NoSuchBucket

Tests, lies and eventual consistency.

```
Finished in 6 minutes 47 seconds (files took 2.51 seconds to load)
327 examples, 0 failures, 2 pending

Randomized with seed 26643

Coverage report generated for RSpec to /Users/julik/Code/we/nu_backend/coverage. 2122 / 2251 LOC (94.27%) covered.
```

🌨️

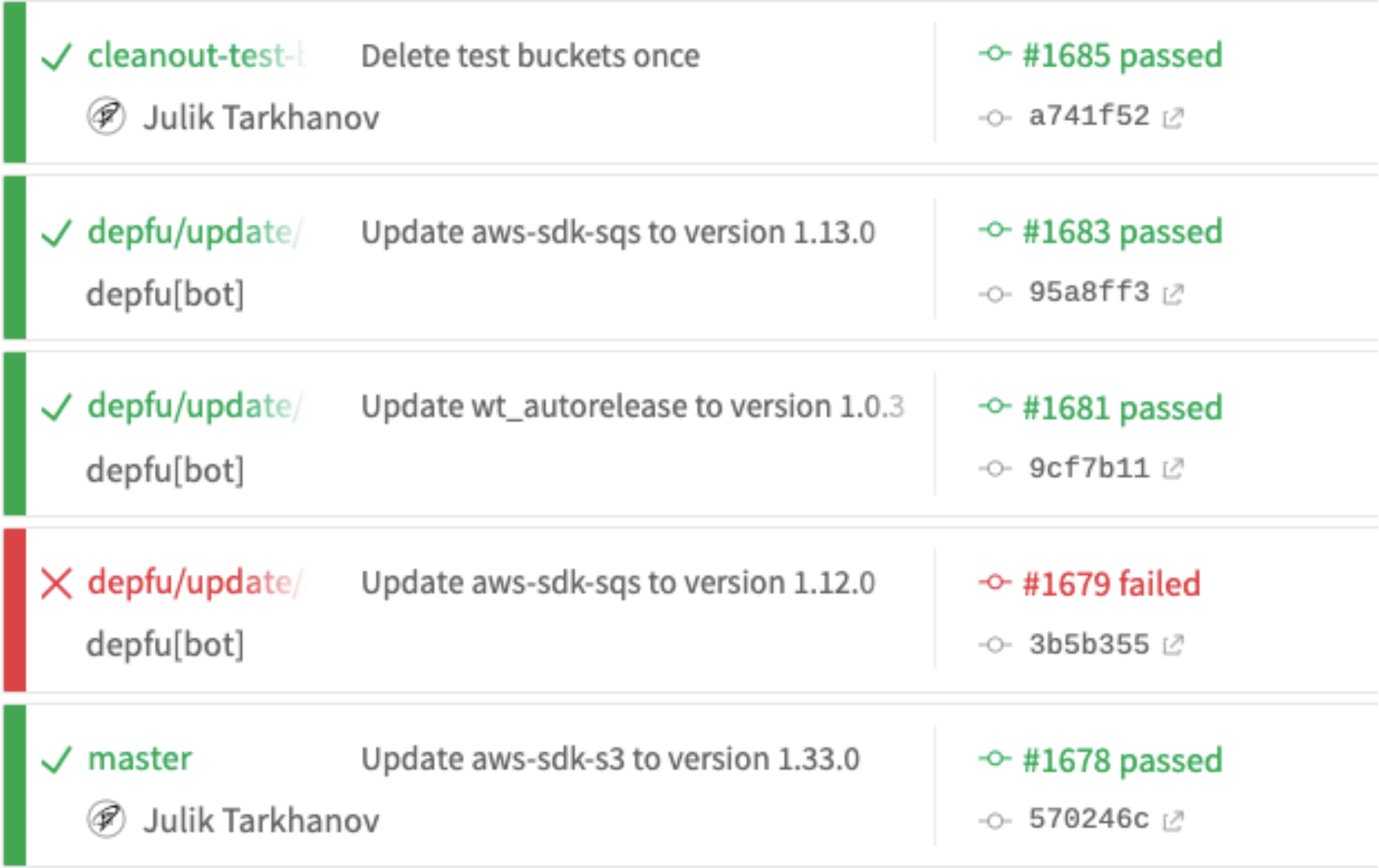
🌨️

🌨️

🌨️

🌨️

```
Failures:
  1) NuBackend::Jobs::FinishTransfer adds the empty directories defined prior to the manifest
     Failure/Error: @in_bucket.objects.batch_delete!
     Aws::S3::Errors::NoSuchBucket:
       The specified bucket does not exist
     # ./vendor/bundle/ruby/2.5.0/gems/aws-sdk-core-3.54.2/lib/seahorse/client/plugins/raise_response_errors.rb:15:in `call'
