 AM - Jeff Burke

Unless the machines are heavily loaded, it is unlikely that kernel scheduling is the cause of significant processing time variation. It seems worth considering such questions in terms of what's going on within NFD, rather than external factors, given that (I think) we are looking at light load circumstances.

#45 - 02/16/2015 02:59 PM - Peter Gusev

NFDs were built in release mode and there was no heavy load on the machines (NFD with logging enabled was taking up to 30% CPU on the intermediate hub).

#46 - 02/16/2015 03:05 PM - Junxiao Shi

note-39 and note-42 are extracted from CONSUMER node. NdnCon may take away considerable CPU time.

Please analyze kernel scheduler log to eliminate that factor, or decrease NICE value of NFD process to reduce its possible influence.

#47 - 02/17/2015 07:54 AM - John DeHart

Junxiao Shi wrote:

> note-39 and note-42 are extracted from CONSUMER node. NdnCon may take away considerable CPU time.
>
> Please analyze kernel scheduler log to eliminate that factor, or decrease NICE value of NFD process to reduce its possible influence.

I am unfamiliar with the 'kernel scheduler log' on MacOS.
Can someone give me a pointer to where and what this is?

#48 - 02/17/2015 08:23 AM - John DeHart

I think it makes sense to take a look at the OS scheduling, load, etc.

But, I would also submit that the problem is not isolated to just the consumer or producer
where NdnCon is running. I have nfd log examples from the hub where there are
currently unexplainable delays.

Here is one example of data processing on the hub that is quite long:

```
1423189811.070385 DEBUG: [Forwarder] onIncomingData face=264
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15753/data/%00%01/2/16297/543/0
1423189811.070726 DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15753/data/%00%01
1423189811.070974 DEBUG: [Strategy] beforeSatisfyPendingInterest
pitEntry=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15753/data/%00%01 inFace=264
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15753/data/%00%01/2/16297/543/0
1423189811.387738 DEBUG: [Forwarder] onOutgoingData face=266
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/15753/data/%00%01/2/16297/543/0
```

These are four log messages taken directly from nfd.log from one of Peter's experiments.
Nothing has been removed from between these messages.

'onIncomingData' to 'onIncomingData matching' : 0.341 ms

'onIncomingData matching' to 'beforeSatisfyPendingInterest': 0.248 ms

'beforeSatisfyingPendingInterest' to 'onOutgoingData' : 316.764 ms

From onIncomingData to onOutgoingData, we have an elapsed time of 317 ms.

#49 - 02/17/2015 10:59 AM - Junxiao Shi

If the Bug is in Forwarding, it should be reproducible using standard tools with any kind of face.

However, I'm unable to reproduce "more than 250ms delay" in either git-HEAD or v0.2.0, which indicates that it's not a Forwarding problem.

I have tried the following setup:

Ubuntu 14.10 with Boost 1.55, ndn-cxx:commit:c759a20046bce82a83c7956d3a28739b0be1f982,
nfd:commit:73b909a5142eb4a62aa9b3c1c9453bcb0f23c9fc, ndn-tlv-ping modified to use SHA256 signing and permit 1ms minimum interval.
Log level is default=INFO Forwarder=DEBUG; log is written to VirtualBox virtual disk which is mapped to a file on SATA hard drive.

```
nohup nfd-start &> nfd.log
nohup ndnpingserver /A &> ping-server.log &
nohup ndnping -i 5 -c 180000 /A &> ping-client.log &
```

NFD's CPU usage is around 25% of a core.
The maximum RTT reported in ping-client.log is 9.415ms, which means there isn't any delay longer than that.

I also tried v0.2.0:

NFD's CPU usage is around 23% of a core.
The maximum RTT reported in ping-client.log is 12.249ms, which means there isn't any delay longer than that.

#50 - 02/17/2015 11:08 AM - Jeff Burke

Thanks for trying this out. However, this experiment only shows that it is not an issue for your test traffic on Linux, not that there is no forwarder issue.

I would encourage you to try to reproduce this bug with ndnrtc itself, so you can examine the same type of traffic that is causing the problem seen by UCLA and WUSTL. Is that possible where you are, or could you login remotely to Peter's hub or WUSTL?

#51 - 02/17/2015 11:39 AM - John DeHart

Update on testing at WUSTL:

We have, so far, been unable to reproduce this problem on ONL (Ubuntu-based) at WUSTL.
We continue to add to our test scenario to try to make it look more like ndnrtc
but as yet, no luck triggering a similar processing delay.

We are also getting ready to try a Mac-based test at WUSTL very like Peter's set up.

