Arvados - Bug #8878

Keep: sudden appearance of "missing" blocks

04/04/2016 04:02 PM - Peter Grandi

Status:	Closed	**Start date:**	04/04/2016
Priority:	Normal	**Due date:**	
Assigned To:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

I had done a "garbage collection" before Easter as follows:

```
2016/03/24 17:06:10 Read and processed 417 collections
2016/03/24 17:06:13 Blocks In Collections: 514668, 
Blocks In Keep: 961866.
2016/03/24 17:06:13 Replication Block Counts:
 Missing From Keep: 0, 
 Under Replicated: 0, 
 Over Replicated: 1650, 
 Replicated Just Right: 513018, 
 Not In Any Collection: 447198. 
Replication Collection Counts:
 Missing From Keep: 0, 
 Under Replicated: 0, 
 Over Replicated: 11, 
 Replicated Just Right: 406.
2016/03/24 17:06:13 Blocks Histogram:
2016/03/24 17:06:13 {Requested:0 Actual:1}:     444455
2016/03/24 17:06:13 {Requested:0 Actual:2}:       2743
2016/03/24 17:06:13 {Requested:1 Actual:1}:     513018
2016/03/24 17:06:13 {Requested:1 Actual:2}:       1647
2016/03/24 17:06:13 {Requested:1 Actual:3}:          3
2016/03/24 17:06:15 Sending trash list to http://keep9.gcam1.example.com:25107
2016/03/24 17:06:15 Sending trash list to http://keep3.gcam1.example.com:25107
2016/03/24 17:06:15 Sending trash list to http://keep6.gcam1.example.com:25107
2016/03/24 17:06:15 Sending trash list to http://keep5.gcam1.example.com:25107
2016/03/24 17:06:15 Sending trash list to http://keep0.gcam1.example.com:25107
2016/03/24 17:06:15 Sending trash list to http://keep4.gcam1.example.com:25107
2016/03/24 17:06:15 Sending trash list to http://keep7.gcam1.example.com:25107
2016/03/24 17:06:15 Sending trash list to http://keep8.gcam1.example.com:25107
2016/03/24 17:06:15 Sending trash list to http://keep1.gcam1.example.com:25107
2016/03/24 17:06:15 Sent trash list to http://keep1.gcam1.example.com:25107: response was HTTP 200
 OK
2016/03/24 17:06:15 Sent trash list to http://keep0.gcam1.example.com:25107: response was HTTP 200
 OK
2016/03/24 17:06:16 Sent trash list to http://keep4.gcam1.example.com:25107: response was HTTP 200
 OK
2016/03/24 17:06:16 Sent trash list to http://keep9.gcam1.example.com:25107: response was HTTP 200
 OK
2016/03/24 17:06:16 Sent trash list to http://keep3.gcam1.example.com:25107: response was HTTP 200
 OK
2016/03/24 17:06:16 Sent trash list to http://keep5.gcam1.example.com:25107: response was HTTP 200
 OK
2016/03/24 17:06:16 Sent trash list to http://keep8.gcam1.example.com:25107: response was HTTP 200
 OK
2016/03/24 17:06:16 Sent trash list to http://keep7.gcam1.example.com:25107: response was HTTP 200
 OK
2016/03/24 17:06:16 Sent trash list to http://keep6.gcam1.example.com:25107: response was HTTP 200
 OK
```

Then after uploading two 4GB collections over the past week, we have deleted the 2 4GB collections that they were meant to replace, and then I run the Data Manager again in dry-run mode and the outcome is:

```
2016/04/04 12:51:17 Read and processed 421 collections
2016/04/04 12:51:19 Blocks In Collections: 782548,
Blocks In Keep: 716788.
2016/04/04 12:51:19 Replication Block Counts:
 Missing From Keep: 65760,
 Under Replicated: 0,
 Over Replicated: 41180,
 Replicated Just Right: 675608,
 Not In Any Collection: 0.
Replication Collection Counts:
 Missing From Keep: 3,
 Under Replicated: 0,
 Over Replicated: 13,
 Replicated Just Right: 405.
2016/04/04 12:51:19 Blocks Histogram:
2016/04/04 12:51:19 {Requested:1 Actual:0}:      65760
2016/04/04 12:51:19 {Requested:1 Actual:1}:     675608
2016/04/04 12:51:19 {Requested:1 Actual:2}:      41177
2016/04/04 12:51:19 {Requested:1 Actual:3}:          3
```

It is disconcerting to see {Requested:1 Actual:0}:   65760 (around 4GiB) but also {Requested:1 Actual:2}:   41177 (around 2.5GiB).

