Infrastructure - Task #7036

Feature # 6498 (Closed): V2 Metacat MN and CN Support

MN.updateSystemetadata calls CN.synchronize to update the system metadata on the cn

2015-04-10 18:12 - Jing Tao

Status:	Closed	**Start date:**	2015-04-10
Priority:	Normal	**Due date:**	
Assignee:	Jing Tao	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	CCI-2.0.0		
Milestone:	None	**Story Points:**	
Product Version:	*		

Description

The method should call cn.updateSystemmetadata synchronizely and asynchronizely?

We have long emails to discuss the issue.
Hi all,

I totally see Ben’s point, and it would be better from the MN perspective to get an affirmative return right away. My original thought was that the MN would just call CN.systemMetadataChanged(), and that’d be that, just like we do in v1 where the CN holds authoritative system metadata and we make an asynchronous call to MN.systemMetadataChanged().

The size of the system metadata is certainly not the issue. I was thinking more about the number of HTTPS requests the CNs will realistically concurrently handle. Our new architecture is push-based in that the MNs hold authoritative system metadata, and on any change, we push the change to the CN. In this model, there’s no throttling. In our current pull model (d1_sync), we throttle to 8 concurrent threads per MN when calling MN.getSystemMetadata(). So, in the event that an MN gets a boat load of updated content, we’d see a flurry of un-throttled calls to the CN. Maybe that’s not a big deal. Perhaps a decent MN implementation strategy would be to 1) call CN.updateSystemMetadata() (which blocks), and in the event of a timeout, 2) tries X times, then 3) falls back to CN.systemMetadataChanged() (async).

Anyway - one question: If the call to the CN blocks, does that mean that the client call to the MN also needs to wait for a successful MN.updateSystemMetadata() call? I’m just wondering if the investigator will see the delay if the MN call to the CN isn’t very speedy.

Cheers,
Chris

On Apr 9, 2015, at 3:17 PM, Robert Nahf rnahf@epscor.unm.edu wrote:

> I'm not worried so much about a long list of pending updateSystemMetadata requests, because that would assumedly be FIFO, in which case, I don't think the MN would be burdened. We already have potentially the same situation with v1.CN.setAccessPolicy() anyway.
>
> Do any of our replication processes create long-lasting locks on the systemMetadata? I think that's the only potential source of delay (meaning, the only process I don't understand enough to know how it does locking).
>
> Rob
>
> Rob Nahf
> ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
> Programmer Analyst for DataONE,
> Office of the Vice President for Research
> The University of New Mexico
> office: 505.814.7600 x8110
> mobile: 520.440.0339
>
> rnahf@unm.edu
> ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
>
> Did you ask.dataone.org?

On Thu, Apr 9, 2015 at 11:44 AM, Christopher Jones cjones@nceas.ucsb.edu wrote:

Andrei,

Well, hmm. I would think that we would want these decoupled, and that we should change the API documentation. Of course, ideally we would want it to know it succeeded, but I’m just not sure that we can ensure that it will happen immediately(ish).

That’s my vote, but others may want to chime in.

Cheers,
Chris

On Apr 9, 2015, at 9:41 AM, Andrei Buium <andreib@dataone.unm.edu> wrote:

Hi Chris,
That makes sense to me. (The diagram in Matt's link shows that's the case too.)
I was assuming it's synchronous because the CN API documentation says it returns true "if the update was successful". Should this be changed to return true if the request to update was sent successfully?
And we'll favor performance (no bottleneck if the MN is doing this in one thread) over assuring that the transaction went through?

On Wed, Apr 8, 2015 at 5:18 PM, Christopher Jones <cjones@nceas.ucsb.edu> wrote:

Hi Andrei,
My understanding was that the call to CNCore.updateSystemMetadata () would be asynchronous and not block. Waiting for the CN to process, potentially a long queue of requests, may be a bottleneck for the MN. If the CN returns an HTTP 200, the MN can move on and the CN can deal with the new SM as soon as it can.
Do I have this right?

