2009-06-09 - Special Topic - Large Datastreams

Time/Place:

- **Time:** 10:00 am, Eastern Daylight Time (GMT -04:00, New York)
- **Public Chat Room:**
  - Join via web (just enter a unique nick): https://www.mibbit.com/?server=irc.freenode.net&channel=%23fcrepo
  - Or point your IRC client to #fcrepo on irc.freenode.net

Agenda:

1. Discussion of "Large Datastream" issues/solutions in Fedora
   a. Copying large (100s of MB to a few GB) files into Fedora as Managed Content is slow. Currently the most efficient way to do this (within Fedora's APIs) is to make the content accessible on a remote server and pass it to Fedora by reference. See FCREPO-453: Allow retrieval of content via file: URI scheme
   b. Copying huge (many GB/TB) files from their current location in Fedora is impractical, yet people want a solution better than "R" and "E" datastreams. See FCREPO-426: Managed External Datastreamsand FCREPO-439: In-Place "Ingest" of local files
   c. Fedora reads FOXML files into memory. FOXML files contain all versions of inline XML, and this can cause Out of Memory errors when 1) inline XML is large, or 2) Many revisions of an inline XML datastream (e.g. RELS-EXT) exist. See FCREPO-492: Allow DC, RELS-EXT, etc to be Managed Content
2. Begin forming a technical team for addressing these issues

Summary:

- Discussed approach for allowing file: resolution (Kai is working on FCREPO-453)
- Decided to create new issue for mime validation (not part of FCREPO-453)
- Discussed "Managed External" datastreams (Kai will attach code to FCREPO-426)
- Identified "Managed Redirect" datastreams as possible feature to coincide with "Managed External"
- Identified multiplexing by size and/or type (content or mime) as feature
- Discussed allowing DC, RELS-EXT, etc to be managed content (AaronB can work on this in early July)

Carrying this work forward:

- See the Large Datastreams Page in the Fedora Roadmap.

Chat Log:

(copied from http://fedora-commons.org/irclogs/index.php?date=2009-06-09)

[10:04] <cwilper> Hi all, thanks for coming. I thought I'd try this vs. voice to see if it works for us.
[10:05] <cwilper> This is our second "Special Topics" meeting...we hope to have one scheduled per month going forward.
[10:07] <cwilper> Agenda: 1. Discussion of "Large Datastream" issues/solutions, 2. Begin forming a technical team for addressing such issues.
[10:08] <cwilper> I trust folks got a chance to look at the specific issues on the agenda? http://fedora-commons.org/confluence/x/YI_v
[10:09] <cwilper> Let's go through A,B,C and see if we're missing anything.
[10:10] <cwilper> A: "Copying large (100s of MB to a few GB) files into Fedora as Managed Content is slow. Currently the most efficient way to do this (within Fedora's APIs) is to make the content accessible on a remote server and pass it to Fedora by reference."
[10:11] <cwilper> There's been some talk about allowing Fedora to suck in (local) content if it's given via the "file:" scheme at ingest time
[10:12] <cwilper> Who's been following this: https://fedora-commons.org/jira/browse/FCREPO-453
[10:12] <ddavis> Large datastreams may be located in SANs, Locally attached storage, NAS and thus are equivalent to files not remotely served objects.
[10:13] <eddie31415> I haven't really looked at the proposed patch for #453, has anyone?
[10:14] <ddavis> This could also characterize files staged in local-like storage prior to ingest.
[10:14] <cwilper> Any thoughts +/- on the suggested way of addressing the performance? I suspect it will help...
[10:14] <eddie31415> I like the idea, however, of writing a policy to limit the allowed file:/// urls
[10:15] <cwilper> I looked at it, but wasn't sure about the policy bit.
[10:16] <ddavis> Some products accept a file pointer by essentially an administrative process which is then pass into the OS ownership of the repository.
[10:16] <ddavis> So the user per-se does not have access to "file" urls.
[10:16] <eddie31415> me neither. I had assumed that if we'd allow for file urls, we'd allow for configuration in fcfg for an explicitly allowed path or paths
[10:17] <eddie31415> dan, I don't know what you mean there
[10:18] <cwilper> Seems like having the policy would be nice for more advanced scenarios, but I worry about people having trouble putting the right policy in place w/the complexities of xacml.
[10:19] <eddie31415> agreed. but i think we can mitigate that by just shipping a default policy
[10:19] <ddavis> If the content is placed on the file system as a pre-ingest item and identified as a file URL. Then during ingest the file URL is rewritten for a managed location ID.
[10:19] <cwilper> true...maybe a couple examples.
[10:20] <cwilper> It's worth noting also that this patch isn't just addressing the ingest piece of the puzzle, but also applies to "E" and "R" datastreams at runtime.
[10:23] <bbranan> note that Kai (who submitted the patch) isn't able to join us today due to the FIZ proxy blocking IRC traffic
[10:23] <cwilper> arg. even the web gateway huh.
[10:24] <bbranan> apparently so
[10:24] <cbeer> cwilper: sorry, I haven't looked at the code -- what do you mean by applies to E/R?
[10:24] <eddie31415> strange...it's over ssl, can't see how that even works
[10:24] <cwilper> btw, benjamin armintor submitted the patch..