The two collections that were uploaded to replace the two that were deleted should have been exactly identical byte for byte, as the re-uploads were from the same files using identically the same file list.

A question I have is whether there is a tool that can tell me which collections and files within them have missing hashes. I think that I can easily modify some of my scripts to that purpose, so I would like to know if there is a tool that I can use as a double check.

The other question is whether I can run with Data Manager further consistency checks, for example as to verifying the hashes of the data blocks.

Related issues:

Related to Arvados - Story #8724: [Keep] Block validation script	**Resolved**	**03/16/2016**
Related to Arvados - Bug #8910: [SDK] arv-put should save manifest text on AP...	**Resolved**	
Related to Arvados - Feature #8993: arv-put: options for 3 modes of "resumption"	**Closed**	**04/14/2016**

History

#1 - 04/05/2016 07:47 AM - Peter Grandi

Oop I forgot this important detail, which is the dry-run report of Data Manager just after the garbage collection:

```
2016/03/24 17:45:21 Read and processed 417 collections
2016/03/24 17:45:22 Blocks In Collections: 514668,
Blocks In Keep: 514668.
2016/03/24 17:45:22 Replication Block Counts:
 Missing From Keep: 0,
 Under Replicated: 0,
 Over Replicated: 1650,
 Replicated Just Right: 513018,
 Not In Any Collection: 0.
Replication Collection Counts:
 Missing From Keep: 0,
 Under Replicated: 0,
 Over Replicated: 11,
 Replicated Just Right: 406.
2016/03/24 17:45:22 Blocks Histogram:
2016/03/24 17:45:22 {Requested:1 Actual:1}:     513018
2016/03/24 17:45:22 {Requested:1 Actual:2}:       1647
2016/03/24 17:45:22 {Requested:1 Actual:3}:          3
```

Between then and now we have (almost) only uploaded 6x ~4TiB data collections, two of them twice (so 8x uploads) and then deleted the earlier uploads of the two that were twice uploaded.
IIRC.

#2 - 04/05/2016 08:23 AM - Peter Grandi

So sequence of (almost all) events:

- We have a 30TiB Keep with around 25TiB of data, we add another 30TiB, we upload an 8TiB collection.

- We then upload a 24TiB collection, and that registers as 100% complete but registering the collection manifest fails because of out-of-RAM.
- At this point the 60TiB total is almost full.
- We try again to upload the 24TiB data set as 6x 4TiB collections, counting on the hash blocks being reused. But that does not happen much, so we get to 100% space used, with around 24TiB in unreferenced blocks as expected. Of the 6x collection 4 report failure to upload because of out-of-space, and 2x (collections 2 and 3) report a successful upload, which is strange, as we did not quite have 8TiB free.
- Run the Data Manager in dry-run mode, and some local scripts, and roughly 1.2TiB of blocks are reported as unnecessary duplicates. I write a script to delete the relevant files and the Data manager in dry-run mode reports no problems. The 1.2TiB free allows us to continue using Keep for some test jobs.
- After consultation it turns out that hashes are computed on a bytestream, not per-file, by arv-put, so the re-upload in 6 subsets resulted in different hashes (except for the first I guess).
- So I run the Data Manager in garbage collection mode and that seems successful. It seems to me to free up a bit too much space, but then we had deleted some collections over the past few months.
- So I run the Data Manager in dry-run mode and the reports looks good.
- We re-upload subsets 1, 4, 5, 6 of the 6x and all report successfully completed and registered.
- We re-upload subsets 2 and 3 to sligthly differently named collections. Curiously even if the list of files passed to arv-put is exactly the same as that for the previous upload, this consumes some space.
- We delete the collections that were the original uploads of subsets 2 and 3. Out server space graphs don't change.
- The Data Manager in dry-run mode reports 4TiB of missing blocks and 2.5TiB of duplicate blocks.

