Arvados - Story #9162

[Block Manager] Trashes unreferenced blocks (with mtime > TTL if desired) and overreplicated block

05/10/2016 05:18 PM - Brett Smith

Status:	Resolved	**Start date:**	05/16/2016
Priority:	Normal	**Due date:**	
Assigned To:	Tom Clegg	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	2016-06-08 sprint		

Description

Functional requirements:

- Write a new component Block Manager. It sends trash lists following the same rules that Data Manager does today. Move code from Data Manager as appropriate.
- When sending trash lists to Keepstores, Block Manager sends lists to correct overreplication.
- Of all the keepstores that currently have the block, the block should be trashed from the server(s) that are lowest in the current rendezvous hash order. e.g., if a block with desired replication 2 has rendezvous hash order D, B, C, A, and it currently lives on D, C, and A, it should be trashed from A.

Subtasks:

Task # 9206: Build data-block map	**Resolved**
Task # 9209: Recommend trash/pull operations for a block	**Resolved**
Task # 9210: Buffer trash/pull operations and forward to keepstores	**Resolved**
Task # 9235: Accept keep services list in config	**Resolved**
Task # 9237: run as a service, wake up on timer and SIGUSR1	**Resolved**
Task # 9229: docs and lint	**Resolved**
Task # 9300: Build & publish package	**Resolved**
Task # 9211: Review 9162-keep-balance	**Resolved**

Related issues:

Related to Arvados - Feature #6611: [Data Manager] Act on pull lists for unde...	**Closed**	**07/14/2015**
Related to Arvados - Story #8201: [Data Manager] Reports underreplicated bloc...	**Closed**	**01/12/2016**

Associated revisions

Revision c900f416 - 05/31/2016 08:23 PM - Tom Clegg

Merge branch '9162-keep-balance'

closes #9162

History

#1 - 05/10/2016 05:20 PM - Brett Smith

- Description updated

#2 - 05/10/2016 05:20 PM - Brett Smith

- Target version set to Arvados Future Sprints

#3 - 05/10/2016 05:36 PM - Brett Smith

- Subject changed from [Data Manager] Deletes overreplicated blocks to [Data Manager] Trashes overreplicated blocks

#4 - 05/10/2016 06:33 PM - Brett Smith

- Description updated

#5 - 05/10/2016 06:44 PM - Brett Smith

Tom's comments:

- It's Block Manager, not Data Manager. All it cares about is making sure blocks live in the right place(s).
- It doesn't report statistics on collections or other tangential jobs like that.

- It doesn't have to hold all the collections in RAM. It just has to iterate them to determine the desired replication level for each block.
- If reporting additional information is the only reason we're building a data structure, I'm inclined to ditch the data structure and have that reporting in a separate component.
- There are functions missing from the SDK, judging by our other tools like block-check. Give me all the blocks on one Keepstore. Give me all the blocks on all Keepstores. Give it to me in a convenient interface, like a reader.
- Data Manager was our first Go client. We know a lot more now.

We're writing a new component blockmanager, codename "Keep It Beautiful." We will pull over code from Data Manager as appropriate.

#6 - 05/11/2016 05:13 PM - Brett Smith

- Subject changed from [Data Manager] Trashes overreplicated blocks to [Block Manager] Trashes unreferenced blocks (with mtime > TTL if desired) and overreplicated block

- Description updated

- Assigned To set to Tom Clegg

- Target version changed from Arvados Future Sprints to 2016-05-25 sprint

#7 - 05/11/2016 07:10 PM - Tom Clegg

- Story points set to 3.0

"keep-rebalance"?