[10:25] <bbranan> right, sorry, Kai just added the tracker item
[10:25] <cwilper> cbeer: it applies to "ExternalContentManager", which is the module in Fedora that resolves content.
[10:26] <cwilper> ExternalContentManager is used at ingest time, when sucking in the content for a "M[anaged]" datastream, but it's also used at runtime. However, not consistently.
[10:28] <eddie31415> so, are we ok (in principle, assuming we deal appropriately w/ security issues) with allowing file urls?
[10:28] <cwilper> So Benjamin's patch proposes consistently using ExternalContentManager everywhere so that, if ECManager is extended to support file: uris, Fedora is capable of resolving them regardless of the context.
[10:28] <cwilper> I am.
[10:28] <cwilper> Hi Kai
[10:28] <eddie31415> me too 🙂
[10:28] <kstrnad> Hi all
[10:29] <cwilper> I just want to make sure there's a way to do it securely...and I think the default should be that no file: URLs are resolved for that reason.
[10:29] <kstrnad> I have already applied Benjamin's patch
[10:30] <eddie31415> what's the verdict?
[10:31] <cwilper> Kai: in a branch? Or locally?
[10:31] <kstrnad> Seems to be working so far. However, as he also pointed out some additional refactoring is needed
[10:31] <kstrnad> locally
[10:32] <kstrnad> The access to ECM is still not consistent. And there should be some kind of lookup mechanism for protocols that is more convenient.
[10:33] <kstrnad> Also, mimetype detection should be done differently.
[10:35] <cwilper> Kai: anything you want to get opinions on while we're here or will you just keep forging ahead with it?
[10:38] <cwilper> all: Additional comments on issue-A before we move on to issue-B?
[10:38] <kstrnad> I would like to know what I could do about the mime types. Afaik there is currently no way of detecting them apart from relying on http properties ..
[10:38] <birkland> I like mime-util
[10:39] <birkland> http://sourceforge.net/projects/mime-util
[10:39] <eddie31415> from the tracker it sounds as if benjamin was proposing a small refactoring of DatastreamReferencedContent
[10:39] <cwilper> seems current/active
[10:40] <kstrnad> see http://64.62.228.82/url/MLGJF5 for a comparison
[10:41] <kstrnad> thanks Aaron
[10:41] <kstrnad> the question is if we need this
[10:42] <cwilper> kai: need refactoring of DatastreamReferencedContent, or need mime detection?
[10:42] <kstrnad> possibly both ?
[10:43] <bbranan> it seems that mime detection would be a nice plug-in feature
[10:44] <kstrnad> so, maybe another tracker item for now ?
[10:44] <eddie31415> i think mime-detection on ingest/modify would be a nice service/plugin...--what bill said 😉
[10:44] <cwilper> Ben's comment said: "The addition of a method to serve content from a DatastreamReferencedContent (rather than a String url) would allow the ExternalContentManager to use the datastream's defined mime-type "
[10:44] <bbranan> yes, sounds like a different tracker item to me
[10:45] <eddie31415> cwilper: right, sounds like that would still be needed so that benjamin's patch can use the declared mime-type
[10:45] <kstrnad> Yes, Chris. And he is right. But still the mime type detection is problematic
[10:47] <cwilper> re: mime detection. In what cases is the mime type not known? It seems that at ingest time, if you pass it "by reference" it's declared in the datastream element in the foxml or in the fedora api request. At dissemination time, it's also available in the Datastream struct as a property.
[10:48] <cwilper> err...attribute
[10:49] <kstrnad> it could be detected automatically
[10:50] <kstrnad> or verified
[10:51] <eddie31415> this just seems like another feature: to allow the mime-type to be unspecified at ingest and then passed off to a service for automatic detection or as kai mentions, for verification
[10:51] <ddavis> In a preservation use case you need both "what you declared it to be" and "what it really is"
[10:51] <cwilper> yes, for validation. wanted to make sure i wasn't missing something w/r/t existing ExternalContentManager use cases
[10:52] <cwilper> it does sounds like a separate feature
[10:52] <bbranan> in the case where we don't know the mimetype on datastream ingest we can go with just a default value until the type-detection feature is added
[10:52] <eddie31415> but as far as ben's patch is concerned, if we were to go ahead w/ it, we'd keep the current ingest process, but still need to patch DatastreamReferencedContent to support mime info
[10:53] <kstrnad> yes
[10:53] <eddie31415> i vote we move along to B in the interest of time =)
[10:53] <bbranan> second
[10:53] <cwilper> B: "Copying huge (many GB/TB) files from their current location in Fedora is impractical, yet people want a solution better than "R" and "E" datastreams."
[10:54] <cwilper> Proposed "External Managed" ds type by escidoc: https://fedora-commons.org/jira/secure/attachment/10310/escidoc-largefiles.pdf
[10:58] <cwilper> Did we ever get code for this? (Eddie?)
[10:58] <eddie31415> I don't think so
[10:58] <kstrnad> Yes, yiu did 😉 (Matthias)
[10:59] <eddie31415> oh =)
[10:59] <cwilper> I remember being worried about an explosion of the # of "Datastream types" E/R/M/X/etc...
[10:59] <cwilper> But this wouldn't seem to lead to that.
[10:59] <kstrnad> Anyway, is this still a way we want to persue? I think with Akubra this approach is a bit outdated? (Matthias)
[11:00] <cwilper> I was going to ask the same of Matthias.
[11:00] <cwilper> I don't think it's necessarily outdated.
[11:01] <kstrnad> We can definitly provide the code once again, but it was written for Fedora 2.x (Matthias)
[11:01] <bbranan> has there been discussion of using an Akubra which is running on a machine apart from Fedora?
[11:03] <cwilper> Akubra basically resolves URIs to content. The way it's plugged in (Fedora 3.2), it's not used in a more sophisticated way that that. For example, partial reads aren't allowed by Fedora's higher-level api, so Akubra isn't plugged in in this way.
[11:04] <kstrnad> I am not too familiar with Akubra, but the main idea behind our propoasal was to allow to send back only parts of a file to the user. The disseminator architecture at that time required to always pipe through the whole file through Fedora before the disseminator could select the small portion requested by the user.
[11:05] <cwilper> Akubra does support communicating over different protocols, but I'm not sure that's the big thing here...
[11:06] <cbeer> cwilper: if we're interested in adding External Managed, I'd propose a similar Redirect Managed as well. It could help in situations with dedicated servers for delivering large files (streaming servers for video, what have you) but continue to let Fedora be aware of the underlying content for