     # ./vendor/bundle/ruby/2.5.0/gems/aws-sdk-s3-1.41.0/lib/aws-sdk-s3/plugins/sse_cpk.rb:22:in `call'
     # ./vendor/bundle/ruby/2.5.0/gems/aws-sdk-s3-1.41.0/lib/aws-sdk-s3/plugins/dualstack.rb:26:in `call'
     # ./vendor/bundle/ruby/2.5.0/gems/aws-sdk-s3-1.41.0/lib/aws-sdk-s3/plugins/accelerate.rb:35:in `call'
     # ./vendor/bundle/ruby/2.5.0/gems/aws-sdk-core-3.54.2/lib/aws-sdk-core/plugins/jsonvalue_converter.rb:20:in `call'
     # ./vendor/bundle/ruby/2.5.0/gems/aws-sdk-core-3.54.2/lib/aws-sdk-core/plugins/idempotency_token.rb:17:in `call'
```



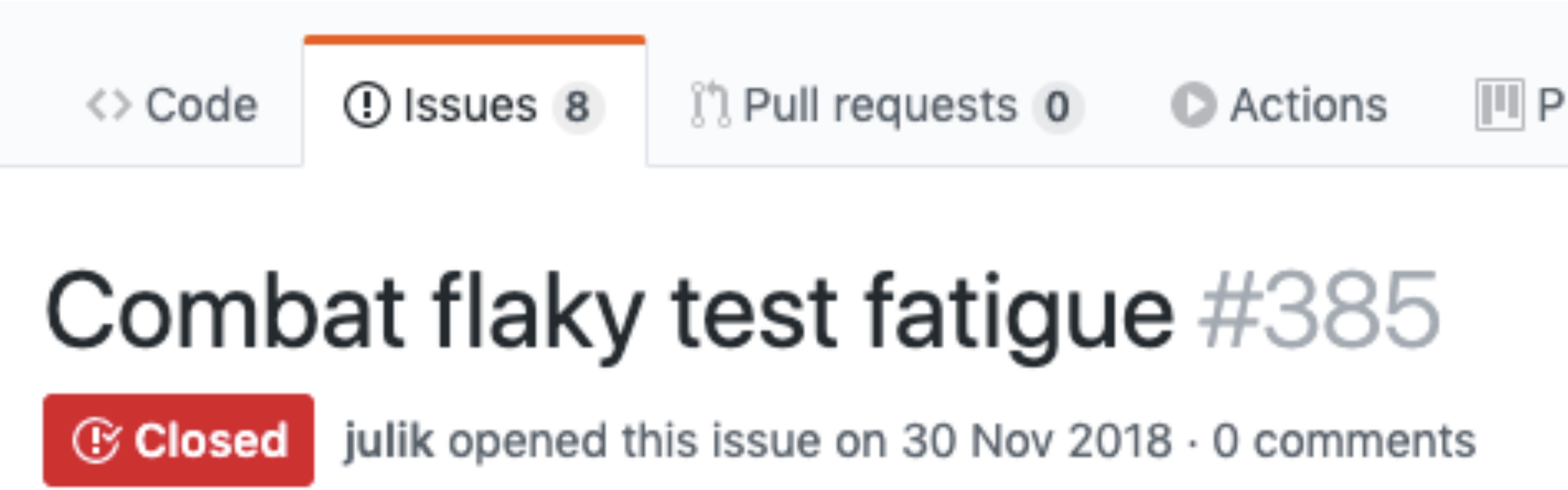
✓ cleanout-test- Delete test buckets once
Julik Tarkhanov
#1685 passed
a741f52

✓ depfu/update/ Update aws-sdk-sqs to version 1.13.0
depfu[bot]
#1683 passed
95a8ff3

