Arvados - Story #10477

[keepstore] switch s3 driver from goamz to a more actively maintained client library

11/08/2016 12:20 AM - Tom Morris

Status:	Resolved	**Start date:**	11/08/2016
Priority:	Normal	**Due date:**	
Assigned To:	Ward Vandewege	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	2020-08-12 Sprint		

Description

Choices include

- official AWS Go SDK, https://github.com/aws/aws-sdk-go. Kubernetes made the switch in May 2015. The Kubernetes PR provides insight into what needs to be changed and possible approaches to migrating the code.
- new/upcoming v2 of official AWS Go SDK, https://github.com/aws/aws-sdk-go-v2
- minio-go, client library from S3-compatible cloud storage system: godoc / API docs ... This might have better compatibility with non-Amazon S3 services like Google cloud storage.

While we will probably choose the official client library when we add Go code that uses EC2, that doesn't necessarily qualify as a reason to choose the official library for storage as well.

Subtasks:

Task # 16609: Review 10477-upgrade-aws-s3-driver	**Resolved**

Related issues:

Related to Arvados Epics - Story #16516: Run Keepstore on local compute nodes	**In Progress**	**10/01/2021**	**11/30/2021**
Related to Arvados - Feature #16513: Get reference Keep performance numbers f...	**Resolved**	**06/15/2020**	

Associated revisions

Revision 9c03beda - 08/04/2020 07:45 PM - Ward Vandewege

Merge branch 'master' into 10477-upgrade-aws-s3-driver

refs #10477

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <ward@curii.com>

Revision a37291f5 - 08/04/2020 07:45 PM - Ward Vandewege

Merge branch '10477-upgrade-aws-s3-driver'

closes #10477

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <ward@curii.com>

History

#1 - 11/11/2016 11:29 PM - Tom Morris

- Description updated

#2 - 03/01/2017 05:22 PM - Tom Morris

- Target version set to Arvados Future Sprints

#3 - 03/21/2017 06:39 PM - Tom Morris

- Story points set to 2.0

#4 - 08/01/2018 06:34 PM - Tom Clegg

- Subject changed from Switch from goamz to aws-sdk-go to [keepstore] switch from goamz to a more actively maintained client library

- Description updated

#5 - 04/30/2020 02:26 PM - Ward Vandewege

- Blocks Feature #16312: Support encrypted S3 buckets added

#6 - 05/14/2020 04:26 PM - Tom Clegg

- Blocks deleted (Feature #16312: Support encrypted S3 buckets)

#7 - 06/16/2020 01:52 PM - Tom Clegg

- Subject changed from [keepstore] switch from goamz to a more actively maintained client library to [keepstore] switch s3 driver from goamz to a more actively maintained client library

#8 - 06/16/2020 01:52 PM - Tom Clegg

- Related to Story #16516: Run Keepstore on local compute nodes added

#9 - 06/30/2020 08:43 PM - Ward Vandewege

- Related to Feature #16513: Get reference Keep performance numbers for Keep-on-S3 added

#10 - 07/15/2020 01:15 PM - Ward Vandewege

- Target version changed from Arvados Future Sprints to 2020-07-15

- Assigned To set to Ward Vandewege

- Status changed from New to In Progress

#11 - 07/15/2020 03:50 PM - Ward Vandewege

- Target version changed from 2020-07-15 to 2020-08-12 Sprint

#12 - 07/22/2020 11:13 AM - Ward Vandewege

First version ready for review at ea57684c255434bcd25ec150a3979ce783a2183c on branch 10477-upgrade-aws-s3-driver. Tests at https://ci.arvados.org/view/Developer/job/developer-run-tests/1967/

#13 - 07/24/2020 05:57 PM - Ward Vandewege

New revision ready at 8f3b2dedef2677654197e9838939d9abe7cc3791 on branch 10477-upgrade-aws-s3-driver. This one has faster upload performance, because we don't use sha-256 anymore.

#14 - 07/28/2020 07:21 PM - Tom Clegg

The benchmark results seem to me a bit out of place in the install guide. I wonder if it would be better to move these details to a separate page (maybe even wiki), and pare back the install guide to something along the lines of "driver A has had more production use, but driver B can improve read performance by 50-100%, see wiki for details"?