preservation, etc
[11:08] <cwilper> kai: No matter where the content is stored, the requirement would be that you can read part of it, then, right? Akubra is a java api that speaks in terms of streams. Therefore, you can skip() and read bits at a time, but it doesn't specify how that might be exposed at an http level, for example.
[11:11] <MatthiasR> I think that this would be sufficient. I imagine a disseminator calling Akubra for accessing e.g. a video stream, extraction the portion the user is interested in and sending it back in a way in an appropriate way to the users client.
[11:11] <cwilper> cbeer: trying to understand... what additional bits of information would be available to Fedora for "Redirect Managed"?
[11:12] <MatthiasR> External content has no versioning whatsoever. Using Redirect Managed would mean that Fedora would still keep track of versions by providing this information to the storage component (e.g. by providing relative path info and filenames)
[11:12] <cbeer> cbeer: if i understand External managed right, Fedora would have knowledge about the filesystem path to the file, but when a client requests the datastream, it'd receive a redirect to a URL
[11:15] <Mike_Durbin> Another relevant requirement (for us at least) would be the ability to have different datastreams in different stores (ie, big archival stuff to a tape system, small delivery stuff to local disk). Different datastream types or the ability to configure multiple akubra stores, would seem to better accommodate this.
[11:15] <bbranan> so both External Managed and Redirect Managed work the same as Managed from an ingest/update perspective - except that the content is being stored on a remote system
[11:16] <bbranan> on dissemination Redirect Managed would would the same as Redirect, and External Managed would work the same as External
[11:16] <cwilper> matthias: "E" datastreams today can have multiple versions. You can keep the same old location (URL) or have a new one when you call modifyDatastream(...).
[11:18] <bbranan> it seems like the ability to do partial reads could be considered a separate feature, one that could be used on many of the datastream types
[11:19] <MatthiasR> Yes, but you have to keep track of the versions instead of letting Fedora do the job. For "E" datastreams, you create a new file version, then update the FOXML by calling modifyDatastream() to let Feodra know.
[11:20] <cwilper> Mike: The AKubra "Multiplexing" implementation (akubra-mux) is trying to enable just that. See http://fedora-commons.org/documentation/akubra/0.1/site/apidocs/org/fedoracommons/akubra/mux/AbstractMuxStore.html
[11:22] <cwilper> bill: it is - see https://fedora-commons.org/jira/browse/FCREPO-182[]
[11:25] <cwilper> Matthias, right. So modification to External Managed can occur in one step instead of two.
[11:27] <MatthiasR> chris: well, that depends. I would say: it occurs in a differen torder. You would ask Fedora to update a datastream. Fedora would update the FOXML and then send you the updated location, where the disseminator would store the file. Ouch, that should be done in a transaction.
[11:28] <cwilper> Will add 3 issues to the list for "Large Datastreams": 1) getting partial content (regardless of datastream type), aka FCREPO-182, 2) "Managed Redirect", and 3) Multiplexing based on size
[11:28] <MatthiasR> I should add: "you" in that case is the disseminator.
[11:30] <cwilper> Matthias, it seems like POST/PUT/DELETE support for disseminators would be needed to do this "right", correct?
[11:30] <bbranan> sounds like multiplexing based on type is also needed (video to streaming server)
[11:30] <MatthiasR> chris: well, that's my pet project 🙂
[11:31] <MatthiasR> bill: or based on the content model?
[11:31] <cwilper> Bill, yes that's another useful dimension for multiplexing.
[11:32] <cwilper> The "mux" impl started as abstract because we figured there would be a lot of rules people would want to employ for making the "which backend store does it go to" decision.