✓ depfu/update/ Update wt_autorelease to version 1.0.3
depfu[bot]
#1681 passed
9cf7b11

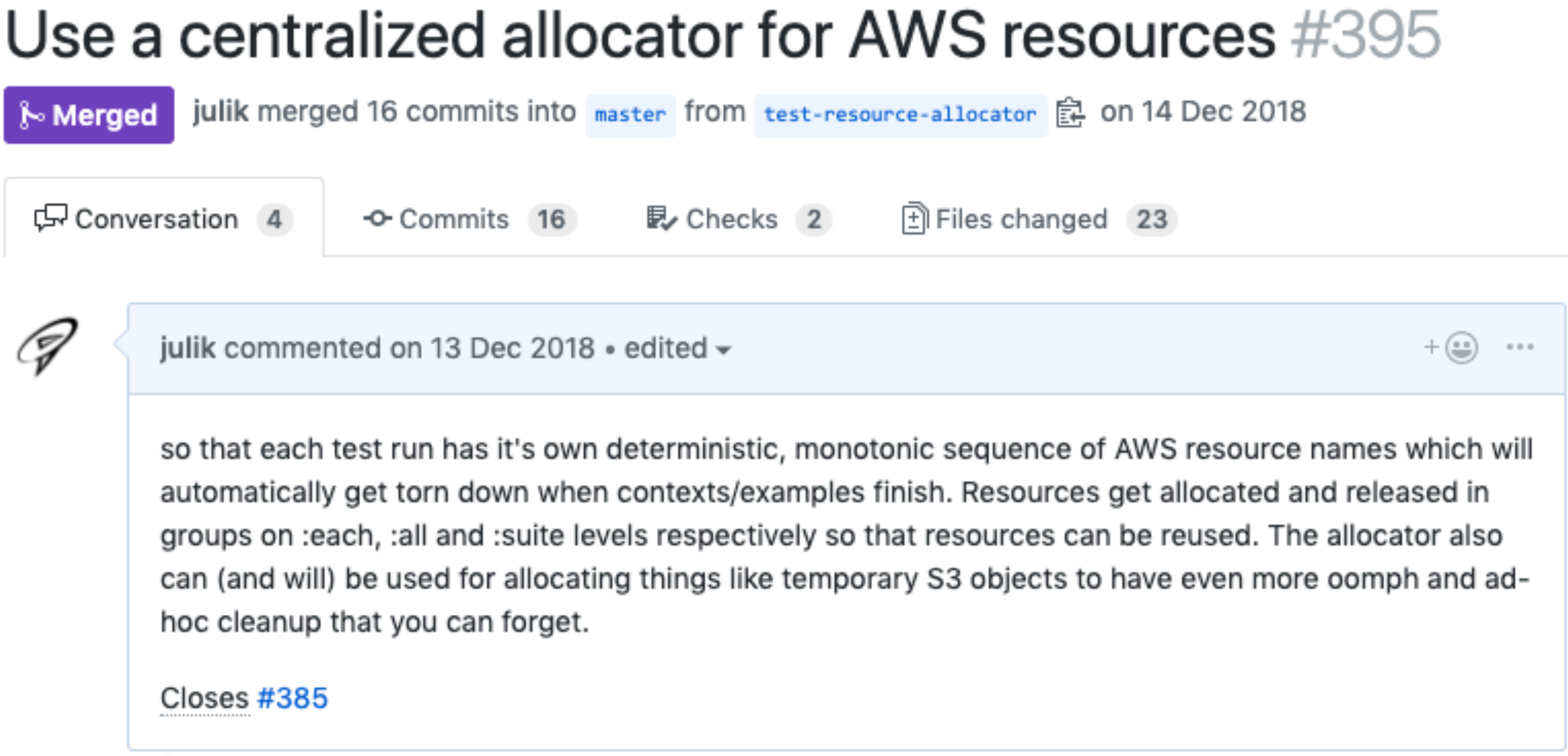
✗ depfu/update/ Update aws-sdk-sqs to version 1.12.0
depfu[bot]
#1679 failed
3b5b355

✓ master Update aws-sdk-s3 to version 1.33.0
Julik Tarkhanov
#1678 passed
570246c