The name "AlternateDriver" seems a bit sketchy wrt the "never reuse config flag to mean something else" story. Perhaps something similar to the "alternate controller code path" flag, like "ExperimentalDriver10477: true"? Then, when this becomes the default, we'd drop this flag and add a flag like "UseOldGoamzDriver: true".

Uploader concurrency 5, uploader partsize 5 MiB, etc. should be defined as consts.

It seems like PutReader is only used to write empty objects for "trash" and "recent" markers, and it has a 3rd copy of the s3manager.NewUploaderWithClient() code that doesn't propagate context. Perhaps this could be cleaned up by replacing PutReader with

```
func (v *S3AWSVolume) writeObject(ctx context.Context, name string, r io.Reader) error { ... }
```

(If len(name)==32 then writeObject can re-encode it as base64 to set ContentMD5.)

Then, WriteBlock() would reduce to something like

```
r := NewCountingReader(...)
err := v.writeObject(ctx, loc, r)
if err != nil { return err }
return v.writeObject(ctx, "recent/"+loc, nil)
```

This comment (copied from old driver which had r=nil) needs to be updated to match the code (which presumably changed because r=nil didn't work). Aside: does bytes.NewReader(nil) work just as well?

```
+       if length == 0 {
+               // aws-sdk-go will only send Content-Length: 0 when reader
+               // is nil due to net.http.Request.ContentLength
+               // behavior.  Otherwise, Content-Length header is
+               // omitted which will cause some S3 services
+               // (including AWS and Ceph RadosGW) to fail to create
```

```
+               // empty objects.
+               r = bytes.NewReader([]byte{})
```

I'm not sure how to manage the copy-paste aspect here. A lot of stuff is copied from the goamz-based driver with slight modifications, and finding the differences is a bit of an exercise. I suppose the implication is that we'll be deleting the goamz driver very soon, because until then we'll be maintaining 2 divergent copies of a lot of stuff.

(*S3AWSVolume)check() should error out if V2Signature is true, since that's not supported (maybe also worth a mention in docs/config example).

This doesn't look right -- surely a base64-encoded md5 will never be "d41d8cd98f00b204e9800998ecf8427e"? -- is this an indication the special case isn't even needed?

```
+       contentMD5 = base64.StdEncoding.EncodeToString(md5)
+       // See if this is the empty block
+       if contentMD5 != "d41d8cd98f00b204e9800998ecf8427e" {
+               uploadInput.ContentMD5 = &contentMD5
+       }
```

Remove commented-out code:

```
+       //var contentMD5, contentSHA256 string
```

These look like they should be Debugf:

```
+               v.logger.Warnf("EmptyTrash: looking for trash marker %s with last modified date %s", *trash.Ke
y, *trash.LastModified)
...
+               v.logger.Infof("HEEEEEEE trashT key: %s, type: %T val: %s, startT is %s", *trash.Key, trashT,
trashT, startT)
...
+                       v.logger.Infof("HERE! trashT for %s is smaller than blobtrashlifetime: %s < %s", *tras
h.Key, startT.Sub(trashT), v.cluster.Collections.BlobTrashLifetime.Duration())
...
+               v.logger.Infof("HERE! trash.Key %s should have been deleted", *trash.Key)
...
+               v.logger.Infof("HERE! recent/%s should have been deleted", loc)
```

Various fmt.Printf() in tests would be better as c.Logf():

```
+               fmt.Printf("USING TIMESTAMP %s to write key %s", t, key)
```

The arg to Unmarshal here can be v rather than &v since v is already a pointer:

```
+       v := &S3Volume{cluster: cluster, volume: volume, metrics: metrics}
+       err := json.Unmarshal(volume.DriverParameters, &v)
```

@ 8f3b2dede, 3 tests are failing. Haven't investigated further.