Related issues: https://dev.arvados.org/issues/8769https://dev.arvados.org/issues/8867

I am about to upload a free-space graph for the relevant server volumes that shows the various phases.

Note: some trivial formatting mistakes fixed 2016-04-08.

#3 - 04/05/2016 08:24 AM - Peter Grandi

- File 160329_arvDiskFree.png added

Free space graph 2016-03-29 (a week ago, just after Easter).

#4 - 04/07/2016 07:33 PM - Peter Amstutz

Hi Peter,

> collections 2 and 3 report a successful upload, which is strange, as we did not quite have 8TiB free.

This sounds like collections 2 and 3 have more deduplication than the other batches. (#8791 is the story to improve the deduplication behavior generally).

> We re-upload subsets 2 and 3 to sligthly differently named collections.

Do you mean just "arv-put --name" is different?

> We delete the collections that were the original uploads of subsets 2 and 3. Out server space graphs don't change.

When you write "delete the collections" do you mean using the "trash" button in Workbench, using "arv collection delete", or are you directly deleting the collection record in the Postgres console?

Does "delete the collections" include running Data Manager garbage collection and/or deleting blocks using your own script?

I'm trying to understand the sequence of events better:

1. You "delete the collections" (assuming this does not include running data manager GC or any other scripts that would delete actual data)
2. "server space graphs don't change" which also suggests that nothing changed on disk
3. running data manager immediately afterwards is reporting "Missing From Keep" blocks when you would actually expect "Not In Any Collection"?

Are you suggesting that files are reported missing when nothing was physically deleted? That would be extremely puzzling. If that is the case, the most likely reason would be if data manager failed to contact one of the keep servers to get a block index, but instead of failing with an error it just went ahead. However, that wouldn't explain why the over replicated block count would go up as well.

Does data manager return the same results over multiple runs?

> A question I have is whether there is a tool that can tell me which collections and files within them have missing hashes. I think that I can easily modify some of my scripts to that purpose, so I would like to know if there is a tool that I can use as a double check.

Data Manager has most of that information, but doesn't currently have the feature to directly report missing blocks and tell you exactly which collections and files are affected.

> The other question is whether I can run with Data Manager further consistency checks, for example as to verifying the hashes of the data blocks.

This is being worked on right now: #8724

#5 - 04/07/2016 11:24 PM - Peter Amstutz

Also, this whole mess could have been avoided if arv-put saved the manifest text from failed collection create API call to facilitate forensic recovery. I've added a story for this at #8910

#6 - 04/08/2016 08:34 AM - Peter Grandi

> > collections 2 and 3 report a successful upload, which is strange, as we did not quite have 8TiB free.
>
> This sounds like collections 2 and 3 have more deduplication than the other batches.

I don't know what the chances of that happening are, because the 25TB dataset is "DDD", which is 3,000 ".bam" files plus their ".bai" files, and file tells me that they are gzipped. Also, since they are uploaded as "bytestreams" even the same data at the same offset in the file might have different hashes because of different offsets in the "bytestream". Then there is the chance that two compressed-data blocks of 64MiB have by coincidence the same hash.

> > We re-upload subsets 2 and 3 to sligthly differently named collections.
>
> Do you mean just "arv-put --name" is different?

Yes, identical command lines from the same shell history with "_2b" and "_3b" instead of "_2" and "_3" as name suffix.

> > We delete the collections that were the original uploads of subsets 2 and 3. Out server space graphs don't change.
>
> When you write "delete the collections" do you mean using the "trash" button in Workbench

Workbench trash.

> Does "delete the collections" include running Data Manager garbage collection and/or deleting blocks using your own script?

No, the space graph should show that.