Combat flaky test fatigue #385

julik opened this issue on 30 Nov 2018 · 0 comments



Use a centralized allocator for AWS resources #395

julik merged 16 commits into `master` from `test-resource-allocator` on 14 Dec 2018

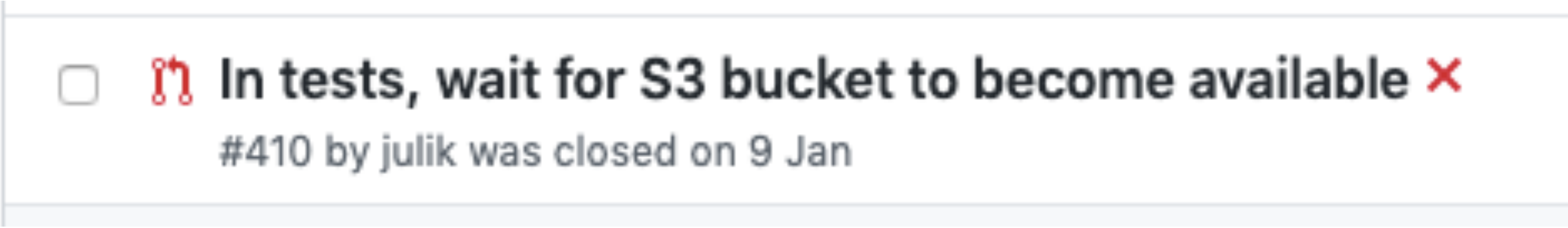
julik commented on 13 Dec 2018 • edited

so that each test run has it's own deterministic, monotonic sequence of AWS resource names which will automatically get torn down when contexts/examples finish. Resources get allocated and released in groups on :each, :all and :suite levels respectively so that resources can be reused. The allocator also can (and will) be used for allocating things like temporary S3 objects to have even more oomph and ad-hoc cleanup that you can forget.

Closes #385

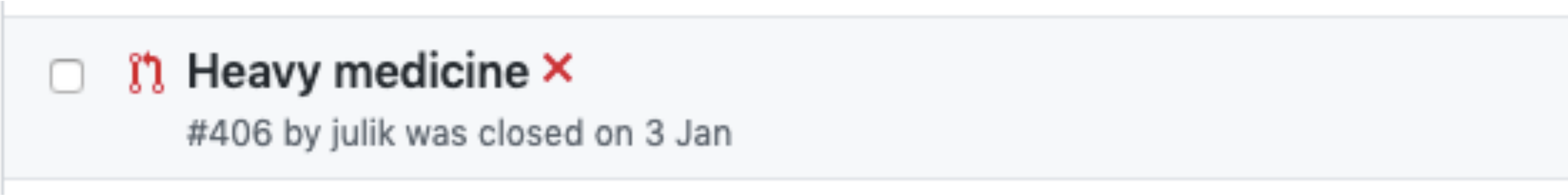
Nope.

Let's try fixing things and throwing them at the wall until it sticks.



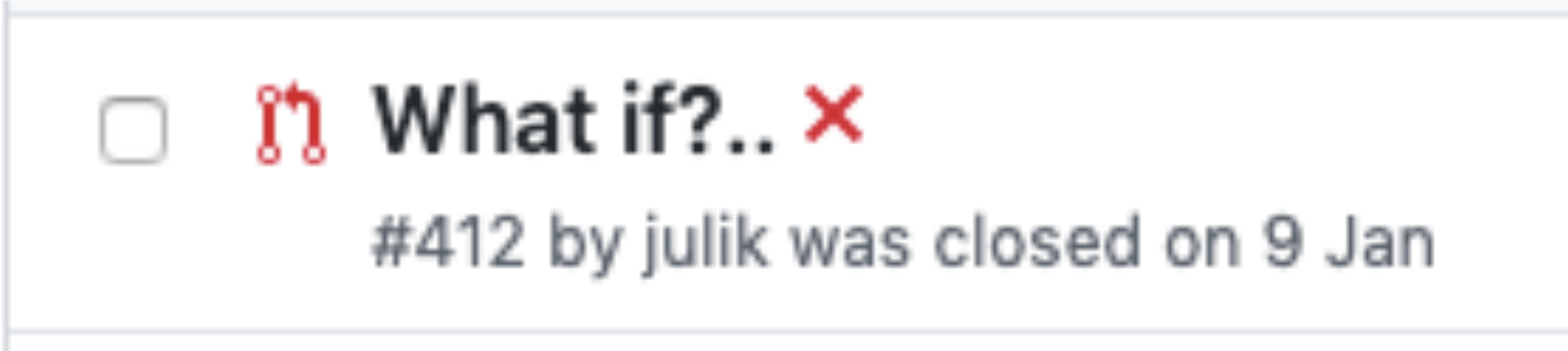
□ In tests, wait for S3 bucket to become available ✗