#15 - 07/29/2020 03:59 PM - Ward Vandewege

Tom Clegg wrote:

> The benchmark results seem to me a bit out of place in the install guide. I wonder if it would be better to move these details to a separate page (maybe even wiki), and pare back the install guide to something along the lines of "driver A has had more production use, but driver B can improve read performance by 50-100%, see wiki for details"?

Sure, done.

> The name "AlternateDriver" seems a bit sketchy wrt the "never reuse config flag to mean something else" story. Perhaps something similar to the "alternate controller code path" flag, like "ExperimentalDriver10477: true"? Then, when this becomes the default, we'd drop this flag and add a flag like "UseOldGoamzDriver: true".

I've changed the flag to "UseAWSS3v2Driver". Including the ticket number would make this too unwieldy imo. The risk of future reuse of that flag seems low.

> Uploader concurrency 5, uploader partsize 5 MiB, etc. should be defined as consts.

Sure, done.

> It seems like PutReader is only used to write empty objects for "trash" and "recent" markers, and it has a 3rd copy of the s3manager.NewUploaderWithClient() code that doesn't propagate context. Perhaps this could be cleaned up by replacing PutReader with

> [...]
>
> (If len(name)==32 then writeObject can re-encode it as base64 to set ContentMD5.)
>
> Then, WriteBlock() would reduce to something like

Yeah, it is much better refactored like that, thanks.

> This comment (copied from old driver which had r=nil) needs to be updated to match the code (which presumably changed because r=nil didn't work).

r=nil still does not work; I've updated the comment along those lines. Could be improved still.

> Aside: does bytes.NewReader(nil) work just as well?

Yes, bytes.NewReader(nil) does work as well; I had used the other form since that's what they did in the AWS driver doc. Is bytes.NewReader(nil) preferred?

> I'm not sure how to manage the copy-paste aspect here. A lot of stuff is copied from the goamz-based driver with slight modifications, and finding the differences is a bit of an exercise.
> I suppose the implication is that we'll be deleting the goamz driver very soon, because until then we'll be maintaining 2 divergent copies of a lot of stuff.

Yeah, that's what I was thinking. We have pretty good test coverage, and Kubernetes switched cold-turkey a few years ago so it can be done. I propose to make the new driver optional in 2.1, default in 2.2, and the old one removed in 2.3.

> (*S3AWSVolume)check() should error out if V2Signature is true, since that's not supported (maybe also worth a mention in docs/config example).

Yes good catch, I've added an error if V2Signature is true. The docs already mention that V2Signature is not supported, at the bottom of the "Configure S3 object storage" page where this driver is introduced.

> This doesn't look right -- surely a base64-encoded md5 will never be "d41d8cd98f00b204e9800998ecf8427e"? -- is this an indication the special case isn't even needed?

You are right, I've run the tests and did some more real-life testing as well, the special case is not needed. I've removed it.

> Remove commented-out code:

Done.

> These look like they should be Debugf:

They shouldn't even be there, I've removed them (and some other leftovers).

> Various fmt.Printf() in tests would be better as c.Logf():

I removed those too, they didn't really serve a purpose anymore.

> The arg to Unmarshal here can be v rather than &v since v is already a pointer:

Fixed. Same thing in the s3_volume.go driver so I fixed that too.

> @ 8f3b2dede, 3 tests are failing. Haven't investigated further.

Hmm, all the Keepstore tests are passing for me locally. I've kicked off a jenkins test run at https://ci.arvados.org/view/Developer/job/developer-run-tests/1973/

Ready for another look at 95babd9e21eb871eed9535fad3d2af8ecdeb471d on branch 10477-upgrade-aws-s3-driver

#16 - 07/31/2020 03:17 PM - Tom Clegg

> changed the flag to "UseAWSS3v2Driver"

Yeah, that seems safer.

> r=nil still does not work; I've updated the comment along those lines. Could be improved still.

Makes much more sense with the new comment, thanks.

> Yes, bytes.NewReader(nil) does work as well; I had used the other form since that's what they did in the AWS driver doc. Is bytes.NewReader(nil) preferred?

One less memory allocation. Wouldn't make a huge difference here, it's just a good habit. Generally, anything that takes a slice/map (and isn't expected to write to it) can take a nil slice/map. len() and range work, map lookups return zero value / not found, etc.