#52 - 02/17/2015 11:47 AM - Jeff Burke

Hm, ok. Will look for mac specific issues, too, then.

#53 - 02/17/2015 01:15 PM - Peter Gusev

- File Archive.zip added

I've run nice'd NFD with logging into a ramdisk:

```
$ sudo nice -n -20 nfd-start &> /Volumes/ramdisk
```

and tested for 20 minutes - no single problem at all.

Then, I run nice'd NFD with logging into a file on harddisk:

```
$ sudo nice -n -20 nfd-start &> ~/nfd.log
```

I got bunch of errors both caused by NdnCon producer and NFD (see more NFD logs excerpts attached):

NFD1:

```
1424204601.596225   DEBUG: [Forwarder] onIncomingInterest face=266
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00
1424204601.596534   DEBUG: [Forwarder] onOutgoingInterest face=261
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00
^ .20 sec, 206.865000 msec
1424204601.803399   DEBUG: [Forwarder] onIncomingInterest face=266
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00
1424204601.803666   DEBUG: [Forwarder] onOutgoingInterest face=261
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00
^ .20 sec, 201.995000 msec
1424204602.005661   DEBUG: [Forwarder] onIncomingData face=261
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00/2/1898/63/1
1424204602.006062   DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00
1424204602.006286   DEBUG: [Forwarder] onOutgoingData face=266
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00/2/1898/63/1
^ .04 sec, 44.710000 msec
1424204602.050996   DEBUG: [Forwarder] onIncomingData face=261
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00/2/1898/63/1
1424204602.051283   DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00
```

NFD2:

```
1424204599.125677   DEBUG: [Forwarder] onIncomingInterest face=266
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00
1424204599.126187   DEBUG: [Forwarder] onOutgoingInterest face=264
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00
^ .40 sec, 405.842000 msec
1424204599.532029   DEBUG: [Forwarder] onIncomingData face=264
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00/2/1898/63/1
1424204599.532451   DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00
1424204599.532813   DEBUG: [Forwarder] onOutgoingData face=266
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00/2/1898/63/1
^ .04 sec, 44.085000 msec
1424204599.576898   DEBUG: [Forwarder] onIncomingInterest face=266
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00
1424204599.577247   DEBUG: [Forwarder] onOutgoingData face=266
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00/2/1898/63/1
^ .10 sec, 101.687000 msec
1424204599.678934   DEBUG: [Forwarder] onInterestFinalize
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00 satisfied
```

NFD3:

```
1424204577.093169   DEBUG: [Forwarder] onIncomingInterest face=265
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00
1424204577.093545   DEBUG: [Forwarder] onOutgoingInterest face=268
interest=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00
^ .05 sec, 58.318000 msec
1424204577.151863   DEBUG: [Forwarder] onIncomingData face=268
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00/2/1898/63/1
1424204577.152303   DEBUG: [Forwarder] onIncomingData
matching=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00
1424204577.152602   DEBUG: [Forwarder] onOutgoingData face=265
data=/ndn/edu/ucla/remap/ndnrtc/user/remap/streams/cam/low/delta/1834/data/%00%00/2/1898/63/1
```

What's interesting in this case is that producer was affected too - I saw enlarged delays b/w frame publishings at random time moments. Can extensive writing to hard disk in one process (NFD in this case) seriously affect other processes that do I/O (enlarged delays between frames being captured from camera)? NdnCon logging was set to default, so there were no extensive logging in this process.

I later run nice'd NFD with logging into a ramdisk test again and confirm - no problems have occurred.

Full logs:

- nice'd NFD + RAM disk - ~20 min
- nice'd NFD + hard disk - ~10 min
- nice'd NFD + RAM disk (check) - ~10 min

#54 - 02/18/2015 08:47 AM - John DeHart

I have followed up on Peter's question

> Can extensive writing to hard disk in one process (NFD in this case) seriously affect other processes that do I/O (enlarged delays between frames being captured from camera)?

I've just run these tests and so things are kind of rough but I wanted
to share as quickly as possible.

I have a simple bash script:

```
#!/bin/bash

rm -f date_and_disk_test.log
touch date_and_disk_test.log

while true
do
   gdate +%s.%N >> date_and_disk_test.log
   sleep 0.5

done
```

This should generate timestamps that are written to a file, each about 0.5 seconds after the previous.