[11:32] <bbranan> Matthias: yes, type could be determined in more than one way (MIME, content model, etc)
[11:34] <cwilper> re:writable disseminators: https://fedora-commons.org/jira/browse/FCREPO-500
[11:35] <bbranan> chris: good plan, I can definitely see people wanting to add new rules - if it's Thursday, all files go to store Y
[11:36] <cwilper> Oh yes. But I think a couple obvious impls are good...most commonly people want to do it by size and/or type.
[11:36] <cwilper> Let's move on to C?
[11:37] <cwilper> I'd like to finish in the next 20 mins so we can keep it under 2 hours 🙂
[11:38] <cwilper> C: "Fedora reads FOXML files into memory. FOXML files contain all versions of inline XML, and this can cause Out of Memory errors when: 1) inline XML is large, 2) Many revisions of inline XML (e.g. RELS-EXT) exist"
[11:38] <cwilper> I believe this one hits eSciDoc in both ways, correct?
[11:40] <kstrnad> yes this is a recurring problem for us
[11:41] <cwilper> would you say 1 or 2 is the more pressing problem?
[11:43] <kstrnad> i think the rels-ext revisions are more problematic, however big xml files are difficult either way
[11:44] <bbranan> it seems that the suggestion to allow all datastreams to be M rather than X https://fedora-commons.org/jira/browse/FCREPO-492 would fix both problems
[11:45] <kstrnad> yes, at some point they would have to be M
[11:45] <cwilper> bill: I guess the obvious question after FCREPO-492 would be: "why would one use Inline XML" instead of "Managed Content" for some datastreams and not others?
[11:45] <bbranan> we could potentially even do a transition from X to M when they hit a certain size
[11:45] <MatthiasR> Addressing both issues properly (i.e. no more reading FOXML into memory) would probably mean re-architecturing major parts of Fedroa.Allowing every datastream to be managed ould at east be a good workaround
[11:46] <eddie31415> agree w/ chris
[11:48] <eddie31415> it seems confusing to have both and requires users to do a lot of head scratching to figure out if their case would be better served inline vs managed and to what purpose
[11:48] <bbranan> chris: minor concerns, like wanting to keep everything together in one file, particularly for small objects like sdefs come to mind
[11:48] <MatthiasR> I think the nice thing about X is that you really have everything pertaining your digital object in a single file.
[11:49] <eddie31415> but at what point does an object w/ inline become too big?
[11:49] <cwilper> "It depends"
[11:49] <eddie31415> exactly
[11:49] <bbranan> we could address that concern by allowing ingest/export to happen as a single file, but then break it apart in the background
[11:50] <ddavis> I don't see why Fedora would not be agile enough to look for inlined managed content then if not found looking for it in the object storage.
[11:51] <kstrnad> finding a set rule may be hard. It comes down to heap sizing and usage
[11:52] <cwilper> dan: that's one technique for implementing it. Thinking more about this, there are certainly a lot of tradeoffs. But I wonder if we'd have a recommendation for the "average user" to just use managed for everything a keep it simple?
[11:52] <MatthiasR> I think I would want to keep control as a user where Fedora actually stores my "stuff". Why not let the user decide?
[11:52] <ddavis> My thought too.
[11:52] <eddie31415> but whatever we default w/ should "just work"
[11:53] <MatthiasR> I wouldn't even change the current "default" behaviour. I think our eSciDoc use cases are very special, even though I am sure they will become more and more important as more users try to build VRE's w/ Fedora.
[11:55] <bbranan> the problem is that if we continue with the current setup then things will be great for a while, until you start getting enough system usage for memory errors to begin to show up, at that poing migrating all of the X datastreams to M could be painful
[11:58] <cwilper> ...well it seems clear that FCREPO-492 needs to happen regardless. I'm not sure exactly how to describe the other proposed thing: to