> I'm trying to understand the sequence of events better:
>
> 1. You "delete the collections" (assuming this does not include running data manager GC or any other scripts that would delete actual data)

Yes, just Workbench trash. Unfortunately not expecting anything strange I did not run dry-run Data Manager just before that. I run it just after that to check whether there were any unreferenced block hashes, which I did not expect.

> 1. "server space graphs don't change" which also suggests that nothing changed on disk

That's what I suppose. But the space taken by the collections on disk is lower than what I would expect; the attached space graph should show that.

> 1. running data manager immediately afterwards is reporting "Missing From Keep" blocks when you would actually expect "Not In Any Collection"?

Reports ~4TB (65,760 blocks) "Missing from Keep", and ~2.5TB (41,177 blocks) of duplicates. I can imagine the duplicates happen because of placement issues during upload. It is the 4TB of missing blocks that perplex me, as between the garbage collection on the 24th and now we have only done uploads and no garbage collections. The attached space graph hows that.

> Are you suggesting that files are reported missing when nothing was physically deleted?

Yes.

> That would be extremely puzzling. If that is the case, the most likely reason would be if data manager failed to contact one of the keep servers to get a block index, but instead of failing with an error it just went ahead. However, that wouldn't explain why the over replicated block count would go up as well.

That is one of the possibilities that occurred to me too. So my plan, which has not happened yet, was to match the manifests in the SQL database with the find of hash files in the keepstores, which I gathered using 'find' on each of them.

My current best "hunch" is that the problem happened during the upload of the original "_2" and "_3" collections, when the Keepstore filetrees were 100% full. Perhaps the keepstore daemon does not handle that too well.

> Does data manager return the same results over multiple runs?

Yes.

> > A question I have is whether there is a tool that can tell me which collections and files within them have missing hashes. I think that I can easily modify some of my scripts to that purpose, so I would like to know if there is a tool that I can use as a double check.
>
> Data Manager has most of that information, but doesn't currently have the feature to directly report missing blocks and tell you exactly which collections and files are affected.

A "-verbose" flag would be welcome.

> > The other question is whether I can run with Data Manager further consistency checks, for example as to verifying the hashes of the data blocks.
>
> This is being worked on right now: #8724

As to that the idea that file hashes are computed on a "bytestream" rather than file base in arv-put by default (which cannot be changed) is a bit disappointing, because it would be nice to be able to compute similar block hashes outside Keep and double check them.

#7 - 04/10/2016 08:34 PM - Peter Amstutz

- File datamanager added

- File keep_block_to_file.py added

To help you recover while we continue to try and diagnose the underlying bug, I've added some additional reporting to datamanager, along with an auxiliary python script. These are in the 8912-missing-blocks-report branch and for your convenience I've attached a binary of datamanager and a copy of the python script keep_block_to_file.py to this ticket.

Running datamanager -dry-run -extra-reports will produce some timestamped files, the formats are timestamp_uuid_index.txt and timestamp_uuid_missing.txt. The former is the indexes returned by each keepstore to datamanager, the latter is the collections with missing blocks.

You then use keep_block_to_file.py *_missing.txt to get the list of specific files within each collection which have missing blocks.

#8 - 04/11/2016 10:11 AM - Peter Grandi

- File 160404_arvDiskFreeNotes.png added

Attached a copy of the freespace graph with notes.