#9 - 05/16/2016 02:18 PM - Tom Clegg

- Status changed from New to In Progress

#10 - 05/20/2016 03:19 PM - Tom Clegg

- File keep-balance added

- File example-config.json added

- Status changed from In Progress to New

(adding a pre-release binary & example config file)

#11 - 05/20/2016 09:29 PM - Tom Clegg

Summary of 9162-keep-balance:

Functional

Three main features:

1. Delete the worst-placed replicas of overreplicated blocks ("worst-placed" according to rendezvous)
2. Make more replicas of underreplicated blocks, and blocks that aren't in the best rendezvous places
3. Report some statistics

Operational

Configuration

- Config comes from a JSON file. "keep-balance -config /path/to/config.json"
- Config can provide either a list of Keep services to balance, or (similar to datamanager) a list of service types to look for at arvados.v1.keep_services.list.
- The list of Keep services can be given in a separate JSON file, in the same form produced by "arv keep_service list". "keep-balance -config config.json -config.KeepServiceList svcs.json"

Runtime options

- Default behavior is run;sleep;repeat -- use "-once" flag to just run once and exit.
- Config file specifies RunPeriod. RunPeriod="10m" means start every 10 minutes (as opposed to sleeping 10 minutes between runs).
- If SIGUSR1 is received while sleeping, the next run starts right away (and the timer gets reset). If SIGUSR1 is received while balancing is already in progress, it is ignored.

Limitations / caveats

Some configurations will cause keep-balance to waste disk space and report confusing statistics.

- If keep is relying on storage volumes for replication (e.g., blob storage) then replication won't be counted properly: currently, keep-balance has no way of knowing what the underlying replication level is.

- If multiple keepstores have access to the same underlying storage volumes, blocks will appear to be overreplicated, but the (seemingly) excess copies won't be deleted: to avoid deleting more than we want to, we assume identical modtimes always mean multiple keepstore servers are sharing a storage volume.

In a supported "simple" setup:

- Each storage volume represents replication=1.
- No storage volume is ever visible to more than one keepstore server at a time.

In a non-simple setup, keep-balance is safe in that it won't delete too much: it may delete too little, and it may waste even more space by increasing replication beyond the current and desired levels.

Even in a simple setup, some cases are not yet handled optimally.

- If a (non-garbage) block has replicas on multiple volumes on a single server, and that server is one of the best rendezvous positions, the extra copies won't be deleted.
- If a block is stored with the same modtime at best- and non-best server positions, the surplus won't be deleted. These incidents aren't reported in the statistics.

Read-only servers

Read-only servers (according to the "read_only" flag in the API response) are treated as temporarily read-only, as opposed to being in a "please drain" state.

If the best pull target for an underreplicated block is on a read-only server, the next-best target will be used instead.

Tracking actual progress

There is no mechanism for tracking whether pull/trash requests are being executed successfully. For example, if a keep service has no writable/non-full volumes, keep-balance will just keep sending pull requests. It will never notice that the pull requests are futile and fall back on a next-best service.

Downtime

If any of the keep servers are down or return errors when fetching indexes, keep-balance just gives up. This means that from the time a keep server goes down to the time it gets (manually?) de-listed in the Arvados database, all rebalancing and garbage collection stops.

Misc

The "with mtime > TTL if desired" feature isn't implemented. To be effective, this would require changes to keepstore: as is, keepstore would ignore such trash requests anyway.

#12 - 05/24/2016 03:11 PM - Radhika Chippada

Review comments (partial):

main.go

- In the following, shouldn’t json.Marshal(config) be executed whether or not debug flag is set (outside of the if debugFlag?

```
    if *debugFlag {
        debugf = log.Printf
        if j, err := json.Marshal(config); err != nil {
            log.Fatal(err)
        } else {
            log.Printf("config is %s", j)
        }
    }
```

- mustReadJSON - can we call it “load config from JSON” or something?

- mustReadJSON - what happens if I have a valid json, but incorrect values such as “ keep1 “ or “keep 1” are provided?