#410 by julik was closed on 9 Jan



□ Heavy medicine ✕

#406 by julik was closed on 3 Jan

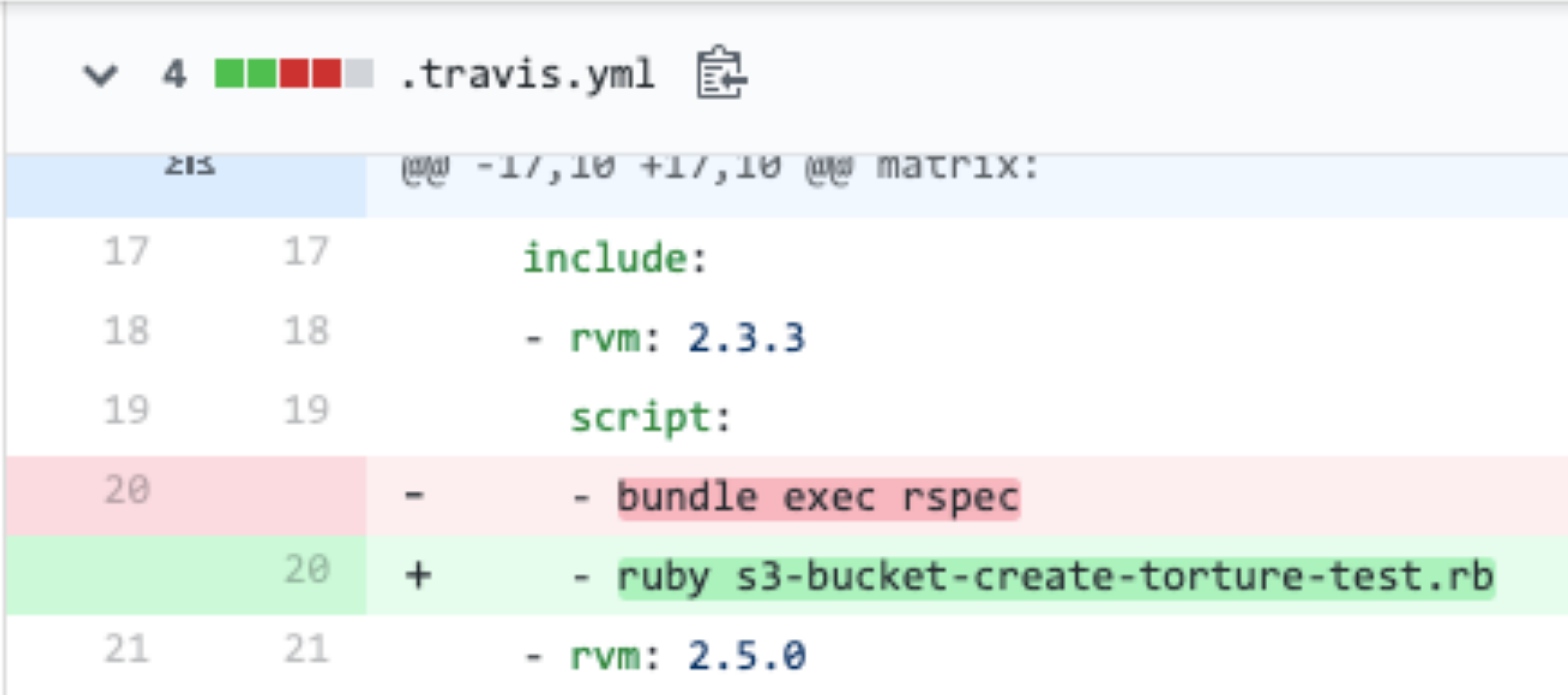


□ What if?.. ✗

#412 by julik was closed on 9 Jan

What if it isn't us?