While the above script is running, I run this on the same machine:

```
sudo find / -name "*a*" -print > /tmp/find.log
```

Then I use a crude python script to process the results:

```
#!/opt/local/bin/python

import sys
import os
import numpy

usage = "Usage: %s filename\n" % sys.argv[0]

def main(arv=None):
    filename = sys.argv[1]
    f = open(filename, 'r')
    line1 = f.readline()
    #print 'line1: ', line1
    while (line1 != ''):
        words = line1.split()
        time1 = words[0]
        line2 = f.readline()
        if line2 == '':
            #print 'Done'
            sys.exit(0)
        #print 'line1: ', line1
        #print 'line2: ', line2
        words = line2.split()
        time2 = words[0]
        timeDiff = float(time2) - float(time1)
        print time1, ' ', time2, ' ', timeDiff
        line1 = line2

if __name__ == '__main__':
    status = main()
    sys.exit(status)
```

And a crude bash script to display the results:

```
#!/bin/bash


./compareTimesInOut.py date_and_disk_test.log   >& timeDiffs
echo -n "total: " ; wc -l timeDiffs
echo -n " 0.4*: "   ; grep   " 0\.4" timeDiffs | wc -l
echo -n " 0.5*: "   ;grep   " 0\.5" timeDiffs | wc -l
echo -n " 0.6*: "   ;grep   " 0\.6" timeDiffs | wc -l
echo -n " 0.7*: "   ;grep   " 0\.7" timeDiffs | wc -l
echo -n " 0.8*: "   ;grep   " 0\.8" timeDiffs | wc -l
echo -n " 0.9*: "   ;grep   " 0\.9" timeDiffs | wc -l
echo -n " 1.0*: "   ;grep   " 1\.0" timeDiffs | wc -l
echo -n " 1.1*: "   ;grep   " 1\.1" timeDiffs | wc -l
echo -n " 1.2*: "   ;grep   " 1\.2" timeDiffs | wc -l
echo -n " 1.3*: "   ;grep   " 1\.3" timeDiffs | wc -l
echo -n " 1.4*: "   ;grep   " 1\.4" timeDiffs | wc -l
echo -n " 1.5*: "   ;grep   " 1\.5" timeDiffs | wc -l
echo -n " 1.6*: "   ;grep   " 1\.6" timeDiffs | wc -l
echo -n " 1.7*: "   ;grep   " 1\.7" timeDiffs | wc -l
echo -n " 1.8*: "   ;grep   " 1\.8" timeDiffs | wc -l
echo -n " 1.9*: "   ;grep   " 1\.9" timeDiffs | wc -l
echo -n " 2.* : "   ;grep   " 2\." timeDiffs | wc -l
echo -n " 3.* : "   ;grep   " 3\." timeDiffs | wc -l
echo -n " 4.* : "   ;grep   " 4\." timeDiffs | wc -l
```

I have run this on my MacBook Pro (MacOS), an ONL pc1core (Ubuntu) and an ONL pc2core (Ubuntu).
Here are the results:

MacOS:

```
total:      891 timeDiffs
 0.4*:        0
 0.5*:      721
 0.6*:       74
 0.7*:       26
 0.8*:       20
 0.9*:       14
 1.0*:       14
 1.1*:        8
 1.2*:        2
 1.3*:        1
 1.4*:        2
 1.5*:        3
 1.6*:        1
 1.7*:        0
 1.8*:        0
 1.9*:        1
 2.* :        3
 3.* :        1
 4.* :        0
```

ONL pc1core:

```
total: 1624 timeDiffs
 0.4*: 0
 0.5*: 1624
 0.6*: 0
 0.7*: 0
 0.8*: 0
 0.9*: 0
 1.0*: 0
 1.1*: 0
 1.2*: 0
 1.3*: 0
 1.4*: 0
 1.5*: 0
 1.6*: 0
 1.7*: 0
 1.8*: 0
 1.9*: 0
 2.* : 0
 3.* : 0
 4.* : 0
```

ONL pc2core:

```
total: 3458 timeDiffs
 0.4*: 0
 0.5*: 3458
 0.6*: 0
 0.7*: 0
 0.8*: 0
 0.9*: 0
 1.0*: 0
 1.1*: 0
 1.2*: 0
 1.3*: 0
 1.4*: 0
 1.5*: 0
 1.6*: 0
 1.7*: 0
 1.8*: 0
 1.9*: 0
 2.* : 0
 3.* : 0
 4.* : 0
```

So, on Ubuntu, I see no sizable impact of the find on the bash script running and outputting to a file.

On MacOS, well, things look pretty bad. There is as much as 3 seconds at times between my timestamps.
I think the answer to Peter's question is a resounding, "On MacOS, YES!"

Where do we go from here?

I will be running more tests like this with different nice levels.