- It seems like “keep-balance.json” needs more detailed help message. May be mention “see example.json” for detail? This example.json itself can use some explanation that serviceListPath and RunPeriod can also be provided as separate arguments? Also, can we add one more argument for KeepServiceTypes; it appears this can only be provided through config file at this time.

```
    configPath := flag.String("config",
        os.Getenv("HOME")+"/.config/arvados/keep-balance.json",
        "`path` of json configuration file")
```

- Does runForever respond to SIGQUIT etc? Or, more precisely, can I kill the process gracefully?

```
        select {
        case <-ticker.C:
        case <-sigUSR1:
```

- The code might read better if "type duration time.Duration" and associated code is placed above / near the main method where it is used

  time_me

- Can we add "took" to the print statement in the func similar to "start"?

```
 - logger.Printf("%s: %v", label, time.Since(t0))
 + logger.Printf("%s took: %v", label, time.Since(t0))
```

  Tests

- Can't run tests

```
./collection.go:34: scanner.Buffer undefined (type *bufio.Scanner has no field or method Buffer)
```

- Depending on the answer to my question about runForever above, we may be able to add another integration test that invokes runForever (may be need a done ticker …)?

  keep_balance -> keep_service.go

- It would be really helpful if you can throw in some new lines and a couple comments to the (srv *KeepService) put func

  block_state.go

- "Replica is a file on disk (or equivalent)" — how about "Replica is a file in storage"
- "the collections we know about) and the replicas found on disk" — should this say "found on storage" instead of disk?

  sdk/go/arvados:

- In DoAndDecode, can the check "if dst == nil" be done at the start of the method instead of after doing a Do?
- KeepServiceIndexEntry is a bit confusing. BlockIndexEntry might be clearer? Also the comment needs updating.
- "Balancing a proxy service (%s) must be a config error" is a bit confusing. How about something like "a proxy service cannot be balaced …"

#13 - 05/24/2016 08:43 PM - Tom Clegg

Radhika Chippada wrote:

> - In the following, shouldn't json.Marshal(config) be executed whether or not debug flag is set (outside of the if debugFlag?

I don't think so: if we're not going to display the config then there's no need to stringify it.

> - mustReadJSON - can we call it "load config from JSON" or something?

Well, it isn't really config-specific... "must" is a Go convention for things that exit on failure instead of returning an error (like regexp.MustCompile).

> - mustReadJSON - what happens if I have a valid json, but incorrect values such as " keep1 " or "keep 1" are provided?

In that case the failure will happen later when we try to connect to the servers...

> - It seems like “keep-balance.json” needs more detailed help message. May be mention “see example.json” for detail?

Agreed. I'm not sure of the best way to present this. Perhaps a link to a wiki page would work? That way we can provide a link to the example config in the source tree, as well as a less terse explanation of how the configs work.

> This example.json itself can use some explanation that serviceListPath and RunPeriod can also be provided as separate arguments? Also, can we add one more argument for KeepServiceTypes; it appears this can only be provided through config file at this time.

I don't want to offer two ways to set every config. The reasoning behind the "-config.KeepServiceList" exception is that it lets you do (either manually or in a script) "arv keep_service list > list.json" and then use that list in its existing format, without having to fiddle around with jq or an editor to merge it into your config file. RunPeriod, credentials, and the list of service types can only be given via config file.

> - Does runForever respond to SIGQUIT etc? Or, more precisely, can I kill the process gracefully?

There's no special handling for other signals, so SIGINT and SIGTERM (for example) will cause it to stop & exit immediately.

> [...]
>
> - The code might read better if “type duration time.Duration” and associated code is placed above / near the main method where it is used

Moved this to duration.go (it isn't especially helpful for understanding how the rest of main.go works, right?)

> - Can we add “took” to the print statement in the func similar to “start”?

Done

> [...]
>
> Tests
>
> - Can’t run tests

I'm guessing this is because you have Go < 1.6: turns out the Buffer() function was only added in 1.6. I've updated the build/test scripts to install/require 1.6.2, and updated Hacking prerequisites.

> [...]
>
> - Depending on the answer to my question about runForever above, we may be able to add another integration test that invokes runForever (may be need a done ticker …)?

Added a RunForever test.

> keep_balance -> keep_service.go
>
> - It would be really helpful if you can throw in some new lines and a couple comments to the (srv *KeepService) put func

Done

> block_state.go
>
> - “Replica is a file on disk (or equivalent)” — how about “Replica is a file in storage”

Well, azure/s3 have blobs/objects instead of files, but point taken... I've expanded on "or equivalent" here.