Cheers,
Chris

On Apr 8, 2015 3:43 PM, Robert Nahf <rnahf@epscor.unm.edu> wrote:
>
> Also, assuming that an update of the authoritative MN's systemMetadata is accompanied by a new dateSystemMetadataModified value, the object will be picked up by synchronization, and the CN will update their systemMetadata then.
>
>
>
> Rob Nahf
> ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
> Programmer Analyst for DataONE,
> Office of the Vice President for Research
> The University of New Mexico
> office: 505.814.7600 x8110
> mobile: 520.440.0339
> rnahf@unm.edu
> ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~
>
> Did you ask.dataone.org?
>
> On Wed, Apr 8, 2015 at 3:36 PM, Andrei Buium <andreib@dataone.unm.edu> wrote:
>>
>> Oh perfect, I was trying to find a diagram like that earlier.
>> I was wondering about v2 mainly.
>> It looks like the MN will be responsible for pushing out changes made, to the CN using CNCore.updateSystemMetadata() and to other MNs using MNStorage.updateSystemMetadata(). That makes sense.
>>

```
    >> Thanks!
    >> -Andrei
    >>
    >>
    >> On Wed, Apr 8, 2015 at 3:17 PM, Matt Jones <jones@nceas.ucsb.edu> wrote:
    >>>
    >>> Andrei --
    >>>
    >>> That looks correct for our v1 API.  I think the MN.updateSystemMetadata() is part of the changes in v2 API that shift maintenance authority for system metadata to Member Nodes.  This is not yet deployed in production.  See the architecture docs for a description of the pending changes:
    >>>
    >>> http://jenkins-1.dataone.org/jenkins/job/API%20Documentation%20-%20trunk/ws/api-documentation/build/html/design/SystemMetadata.html#roadmap-to-system-metadata-control-changes-draft-to-be-reviewed
    >>>
    >>> Matt
    >>>
    >>>
    >>> On Wed, Apr 8, 2015 at 12:49 PM, Andrei Buium <andreib@dataone.unm.edu> wrote:
    >>>>
    >>>> Hey all,
    >>>> I have a couple questions on MN updates of system metadata.
    >>>>
    >>>> Here is the sequence of events I've put together from the API documentation.
    >>>> How accurate is it?
    >>>>
    >>>> - an MN updates an object's sysmeta locally
    >>>> - that MN notifies the CN using CNCore.updateSystemMetadata()
    >>>> updateSystemMetadata() blocks / runs synchronously
    >>>> (it returns true if the update was successful, implying it's synchronous)
    >>>> this call updates the sysmeta on the CN
    >>>> - the CN is responsible for notifying other MNs (those holding replicas) of the update
    >>>> it calls MNRead.systemMetadataChanged() for each MN that needs to know
    >>>> systemMetadataChanged() runs asynchronously, returning true if the message was received
    >>>> - the replica-holding MNs can use CN.getSystemMetadata() to update their copy
    >>>> we don't enforce when this happens
    >>>> The above has the CN using MN.systemMetadataChanged() to notify of the update.
    >>>> There's also MN.updateSystemMetadata(). When / by whom is that used?
    >>>>
    >>>> Thanks!
    >>>> -Andrei
    >>>>
    >>>> _______________________________________________
    >>>> developers mailing list
    >>>> developers@dataone.org
    >>>> To unsubscribe or change your subscription, email support@dataone.org
    >>>
    >>>
    >>
    >>
    >> _______________________________________________
    >> developers mailing list
    >> developers@dataone.org
    >> To unsubscribe or change your subscription, email support@dataone.org
    >
    >
```

____
developers mailing list
developers@dataone.org

```
To unsubscribe or change your subscription, email support@dataone.org
```

Related issues:

Blocked by Infrastructure - Story #7073: Create CN.synchronize API on cn_rest...	**Closed**	**2015-04-30**

History

#1 - 2015-04-15 19:54 - Jing Tao

- Assignee set to Jing Tao

#2 - 2015-04-30 21:37 - Jing Tao

client -> mn.updateSystemMetadata
-> cn.synchronize (this api is in the d1_rest_service. async, put the system metadata into a queue)

-> d1_synchronization will access the queue
-> call CN.updateSystemMetadata(in CN.storage)
-> for each MN except authoritative MN (in d1_synchronization):
-> mn.systemMetadataChanged

https://docs.google.com/document/d/1AuWEofJVlj__UnBm-X62THYGJQ-F7CFfJEJC9zWH-v4/edit

#3 - 2015-04-30 21:58 - Jing Tao

- Blocked by Story #7073: Create CN.synchronize API on cn_rest_service added

#4 - 2015-07-16 22:54 - Jing Tao

- Status changed from New to In Progress

- % Done changed from 0 to 30

#5 - 2015-07-28 21:29 - Jing Tao

- Subject changed from The behavior when MN.updateSystemetadata is called. to MN.updateSystemetadata calls CN.synchronize to update the system metadata on the cn