> The arg to Unmarshal here can be v rather than &v since v is already a pointer:

> Fixed. Same thing in the s3_volume.go driver so I fixed that too.

Ah, nice. Copy&paste&review win. :D

> @ 8f3b2dede, 3 tests are failing. Haven't investigated further.

> Hmm, all the Keepstore tests are passing for me locally. I've kicked off a jenkins test run at https://ci.arvados.org/view/Developer/job/developer-run-tests/1973/

Ah. The tests that fail are all timestamp-related. Seems like a timezone thing -- try running tests with TZ=EDT. Could be a testing-only problem, but worth fixing either way.

```
FAIL: s3aws_volume_test.go:76: StubbedS3AWSSuite.TestGeneric

s3aws_volume_test.go:77:
    DoGenericVolumeTests(c, false, func(t TB, cluster *arvados.Cluster, volume arvados.Volume, logger logrus.F
ieldLogger, metrics *volumeMetricsVecs) TestableVolume {
        // Use a negative raceWindow so s3test's 1-second
        // timestamp precision doesn't confuse fixRace.
        return s.newTestableVolume(c, cluster, volume, metrics, -2*time.Second)
    })
volume_generic_test.go:438:
    t.Errorf("got %d for TestHash timestamp, expected %d <= t <= %d",
        mtime, minMtime, maxMtime)
... Error: got 1596128123118000000 for TestHash timestamp, expected 1596142523000000000 <= t <= 15961425240000
00000
```

"got 1596128123118000000" is 4h earlier than expected, and my TZ is UTC-4.

#17 - 08/03/2020 03:15 PM - Ward Vandewege

Tom Clegg wrote:

> Ah. The tests that fail are all timestamp-related. Seems like a timezone thing -- try running tests with TZ=EDT. Could be a testing-only problem, but worth fixing either way.
>
> [...]
>
> "got 1596128123118000000" is 4h earlier than expected, and my TZ is UTC-4.

This was reproducible by having /etc/timezone set to something other than UTC, and having TZ **unset**. The fix is in ba2e24710fb7c6d8236f81ee79ca30ca7dcbcf9c.

#18 - 08/03/2020 03:24 PM - Tom Clegg

Nit: better to put the "always send UTC to gofakes3" rule in the s3AWSFakeClock code, instead of relying on the caller. This works for me:

```
diff --git a/services/keepstore/s3aws_volume_test.go b/services/keepstore/s3aws_volume_test.go
index 33e671f3b..97045a660 100644
--- a/services/keepstore/s3aws_volume_test.go
+++ b/services/keepstore/s3aws_volume_test.go
```

```
@@ -43,7 +43,7 @@ func (c *s3AWSFakeClock) Now() time.Time {
         if c.now == nil {
                 return time.Now().UTC()
         }
-        return *c.now
+        return c.now.UTC()
 }

 func (c *s3AWSFakeClock) Since(t time.Time) time.Duration {
@@ -336,7 +336,7 @@ func (s *StubbedS3AWSSuite) TestBackendStates(c *check.C) {
                 }
         }

-        t0 := time.Now().UTC()
+        t0 := time.Now()
         nextKey := 0
         for _, scenario := range []struct {
                 label               string
```

#19 - 08/03/2020 04:05 PM - Ward Vandewege

Tom Clegg wrote:

> Nit: better to put the "always send UTC to gofakes3" rule in the s3AWSFakeClock code, instead of relying on the caller. This works for me:
>
> [...]

Cool, I've rebased like that in 5771cf273a4e09a5666122a7f67b4f088927e29d

#20 - 08/04/2020 08:07 PM - Ward Vandewege

- % Done changed from 0 to 100

- Status changed from In Progress to Resolved

Applied in changeset arvados|a37291f5f992a082d88efb9cdf57cd92c710e883.

#21 - 08/04/2020 08:44 PM - Ward Vandewege

Provisional migration plan: behind feature flag on 2.1; default in 2.2; remove old driver in 2.3.

#22 - 10/07/2020 02:11 AM - Peter Amstutz

- Release set to 25