> - “the collections we know about) and the replicas found on disk” — should this say “found on storage” instead of disk?

Changed to "actually stored".

> sdk/go/arvados:
>
> - In DoAndDecode, can the check “if dst == nil” be done at the start of the method instead of after doing a Do?

Not really... If dst is nil, the caller doesn't care what the response body says, but we still want to do the request and check the status code so we know whether it worked.

> - KeepServiceIndexEntry is a bit confusing. BlockIndexEntry might be clearer? Also the comment needs updating.

Updated the comment. I think "KeepServiceIndexEntry" makes more sense in that it corresponds to the Keep service's "index" API specifically. E.g., if we change the keepstore index format to return a volume ID with each block, we'd certainly add that here.

> - “Balancing a proxy service (%s) must be a config error” is a bit confusing. How about something like “a proxy service cannot be balaced …”

Changed to: "config error: zzzzz-bi6l4-xyajj22w3ipge3o (keep99.zzzzz.arvadosapi.com:443, proxy): proxy servers cannot be balanced"

I've made a couple of other changes:

- Address a race condition where keep services are still processing old trash lists while we're making decisions
- Add replication level histogram (using Josh's code from #9232)
- Adjust dump format for easier grep (feedback from Nico)

Now at ac9cacc

#14 - 05/25/2016 01:59 AM - Radhika Chippada

A few more comments (still haven't completed going through balance*test.go)

balance.go

- Can you please add block level comments (for each go routine etc) in GetCurrentState. It would help quickly understand what each block is doing.

- Why it is 1000 here “collQ := make(chan arvados.Collection, 1000)” ? I see that EachCollection uses 1000 limit, but if I am reading it correctly, EachCollection will try to get all collections from server (items_available) which could be more than 1000.

- Why is it 16 here “todo := make(chan balanceTask, 16)“?

  sdk/og/arvados/collection.go

- It appears that this sdk is doing a lot that may only be applicable to balancer (data manager)? Should EachCollection be in the balance package itself or do you see other clients needing similar functionality? I am asking because we felt the original “data manager” code is hard to grasp and I just wanted to make sure we talk about the scope of the sdk in this context.

  Regarding some of the responses from note 13

- "example-config.json -- I'm not sure of the best way to present this. Perhaps a link to a wiki page would work?" -- If example.json is not actually used in tests (I am yet to make a thorough reading of the tests), I think we can replace as "example-config-json" and add all the detailed description at the top of the file followed by the json. Alternatively, you can add a doc.go in this directory?

- "Moved this to duration.go (it isn't especially helpful for understanding how the rest of main.go works, right?)" -- This is much clearer

#15 - 05/25/2016 02:36 PM - Tom Clegg

Radhika Chippada wrote:

> - Can you please add block level comments (for each go routine etc) in GetCurrentState. It would help quickly understand what each block is doing.

Done

> - Why it is 1000 here “collQ := make(chan arvados.Collection, 1000)” ? I see that EachCollection uses 1000 limit, but if I am reading it correctly, EachCollection will try to get all collections from server (items_available) which could be more than 1000.

Added a comment here. (1000 is just a buffer between fetch and process -- it doesn't limit the total number of collections to process.)

> - Why is it 16 here “todo := make(chan balanceTask, 16)“?

Another arbitrary size. I've changed it to nWorkers, but it's not critical.

> - It appears that this sdk is doing a lot that may only be applicable to balancer (data manager)? Should EachCollection be in the balance package itself or do you see other clients needing similar functionality? I am asking because we felt the original “data manager” code is hard to grasp and I just wanted to make sure we talk about the scope of the sdk in this context.

I think you're right. Initially I hoped it would be a generic iterator, but the sanity check, sorting, and choice of "select" fields are not really generic. Moved to keep-balance.

> If example.json is not actually used in tests (I am yet to make a thorough reading of the tests), I think we can replace as "example-config-json" and add all the detailed description at the top of the file followed by the json. Alternatively, you can add a doc.go in this directory?

No, the example file isn't used in tests. I've moved it into usage.go such that "keep-balance -h" shows it (along with some supporting comments).