#6 - 2015-07-29 22:59 - Jing Tao

Hi Rob:

After you upgrade the cn-dev, I tested this again and still got the null pointer exception:

```
metacat 20150729-22:38:24: [DEBUG]: D1NodeService.updateSystemMetadata() called. [edu.ucsb.nceas.metacat.dataone.D1NodeService]
metacat 20150729-22:38:24: [DEBUG]: Storing System Metadata to store: tao.13350.1
[edu.ucsb.nceas.metacat.dataone.hazelcast.SystemMetadataMap]
metacat 20150729-22:38:24: [DEBUG]: Entry added/updated to System Metadata map: tao.13350.1
[edu.ucsb.nceas.metacat.dataone.hazelcast.HazelcastService]
org.dataone.service.exceptions.ServiceFailure: Unexpected Exception in CN.synchronize: progress: (b) got pid from request: tao.13350.1::
java.lang.NullPointerException
at org.dataone.service.util.ExceptionHandler.deserializeXml(ExceptionHandler.java:633)
at org.dataone.service.util.ExceptionHandler.deserializeXmlAndThrowException(ExceptionHandler.java:517)
at org.dataone.service.util.ExceptionHandler.deserializeAndThrowException(ExceptionHandler.java:363)
at org.dataone.service.util.ExceptionHandler.deserializeAndThrowException(ExceptionHandler.java:313)
at org.dataone.service.util.ExceptionHandler.filterErrors(ExceptionHandler.java:107)
at org.dataone.service.util.ExceptionHandler.filterErrors(ExceptionHandler.java:82)
at org.dataone.client.rest.HttpMultipartRestClient.doPostRequest(HttpMultipartRestClient.java:448)
at org.dataone.client.v2.impl.MultipartCNode.synchronize(MultipartCNode.java:702)
at edu.ucsb.nceas.metacat.dataone.MNodeService.updateSystemMetadata(MNodeService.java:2248)
at edu.ucsb.nceas.metacat.restservice.v2.MNResourceHandler.updateSystemMetadata(MNResourceHandler.java:1638)
at edu.ucsb.nceas.metacat.restservice.v2.MNResourceHandler.handle(MNResourceHandler.java:269)
at edu.ucsb.nceas.metacat.restservice.D1RestServlet.doPut(D1RestServlet.java:102)
at javax.servlet.http.HttpServlet.service(HttpServlet.java:649)
at javax.servlet.http.HttpServlet.service(HttpServlet.java:727)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:303)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:208)
at edu.ucsb.nceas.metacat.restservice.D1URLFilter.doFilter(D1URLFilter.java:48)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:241)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:208)
at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:220)
at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:122)
at org.apache.catalina.authenticator.AuthenticatorBase.invoke(AuthenticatorBase.java:501)
at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:170)
at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:98)
at org.apache.catalina.valves.AccessLogValve.invoke(AccessLogValve.java:950)
at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:116)
at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:408)
at org.apache.coyote.ajp.AjpProcessor.process(AjpProcessor.java:193)
at org.apache.coyote.AbstractProtocol$AbstractConnectionHandler.process(AbstractProtocol.java:607)
at org.apache.tomcat.util.net.JIoEndpoint$SocketProcessor.run(JIoEndpoint.java:313)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1145)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:615)
at java.lang.Thread.run(Thread.java:745)
metacat 20150729-22:38:24: [ERROR]: It is a DataONEBaseException and its detail code is 4961 and its code is 500
[edu.ucsb.nceas.metacat.dataone.MNodeService]
metacat 20150729-22:38:24: [ERROR]: Can't update the systemmetadata of pid tao.13350.1 in CNs since Unexpected Exception in CN.synchronize:
progress: (b) got pid from request: tao.13350.1:: java.lang.NullPointerException [edu.ucsb.nceas.metacat.dataone.MNodeService]
```

#7 - 2015-08-14 23:39 - Jing Tao

- Status changed from In Progress to Closed

- % Done changed from 30 to 100

- translation missing: en.field_remaining_hours set to 0.0

Called MN.updateSystemMetadata on the mn-demo-6 and we saw that updated MN systemMeta was synchronized to the cn-dev.