- Mockists were right after all?
- Cosmic rays?
- Everything must be rewritten in Go ASAP?



4 🟩🟩🟥🟥⬜ .travis.yml

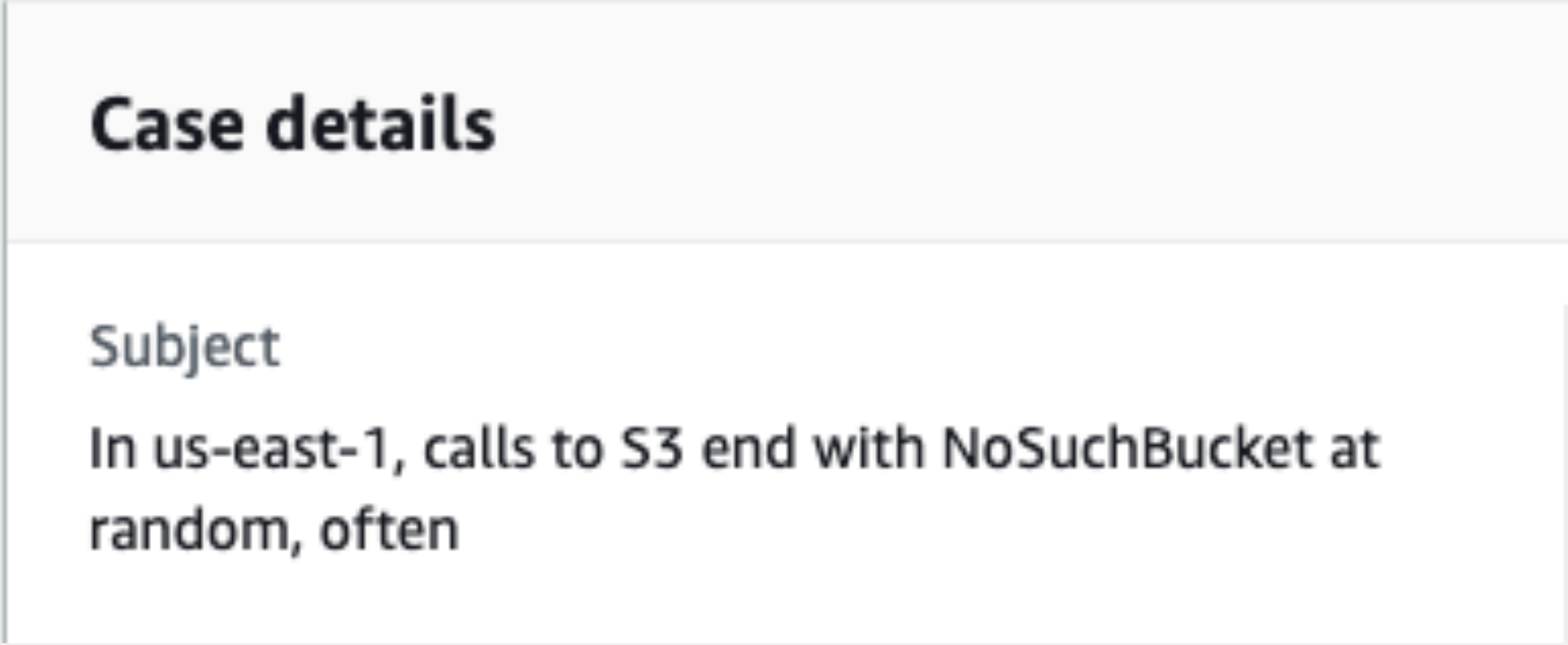
```
@@ -17,10 +17,10 @@ matrix:
   include:
   - rvm: 2.3.3
     script:
-    - bundle exec rspec
+    - ruby s3-bucket-create-torture-test.rb
   - rvm: 2.5.0
```

```ruby
total = 50
failed = 0
ok = 0

total.times do |n|
  bucket_name = "wt-nbt-#{random_salt}-#{n}"
  s3_client = Aws::S3::Client.new
  begin
    s3_client.create_bucket(bucket: bucket_name)

    bucket_resource = Aws::S3::Bucket.new(bucket_name)
    bucket_resource.object('.keep').put(body: StringIO.new(''))

    $stderr << '.'
    ok += 1
  rescue Aws::S3::Errors::NoSuchBucket
    $stderr << 'X'
    failed += 1
  ensure
    bucket_resource.delete! rescue nil
  end
end
```

```
..X...........X....................X................X...............................................
RuntimeError: Of 100 calls total, 4 failed with NoSuchBucket. This is no good.
```

Mean 5% failure rate

For all buckets.
In all cases.
In all regions.