Now at 595f455

#16 - 05/25/2016 02:46 PM - Radhika Chippada

A few more comments (still at ac9cacc9):

- Can you please add a test with json file for config?
- Can duration.go be moved into sdk to allow future reusability? also, do we need the check string(data[1 : len(data)-1]) here ? Can it be just string(data)?
- In balance_run_test — In TestDryRun, please add a few more assertions such as under replicated etc?
- In balance_test — is serviceByUUID used?
- In collection.go: in this (if last.ModifiedAt != nil && *last.ModifiedAt == *coll.ModifiedAt && last.UUID >= coll.UUID) sh we just check last.UUID and coll.UUID are equal instead?
- It seems like we do not have such a test: countCollections returns 10, but the callCount is < 10 (something like the race condition you addressed where a separate balancer doing some trashing while we are preparing a change set). If so, can we add such a test?

#17 - 05/25/2016 06:02 PM - Tom Clegg

- Status changed from New to In Progress

#18 - 05/25/2016 06:59 PM - Brett Smith

- Target version changed from 2016-05-25 sprint to 2016-06-08 sprint

#19 - 05/25/2016 07:06 PM - Tom Clegg

- Story points changed from 3.0 to 0.5

#20 - 05/26/2016 03:07 AM - Tom Clegg

Radhika Chippada wrote:

> - Can you please add a test with json file for config?

Done

> - Can duration.go be moved into sdk to allow future reusability?

Done

> also, do we need the check string(data[1 : len(data)-1]) here ? Can it be just string(data)?

This skips the quotation marks. data is "\"123s\"" but ParseDuration only understands "123s".

> - In balance_run_test — In TestDryRun, please add a few more assertions such as under replicated etc?

Added checks for overrep!=0 and underrep!=0 to help convince us that "dry run" is the real reason we didn't commit any changes, as opposed to "didn't come up with any changes".

> - In balance_test — is serviceByUUID used?

No, you're right, that's dead code. Removed.

- In collection.go: in this (if last.ModifiedAt != nil && *last.ModifiedAt == *coll.ModifiedAt && last.UUID >= coll.UUID) sh we just check last.UUID and coll.UUID are equal instead?

Not quite.

Say we get a page ending with collections A1, C1, E1 with identical mtimes (1), but different UUIDs (A, C, E). The next query will use filters "mtime >= 1" and "uuid != E". The next page could start off with A1, C1, G1, B2, D2. We have to skip A1 and C1 (to avoid double-counting them and thwarting our "total number of collections received" sanity check later) but of course we can't skip G1 or B2 because we haven't processed them yet.

That condition matches A1 and C1, but not G1 or B2.

It also matches E1, although that should be unnecessary -- it just means we avoid double-counting E1 even if the API ignores our "uuid != E1" filter.

- It seems like we do not have such a test: countCollections returns 10, but the callCount is < 10 (something like the race condition you addressed where a separate balancer doing some trashing while we are preparing a change set). If so, can we add such a test?

Done (good thing -- also found & fixed a couple of bugs as a result)

Now at 8b97af2

#21 - 05/29/2016 10:01 PM - Radhika Chippada

usage.go

- “keep-balance copies blocks to better positions so clients find them” — can you please add “makes sure desired number of replications …” here
- “service mode” : this phrase is a bit confusing because it is a derived concept, not something explicitly configured. Can you please consider something like “Run as a service or once”? Also this section can say RunPeriod must be specified unless using -once option. If possible please change all “service mode” references to say “as a service”
- “By default, keep-service computes and reports changes but does not implement them …” — typo here “keep-service”. Can you also say “dry run” in this context. Also, you might want to consider section title “Committing changes” here.
- It feels like RunPeriod should be part of RunOptions and should also be specifiable similar to Once and CommitPulls etc. Removing it from Config struct and making it part of RunOptions should also make reading and understanding the whole usage and config json etc a bit easier by allowing all the “run options” into one section. Currently, it still feels a bit heavy and scattered