#9 - 04/11/2016 01:41 PM - Peter Grandi

Just run the extended {{datamanager}}:

```
manager@hebrides:~$ ls -ld *index.txt *missing.txt
-rw-rw-r-- 1 manager manager  815976 Apr 11 13:34 2016-04-11T13:34:34Z_gcam1-4zz18-2ob6abc4jcp8bzy_missing.txt
-rw-rw-r-- 1 manager manager 1131438 Apr 11 13:34 2016-04-11T13:34:34Z_gcam1-4zz18-idu3dl9xvky1rke_missing.txt
-rw-rw-r-- 1 manager manager  814506 Apr 11 13:34 2016-04-11T13:34:34Z_gcam1-4zz18-u88ijgfjz8y5cjh_missing.txt
-rw-rw-r-- 1 manager manager 2363775 Apr 11 13:34 2016-04-11T13:34:34Z_gcam1-bi6l4-1j5ilfcaic0ude8_index.txt
-rw-rw-r-- 1 manager manager 4140699 Apr 11 13:34 2016-04-11T13:34:34Z_gcam1-bi6l4-3bbqgnonzaux8k0_index.txt
-rw-rw-r-- 1 manager manager 4179572 Apr 11 13:34 2016-04-11T13:34:34Z_gcam1-bi6l4-cga8vx6ihr6rq8a_index.txt
-rw-rw-r-- 1 manager manager 3781747 Apr 11 13:34 2016-04-11T13:34:34Z_gcam1-bi6l4-h8s8tzzsldt1kqh_index.txt
-rw-rw-r-- 1 manager manager 3784458 Apr 11 13:34 2016-04-11T13:34:34Z_gcam1-bi6l4-l3i1sdlfr38jjtn_index.txt
-rw-rw-r-- 1 manager manager 2354311 Apr 11 13:34 2016-04-11T13:34:34Z_gcam1-bi6l4-pkp2i499ekxrrtk_index.txt
-rw-rw-r-- 1 manager manager 2371703 Apr 11 13:34 2016-04-11T13:34:34Z_gcam1-bi6l4-sztrsz43hs9w28d_index.txt
-rw-rw-r-- 1 manager manager 2367525 Apr 11 13:34 2016-04-11T13:34:34Z_gcam1-bi6l4-tkk9ygrftp95d8r_index.txt
-rw-rw-r-- 1 manager manager 2354036 Apr 11 13:34 2016-04-11T13:34:34Z_gcam1-bi6l4-vr8fxkbqu84w2um_index.txt
-rw-rw-r-- 1 manager manager 4133209 Apr 11 13:34 2016-04-11T13:34:34Z_gcam1-bi6l4-z7kt05yz1ng7f5w_index.txt
```

```
manager@hebrides:~$ wc -l *missing.txt
```

```
  19428 2016-04-11T13:34:34Z_gcam1-4zz18-2ob6abc4jcp8bzy_missing.txt
  26939 2016-04-11T13:34:34Z_gcam1-4zz18-idu3dl9xvky1rke_missing.txt
  19393 2016-04-11T13:34:34Z_gcam1-4zz18-u88ijgfjz8y5cjh_missing.txt
  65760 total
```

#10 - 04/11/2016 01:48 PM - Peter Grandi

I had expected "_2b" and "_3b" (renamed to "-103001-104929" and "-104930-106319") but it is instead "_1" (renamed to "-100004-102995") and "_4" and "_5":

```
            uuid             |                 name
-----------------------------+---------------------------------------
 gcam1-4zz18-u88ijgfjz8y5cjh | DDD_WGS_EGAD00001001114_4
 gcam1-4zz18-2ob6abc4jcp8bzy | DDD_WGS_EGAD00001001114_5
 gcam1-4zz18-idu3dl9xvky1rke | DDD_WGS_EGAD00001001114-100004-102995
(3 rows)

arvados=# select uuid,name from collections where name like 'DDD%' order by name;
            uuid             |                 name
-----------------------------+---------------------------------------
 gcam1-4zz18-6m5xpktrmjsl9p1 | DDD_VARINFO_2015-08-14
 gcam1-4zz18-idu3dl9xvky1rke | DDD_WGS_EGAD00001001114-100004-102995
 gcam1-4zz18-4lrd2smmw9u6yni | DDD_WGS_EGAD00001001114-103001-104929
 gcam1-4zz18-gbf33ha80xmcxk6 | DDD_WGS_EGAD00001001114-104930-106319
 gcam1-4zz18-oytr46viwy7ikxj | DDD_WGS_EGAD00001001114_2
 gcam1-4zz18-0yjqcuawjvntzaq | DDD_WGS_EGAD00001001114_3
 gcam1-4zz18-u88ijgfjz8y5cjh | DDD_WGS_EGAD00001001114_4
 gcam1-4zz18-2ob6abc4jcp8bzy | DDD_WGS_EGAD00001001114_5
 gcam1-4zz18-pxfnfhjpx218suk | DDD_WGS_EGAD00001001114_6
(9 rows)
```