Case details

Subject

In us-east-1, calls to S3 end with NoSuchBucket at random, often

…check if the bucket exists before before performing an operation.

```
s3.wait_until(:bucket_exists, bucket:bucket_name)
```

Mean 2% failure rate

For all buckets.
In all cases.
In all regions.

...inserted sleep(3) to wait for 3 seconds before putting the object, and the number of exceptions was 0

327 test cases * sleep(3) = ...

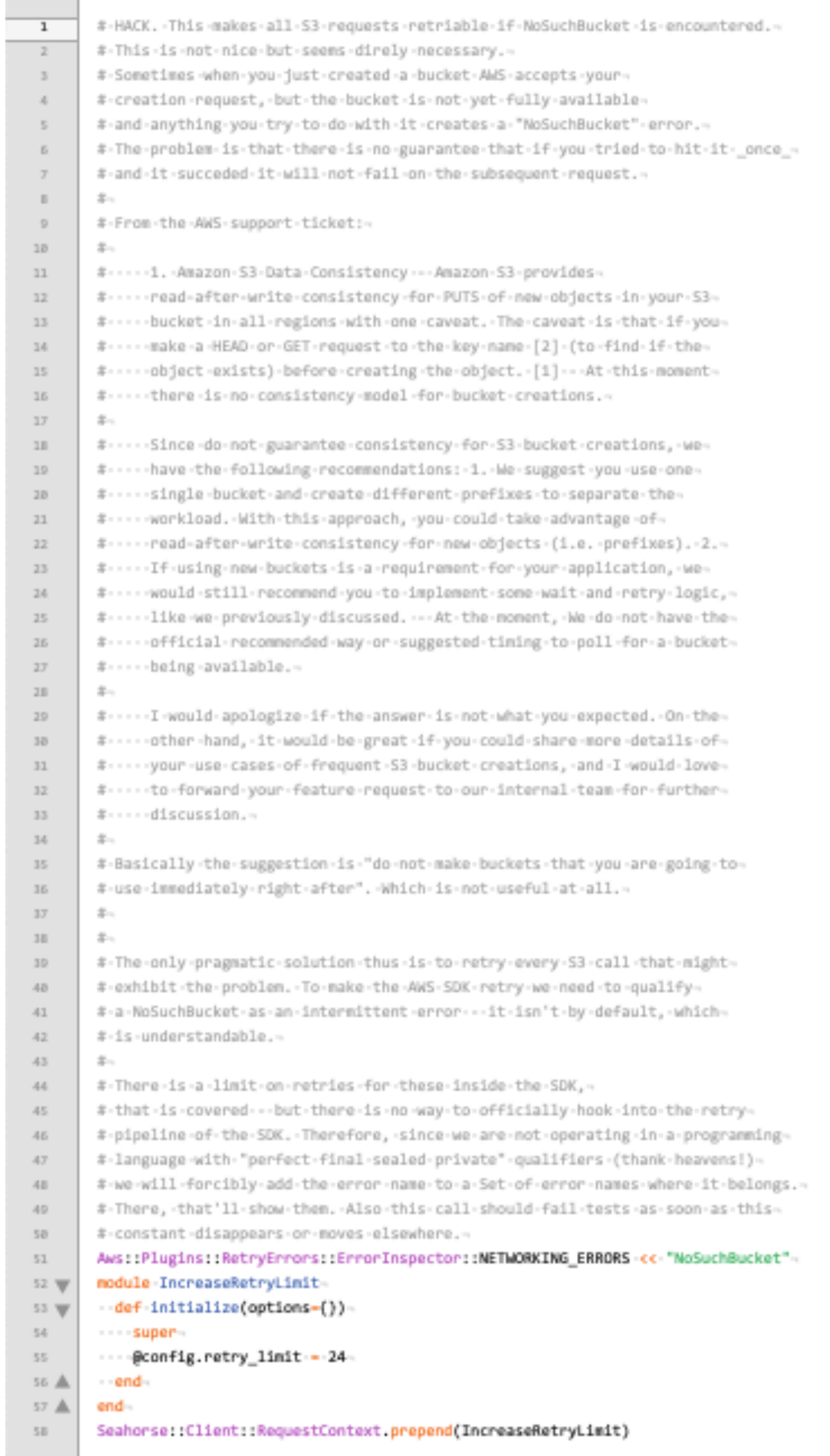
☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕
☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕
☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕
☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕☕

...do not have the official recommended way or suggested timing to poll for a bucket being available.

- S3 bucket creation is eventually consistent
- Some requests to a fresh bucket might fail while the one before succeeded
- Inherent property of the system

How do we make every SDK operation retry `NoSuchBucket`?

```
sealed final concrete inline private retryPolicy(...) {
   // keep out, our code is perfection distilled.
}
```



```ruby
# HACK. This makes all S3 requests retriable if NoSuchBucket is encountered.
# This is not nice but seems direly necessary.
# Sometimes when you just created a bucket AWS accepts your
# creation request, but the bucket is not yet fully available
# and anything you try to do with it creates a "NoSuchBucket" error.
# The problem is that there is no guarantee that if you tried to hit it _once_
# and it succeded it will not fail on the subsequent request.
#
# From the AWS support ticket:
#
#     1. Amazon S3 Data Consistency   Amazon S3 provides
#     read after write consistency for PUTS of new objects in your S3
#     bucket in all regions with one caveat. The caveat is that if you
#     make a HEAD or GET request to the key name [2] (to find if the
#     object exists) before creating the object. [1]  At this moment
#     there is no consistency model for bucket creations.
#
#     Since do not guarantee consistency for S3 bucket creations, we
#     have the following recommendations: 1. We suggest you use one
#     single bucket and create different prefixes to separate the
#     workload. With this approach, you could take advantage of
#     read after write consistency for new objects (i.e. prefixes). 2.
#     If using new buckets is a requirement for your application, we
#     would still recommend you to implement some wait and retry logic,
#     like we previously discussed.   At the moment, We do not have the
#     official recommended way or suggested timing to poll for a bucket
#     being available.
#
#     I would apologize if the answer is not what you expected. On the
#     other hand, it would be great if you could share more details of
#     your use cases of frequent S3 bucket creations, and I would love
#     to forward your feature request to our internal team for further
#     discussion.
#
# Basically the suggestion is "do not make buckets that you are going to
# use immediately right after". Which is not useful at all.
#
#
# The only pragmatic solution thus is to retry every S3 call that might
# exhibit the problem. To make the AWS SDK retry we need to qualify
# a NoSuchBucket as an intermittent error  it isn't by default, which
# is understandable.
#
# There is a limit on retries for these inside the SDK,
# that is covered  but there is no way to officially hook into the retry
# pipeline of the SDK. Therefore, since we are not operating in a programming
# language with "perfect final sealed private" qualifiers (thank heavens!)
# we will forcibly add the error name to a Set of error names where it belongs.
# There, that'll show them. Also this call should fail tests as soon as this
# constant disappears or moves elsewhere.
Aws::Plugins::RetryErrors::ErrorInspector::NETWORKING_ERRORS << "NoSuchBucket"
module IncreaseRetryLimit
  def initialize(options={})
    super
    @config.retry_limit = 24
  end
end
Seahorse::Client::RequestContext.prepend(IncreaseRetryLimit)
```

```
Aws::Plugins::RetryErrors::ErrorInspector::NETWORKING_ERRORS << "NoSuchBucket"
module IncreaseRetryLimit
  def initialize(options={})
    super
    @config.retry_limit = 24
  end
end
Seahorse::Client::RequestContext.prepend(IncreaseRetryLimit)
```

❤ AWS Ruby SDK team

Be aware of the systems you use.

Ask your friendly vendor for help, because they know the thing and you probably don't.

And finally….

Sometimes it is not you.

Fixing Flaky Tests Like a Detective by Sonja Peterson
https://www.youtube.com/watch?v=qTyoMg_rmrQ

WeTransfer