main.go

- It doesn’t appear that we are actually using RunOptions.Once
- You mention “dry run” in a comment but do not clarify what it is (no CommitPulls and CommitTrash)
- “Config is loaded from a JSON config file” — please expand this to say api config from file and keep server config from file or from command line arg
- "`path` of json configuration file" — can you please add “use `keep-balance -help` for further detail”?
- Instead of “waking up”, we might want to say “Starting next balance run”
- It would be nice if you can add a test with config params (may be move code from move into a doMain …)
- It would be nice to also have a test with exampleConfigFileWithKeepSerivces also
- Also, it would be nice to create config files in the test and test with mustReadJSON

balance.go

- should “if len(config.KeepServiceList.Items) > 0 && config.KeepServiceTypes != nil” be in main.go and do a log.Fatalf? In this case, there won’t be any benefit in sleeping and retrying?

block_state.go

- “ (bsm *BlockStateMap) get “ changes the state on bsm. I think it would help to clarify in it’s comment that caller needs to lock using the mutex (both the current usages are already correctly do so).

Misc

- I think time_me can be reusable and belongs in the sdk (in the util package?). Also duration.go does not actually seem to belong in “arvados” sdk package and may belong in the “util” package?
- go vet error - can we instead use this in ClearTrashList?

```
        for _, srv := range bal.KeepServices {
-               srv.ChangeSet = ChangeSet{}
+               srv.ChangeSet.Pulls = make([]Pull, 0)
+               srv.ChangeSet.Trashes = make([]Trash, 0)
        }
```

- 2 test issues

#22 - 05/30/2016 07:52 PM - Tom Clegg

Radhika Chippada wrote:

> usage.go
>
> - “keep-balance copies blocks to better positions so clients find them” — can you please add “makes sure desired number of replications …” here
> - “service mode” : this phrase is a bit confusing because it is a derived concept, not something explicitly configured. Can you please consider something like “Run as a service or once”? Also this section can say RunPeriod must be specified unless using -once option. If possible please change all “service mode” references to say “as a service”
> - “By default, keep-service computes and reports changes but does not implement them …” — typo here “keep-service”. Can you also say “dry run” in this context. Also, you might want to consider section title “Committing changes” here.

Updated these comments/docs.

> - It feels like RunPeriod should be part of RunOptions and should also be specifiable similar to Once and CommitPulls etc. Removing it from Config struct and making it part of RunOptions should also make reading and understanding the whole usage and config json etc a bit easier by allowing all the “run options” into one section. Currently, it still feels a bit heavy and scattered

I'd rather move flags to the config file than the other way around (ops are asking for config files).

"once" and "commit/no-commit" aren't in the config because they seem like they're only useful in an interactive setting, where "use my config file, but help me debug/preview what's going on" seems like the most common use case.

> main.go
>
> - It doesn’t appear that we are actually using RunOptions.Once

Indeed. Fixed by using runOptions.Once instead of local var "once".

> - You mention “dry run” in a comment but do not clarify what it is (no CommitPulls and CommitTrash)

Fixed

> - “Config is loaded from a JSON config file” — please expand this to say api config from file and keep server config from file or from command line arg

Added a reference to usage() here. (Seems to me if we're going to expand on the discussion of how to configure, we should do it in the -help message where the person configuring it can see it...)

> - "`path` of json configuration file" — can you please add “use `keep-balance -help` for further detail”?

AFAIK the only place this string gets displayed is in the middle of the -help message -- wouldn't this be redundant?

> - Instead of “waking up”, we might want to say “Starting next balance run”

Changed to "starting next run"

> - It would be nice if you can add a test with config params (may be move code from move into a doMain …)

You mean test the command line flag parsing? (I'm not sold on this -- it seems like it would mostly test whether the flag package works.)

> - It would be nice to also have a test with exampleConfigFileWithKeepSerivces also

Added

> - Also, it would be nice to create config files in the test and test with mustReadJSON

I'm not convinced. That would mostly test stdlib?

> balance.go
>
> - should "if len(config.KeepServiceList.Items) > 0 && config.KeepServiceTypes != nil" be in main.go and do a log.Fatalf? In this case, there won't be any benefit in sleeping and retrying?