#11 - 04/11/2016 02:20 PM - Peter Grandi

On 2016-03-29 {{arv-put}} reported success with:

```
librarian@biscay$ time arv-put --replication 1 --resume --project-uuid gcam1-j7d0g-k25rlhe6ig8p9na
 --name DDD_WGS_EGAD00001001114_1 $(< ~/l1)
arv-put: Resuming previous upload from last checkpoint.
         Use the --no-resume option to start over.
4282537M / 4282637M 100.0%
Collection saved as 'DDD_WGS_EGAD00001001114_1'
gcam1-4zz18-idu3dl9xvky1rke

real    3439m58.024s
user    687m21.092s
sys     330m3.305s

librarian@sole$ time arv-put --replication 1 --resume --project-uuid gcam1-j7d0g-k25rlhe6ig8p9na
 --name DDD_WGS_EGAD00001001114_4 $(< ~/l4)
arv-put: Resuming previous upload from last checkpoint.
         Use the --no-resume option to start over.
4228156M / 4228192M 100.0%
Collection saved as 'DDD_WGS_EGAD00001001114_4'
gcam1-4zz18-u88ijgfjz8y5cjh

real    3979m16.365s
user    864m45.066s
sys     382m57.657s

2016-03-27 14:42:04 arvados.keep[13012] DEBUG: Request: PUT http://keep8.gcam1.camdc.genomicsplc.com:25107/c98
04a232cd8802a4315a879d701c6f2
2016-03-27 14:42:04 arvados.keep[13012] INFO: PUT 200: 3307667 bytes in 23.0910778046 msec (136.608 MiB/sec)
2016-03-27 14:42:04 arvados.keep[13012] DEBUG: KeepWriterThread <KeepWriterThread(Thread-479423, started 13975
8258013952)> succeeded c9804a232cd8802a4315a879d701c6f2+3307667 http://keep8.gcam1.camdc.genomicsplc.com:25107
/
4311611M / 4311683M 100.0%
Collection saved as 'DDD_WGS_EGAD00001001114_5'
gcam1-4zz18-2ob6abc4jcp8bzy

real    4137m27.994s
user    867m54.495s
sys     364m37.592s

librarian@sole$ du -sm DDD*_[1-6]
4282638 DDD_WGS_EGAD00001001114_1
```

```
4286404 DDD_WGS_EGAD00001001114_2
4525616 DDD_WGS_EGAD00001001114_3
4228194 DDD_WGS_EGAD00001001114_4
4311684 DDD_WGS_EGAD00001001114_5
4321623 DDD_WGS_EGAD00001001114_6
```

#12 - 04/11/2016 02:41 PM - Peter Grandi

The ~4,000 minutes reported for the {{arv-put}}s above are nearly 3 days, and the uploads were started March 24th, so they finished during the weekend as the free space graph shows.

#13 - 04/11/2016 02:52 PM - Peter Grandi

- File 2016-04-11T13_34_34Z_filescount.txt added

```
manager@hebrides:~$ python2 160410_keep_block_to_file.py *_missing.txt >| 2016-04-11T13:34:34Z_files.txtmanage
r@hebrides:~$ cut -d, -f 2 2016-04-11T13:34:34Z_files.txt | sort | uniq -c >| 2016-04-11T13:34:34Z_filescount.
txt
manager@hebrides:~$ wc -l 2016-04-11T13:34:34Z_filescount.txt
1101 2016-04-11T13:34:34Z_filescount.txt
manager@hebrides:~$ wc -l 2016-04-11T13:34:34Z_files.txt
66858 2016-04-11T13:34:34Z_files.txt
```

#14 - 04/14/2016 12:14 PM - Peter Grandi

My impressions from a long IRC discussion:

- arv-put by default uses option --resume and that means that it keeps a history of blocks that has already uploaded in previous partial uploads.
- The blocks in that list are deemed, when --resume is used, to be present on the Keep servers without any further checks.
- If a garbage collection intervenes and those blocks are not shared with already registered manifests, they are going to be deleted, but arv-put, since it does not check, will not know.
- There is a partial protection against mishaps in that each block in the list kept by arv-put is tagged by the garbage-collector TTL/delayed-delete time, so arv-put won't consider it present on Keep after that time.
- The garbage-collector TTL/delayed-delete time is sort of the same as that described under permissions in: https://dev.arvados.org/projects/arvados/wiki/Keep_server#Permission, that is arv-put will assume that a block is going to be available for as long as its permissions tocken lasts.
- But if the garbage-collector delayed-delete time is reduced in-between a failed upload and its resumption, arv-put will believe the delayed-delete time associated with the permissions token at upload time, not the current delayed-delete time.

While each aspect of the previous story makes sense on its own, my impression is that it means overall that Keep is no longer stateless/idempotent, but there is state outside it that might become stale and yet its currency is not verified. It is a classic case of treating a hint value as if it were a cached value.

Overall I think that this is the result of attempting to optimize arv-put which is both a very critical tool, and has some subtle state and logic inside. If there was a way to treat the --resume list as a hint to be verified it would be safer.

#15 - 04/14/2016 12:22 PM - Peter Grandi

Anyhow, the story above seems to validate my impressions that the missing data had not been lost, but never uploaded, given the free space graphs showing unlikely amounts of deduplication, and the Data Manager showing a good state just after garbage collection, and eventually the list of files with missing blocks showed that they were all at the beginning of the list of files for each sub-collection.

So a re-re-re-upload with arv-put --no-resume (to be sure) and exactly the same file list (because of #8769) previously used should have resulted in the missing blocks being uploaded and all other blocks being found as present already.

Currently the re-re-re-upload of subcollections 1, 4, 5 has reached nearly 60% and indeed the Data manger in -dry-run mode reports a wholesome situation:

```
2016/04/14 10:38:27 Returned 10 keep disks
2016/04/14 10:38:27 Replication level distribution: map[1:741370 2:41177 3:3]
2016/04/14 10:38:29 Blocks In Collections: 782550,
Blocks In Keep: 782550.
2016/04/14 10:38:29 Replication Block Counts:
 Missing From Keep: 0,
 Under Replicated: 0,
 Over Replicated: 41180,
 Replicated Just Right: 741370,
 Not In Any Collection: 0.
Replication Collection Counts:
 Missing From Keep: 0,
 Under Replicated: 0,
 Over Replicated: 13,
 Replicated Just Right: 416.
2016/04/14 10:38:29 Blocks Histogram:
2016/04/14 10:38:29 {Requested:1 Actual:1}:     741370
```

```
2016/04/14 10:38:29 {Requested:1 Actual:2}:      41177
2016/04/14 10:38:29 {Requested:1 Actual:3}:          3
```

#16 - 04/14/2016 01:45 PM - Peter Grandi

I have written a feature request for 3 instead of 2 modes of "resumption" here, to include a mode where block presence is checked at the moment of upload, instead of assumed from a previous upload:

https://dev.arvados.org/issues/8993

#17 - 07/15/2016 09:23 PM - Tom Morris

Do the new feature requests stories capture everything from here? ie can this issue be safely closed?

#18 - 03/15/2017 03:12 PM - Tom Clegg

- Priority changed from Urgent to Normal

#19 - 05/30/2019 05:54 PM - Tom Morris

- Related to Feature #8993: arv-put: options for 3 modes of "resumption" added

#20 - 01/18/2020 01:10 AM - Peter Amstutz

- Status changed from New to Closed

Files

160329_arvDiskFree.png	37.6 KB	04/05/2016	Peter Grandi
keep_block_to_file.py	941 Bytes	04/10/2016	Peter Amstutz
datamanager	8.41 MB	04/10/2016	Peter Amstutz
160404_arvDiskFreeNotes.png	50.2 KB	04/11/2016	Peter Grandi
2016-04-11T13_34_34Z_filescount.txt	12.5 KB	04/11/2016	Peter Grandi