We should probably also understand MacOS Spotlight indexing. How often does
it run to re-index? Could it be causing some of NdnCon's and nfd's problems?

If it turns out that these results hold up, then we will need to come up
with a set of configuration suggestions for MacOS NDN end hosts. Otherwise
apps will suffer greatly.

#55 - 02/18/2015 08:53 AM - John DeHart

Sorry, posted the wrong version of one of the scripts above.
The '\' to escape the '.' was left out.

```
#!/bin/bash

./compareTimesInOut.py date_and_disk_test.log   >& timeDiffs
echo -n "total: " ; wc -l timeDiffs
echo -n " 0.4*: "   ; grep   " 0.4" timeDiffs | wc -l
echo -n " 0.5*: "   ;grep   " 0.5" timeDiffs | wc -l
echo -n " 0.6*: "   ;grep   " 0.6" timeDiffs | wc -l
echo -n " 0.7*: "   ;grep   " 0.7" timeDiffs | wc -l
echo -n " 0.8*: "   ;grep   " 0.8" timeDiffs | wc -l
echo -n " 0.9*: "   ;grep   " 0.9" timeDiffs | wc -l
echo -n " 1.0*: "   ;grep   " 1.0" timeDiffs | wc -l
echo -n " 1.1*: "   ;grep   " 1.1" timeDiffs | wc -l
echo -n " 1.2*: "   ;grep   " 1.2" timeDiffs | wc -l
echo -n " 1.3*: "   ;grep   " 1.3" timeDiffs | wc -l
echo -n " 1.4*: "   ;grep   " 1.4" timeDiffs | wc -l
echo -n " 1.5*: "   ;grep   " 1.5" timeDiffs | wc -l
echo -n " 1.6*: "   ;grep   " 1.6" timeDiffs | wc -l
echo -n " 1.7*: "   ;grep   " 1.7" timeDiffs | wc -l
echo -n " 1.8*: "   ;grep   " 1.8" timeDiffs | wc -l
echo -n " 1.9*: "   ;grep   " 1.9" timeDiffs | wc -l
echo -n " 2.* : "   ;grep   " 2." timeDiffs | wc -l
echo -n " 3.* : "   ;grep   " 3." timeDiffs | wc -l
echo -n " 4.* : "   ;grep   " 4." timeDiffs | wc -l
```

#56 - 02/20/2015 10:18 PM - Junxiao Shi

- Category set to Core

20150220 conference call discussed this Bug.

It's believed that this Bug is caused by IO contention due to excess logging, and the current workaround is disabling logging.

#2513 may help, but a deeper problem is: debug logs shouldn't be enabled on a production system (both routers and laptops).

Alex suggests having an *access log* such as those from Apache HTTP Server, that writes one line per *request*.

Junxiao points out that the concept of *request* in NFD would be same as *packet*.

Beichuan thinks this unnecessary because IP end hosts and routers won't log every IP packet.

Lan suggests keeping logging disabled (set to INFO level or above), and use ndndump when packet-level network diagnostic is needed.

#57 - 02/21/2015 05:44 AM - John DeHart

My test results agree that it is IO contention but we should also note that it appears
to only be an issue on MacOS. I have not been able to produce any similar problems on Ubuntu.

I have also not been able to find any MacOS configuration that will work around the problem
while we have nfd DEBUG logging turned on.

We should also keep in mind that this is almost certainly a problem for applications as well
as nfd. If an application has heavy logging, or perhaps as part of the application they are
doing a lot of File IO, there could be performance problems on MacOS.

#58 - 02/21/2015 12:12 PM - Junxiao Shi

> I have also not been able to find any MacOS configuration that will work around the problem while we have nfd DEBUG logging turned on.

Could you please run an additional experiment: disable NFD logging, but run another process that does roughly same amount of disk writes, and see whether NFD is affected.

The outcome of this experiment can predict whether #2513 would be an effective solution.

#59 - 12/16/2015 09:48 PM - Alex Afanasyev

- Target version changed from v0.3 to v0.5

#60 - 01/22/2018 01:14 AM - Junxiao Shi

- Status changed from New to Closed

note-58 went unanswered, and there's no more complaint of this issue, so I assume it's resolved.

#61 - 01/22/2018 03:16 AM - Jeff Burke

I don't understand how non-response means the issue is resolved. We are still seeing packet processing delays in NDN-RTC, I think, as recently discussed; but let's leave closed and re-open if it becomes relevant again.

Files

IMG_5708.JPG.jpeg	661 KB	02/09/2015	Peter Gusev
Archive.zip	7.34 KB	02/17/2015	Peter Gusev