Indeed. Moved that check to a CheckConfig() function so main() can call that before starting the timer loop.

> block_state.go
>
> - " (bsm *BlockStateMap) get " changes the state on bsm. I think it would help to clarify in it's comment that caller needs to lock using the mutex (both the current usages are already correctly do so).

Added a comment.

> Misc
>
> - I think time_me can be reusable and belongs in the sdk (in the util package?).

Meh, it's 3 LOC -- it's OK where it is, IMO.

> Also duration.go does not actually seem to belong in "arvados" sdk package and may belong in the "util" package?

If we want Arvados components to read/write durations as "600s" in JSON docs, an arvados.Duration type seems reasonable.

Also:

- no "util" or "misc" packages, please
- duration.go moved to the SDK after you asked for it in note-16...!

> - go vet error - can we instead use this in ClearTrashList?

Changed it to a *ChangeSet so callers can clear it by assigning a zero value, instead of by clearing only the public fields.

> - 2 test issues

Any further clues? :)

Now at 335f749

#23 - 05/31/2016 01:21 AM - Radhika Chippada

- The usage and other comments are much easier to read and follow now. Thanks.

- Please run go fmt (2 files)

- Failing tests: Here is the error when I run keep-balance using run-tests (after a fresh install)

```
           ********** Running services/keep-balance tests **********

2016/05/30 21:16:19 authSettings: map[ARVADOS_API_HOST:0.0.0.0:38472 ARVADOS_API_HOST_INSECURE:true ARVADO
S_API_TOKEN:4axaw8zxe0qm22wa6urpp5nskcne8z88cvbupv653y1njyi05h]
Traceback (most recent call last):
  File "run_test_server.py", line 723, in <module>
    run_keep(enforce_permissions=args.keep_enforce_permissions, num_servers=args.num_keep_servers)
  File "run_test_server.py", line 409, in run_keep
    port = _start_keep(d, keep_args)
  File "run_test_server.py", line 369, in _start_keep
    keep_cmd, stdin=open('/dev/null'), stdout=logf, stderr=logf, close_fds=True)
  File "/usr/lib/python2.7/subprocess.py", line 710, in __init__
    errread, errwrite)
  File "/usr/lib/python2.7/subprocess.py", line 1335, in _execute_child
    raise child_exception
OSError: [Errno 2] No such file or directory

----------------------------------------------------------------------
FAIL: integration_test.go:27: integrationSuite.SetUpSuite

integration_test.go:43:
    s.putReplicas(c, "foo", 4)
integration_test.go:50:
    c.Assert(err, check.IsNil)
... value *errors.errorString = &errors.errorString{s:"Could not write sufficient replicas"} ("Could not w
rite sufficient replicas")

OOPS: 19 passed, 1 FAILED, 1 MISSED
--- FAIL: Test (7.92s)
FAIL
coverage: 78.1% of statements
FAIL    git.curoverse.com/arvados.git/services/keep-balance    7.942s

     ********** !!!!!! services/keep-balance tests FAILED !!!!!! **********
```

- Please change the category on this ticket to [keep] or [keep-balance]

#24 - 05/31/2016 02:01 PM - Tom Clegg

Radhika Chippada wrote:

> - Please run go fmt (2 files)

Done

> - Failing tests: Here is the error when I run keep-balance using run-tests (after a fresh install)

Aha. Couldn't run keepstore, because keepstore install failed, because I hadn't merged master to sync with the latest azure-sdk-for-go. Should be fixed now.

#25 - 05/31/2016 03:43 PM - Radhika Chippada

Tests passing now. LGTM

#26 - 05/31/2016 08:30 PM - Tom Clegg

- Status changed from In Progress to Resolved

- % Done changed from 88 to 100

Applied in changeset arvados|commit:c900f416c36cd74675c5bf4c33ad1dbe5d1e78fa.

Files

keep-balance	8.45 MB	05/20/2016	Tom Clegg
example-config.json	498 Bytes	05/20/2016	Tom Clegg