have Fedora *not* read "X" datastreams into memory?
[11:59] <bbranan> moving all X datastreams to M would likely incur a small performance hit, as many operations likely need to see both the foxml and either dc or rels-ext
[12:01] <cwilper> But you require potentially a lot more inodes if you store things in separate files. And if you have less of a chance of the data being contiguous on the block device. I suspect that could slow things down significantly in certain cases.
[12:02] <cwilper> Let's wrap up...
[12:03] <cwilper> Anyone who hasn't spoken want to get involved in implementing solutions for what we've talked about?
[12:04] <cwilper> It seems clear that Kai's got the lead on A (FCREPO-453)
[12:07] <cwilper> B and C need more discussion, but we should at least get the code for "Managed External" attached to FCREPO-426 (Kai, can you do that?)
[12:08] <birkland> For C, if we simply allow managed "special" datastreams if the user specifies them, there may not be much work involved, right?
[12:08] <kstrnad> yes of course
[12:09] <cwilper> And we identified a few more suggested features. I will put these together on the "Large Datastreams" page linked from the roadmap.
[12:10] <bbranan> chris: I agree that for C, at least FCREPO-492 is in the clear to be worked
[12:10] <cwilper> Aaron, for C not sure what you mean by managed "special"?
[12:10] <cwilper> Oh sorry, you mean FCREPO-492
[12:11] <birkland> was using that as a blanket term for datastreams such as DC, RELS-EXT
[12:11] <birkland> i.e. the ones were talking about
[12:12] <birkland> I don't think there much, if any, specialized code that will break if we allow them to be 'M', or something else
[12:12] <cwilper> Aaron: Gotcha. That's what FCREPO-492 proposes to do, and I agree, it shouldn't be too difficult.
[12:13] <birkland> I think it would be mostly verification, testing, and tying up loose ends
[12:13] <cwilper> Anyone want to sign up for that one now?
[12:14] <birkland> I can, with a caveat that I'll be gone for the rest of the month on vacation and won't be able to jump into it until some time in early July
[12:16] <cwilper> Excellent.. I think early July is fine. Unless a serious problem comes up with 3.2, I don't suppose we'll be pushing for a 3.3 release 'till early Fall.
[12:17] <cwilper> Thanks everyone. I'll put notes on the wiki.