Ruby trunk - Misc #15220

Adding OpenSSL 1.1.1 on Travis CI gcc-8 case

10/09/2018 05:30 PM - jaruga (Jun Aruga)

Status:	Closed	
Priority:	Normal	
Assignee:		

Description

Currently Travis CI test cases are running on OpenSSL 1.0.1f as the default setting.

I want to add the latest version OpenSSL 1.1.1 to the gcc-8 test case on Travis CI.

I sent pull-request for that.
https://github.com/ruby/ruby/pull/1980

Motivation

The motivation is that ruby/openssl has very good CI environment covering the supported SSL collections.

However the benefits to enable OpenSSL 1.1.1 for the main repository (ruby/ruby) are

1. To make us check the entire logic on the latest OpenSSL as early as possible. For example, I can find this kind of issue [1] as early as possible.

2. We can check it on ruby-2.5 branch too. That is related to #15219 [2]

As we are working for OpenSSL 1.1.1 on Windows CI [3] and python project is testing on the custom OpenSSL built from the source code [4], I think that we can adapt it to Travis CI.

Detail of implementation

In the new process, the OpenSSL is built from source code.
That takes 134.63 sec = 2 min 14 sec
I could not find the better idea than that.

I found the binary openssl package that someone is managing [5].
But it might not be valid for Trusty. And it seems that the repository is quite personal one.

I am using cache/directories element [6].

I created tool/install_openssl.sh script. But some logic might be moved to configure.ac like [7].

How do you think?

- [1] net/http, net/ftp: fix session resumption with TLS 1.3: https://github.com/ruby/ruby/commit/1dfc377
- [2] Ruby 2.5.X supporting OpenSSL 1.1.1 and TLS 1.3: https://bugs.ruby-lang.org/issues/15219
- [3] OpenSSL 1.1.1 soon available from MSYS2: https://bugs.ruby-lang.org/issues/15171
- [4] Python with custom OpenSSL: https://github.com/python/cpython/blob/master/.travis.yml#L15
- [5] Personal(?) openssl repository: https://launchpad.net/~xnox/+archive/ubuntu/openssl
- [6] Travis cache: https://docs.travis-ci.com/user/caching/#arbitrary-directories
- [7] deduce versioned tools from CC: https://github.com/ruby/ruby/commit/f677ba8

History

#1 - 10/10/2018 09:44 AM - jaruga (Jun Aruga)

- Description updated

#2 - 10/11/2018 05:30 AM - shyouhei (Shyouhei Urabe)

First, I understand the motivation behind this request. We should enrich our build matrix.

That said, 2+ min overhead for each and every time we commit something is too much. I can hardly +1 this particular patch.

Maybe we can use docker like ruby/openssl does today, or maybe we can have our own .deb prepared somewhere.

#3 - 10/12/2018 04:31 PM - jaruga (Jun Aruga)

> Maybe we can use docker like ruby/openssl does today, or maybe we can have our own .deb prepared somewhere.

I sent pull-request to use docker on the only gcc-8 test case as an experiment.
https://github.com/ruby/ruby/pull/1983

The result on my repository is here.
https://travis-ci.org/junaruga/ruby/builds/440699457

I installed needed OpenSSL 1.1.1 in the docker container image in advance.
But unfortunately the total running time is longer than above OpenSSL source compiled case.
But I think this experiment includes some tips to run ruby tests in the docker container.

I did put the built container image here.
https://hub.docker.com/r/junaruga/ruby-docker/tags/

Top directory's Dockerfile would refer this container image.

Use case

Build the base container, and push the built image to the repository.

```
$ cd tool/ci
$ docker build --rm --no-cache -t ruby-docker .
$ docker login docker.io
$ docker tag ruby-docker docker.io/junaruga/ruby-docker:test
$ docker push docker.io/junaruga/ruby-docker:test
```

Run the test on local.
If you are using docker >= 17.06, maybe you need --network=host option to docker run.
See .travis.yml for detail.

```
$ cd ../.. (<= Back to top directory)
```

```
$ docker build --rm -t ruby .
$ docker run --rm -t ruby tool/ci/test.sh
```

You can set arguments optionally.

```
$ docker build --rm \
  --build-arg TEST_USER=travis \
  --build-arg WORK_DIR=$(pwd) \
  -t ruby .
```

```
$ docker run \
  -e CC="gcc-8" \
  -e OPENSSL_VERSION="1.1.1" \
  ruby tool/ci/test.sh
```

- tool/ci/test.sh is same with commands on previous Travis's before_install, before_script and script section.
- We might be able to run below test commands in parallel with multi Travis instances to save the running time.

```
make -s test => Travis instance 1
make -s test-all => Travis instance 2
make -s test-spec => Travis instance 1
```

#4 - 10/12/2018 04:51 PM - jaruga (Jun Aruga)

- As an another idea to run the heavy task such as installing openssl from source or running the docker. If we are running on Travis's cron mode too, we can run the heavy tasks for only cron running.

.travis.yml

```
script: |
  if [ "${TRAVIS_EVENT_TYPE}" = "cron" ]; then
    a heavy task
  else
    a light task
  fi
```

#5 - 10/16/2018 11:30 AM - jaruga (Jun Aruga)

So far I proposed 2 possible solutions to test with OpenSSL 1.1.1 with the pull-requests.

- 1. Add OpenSSL 1.1.1 test case to Travis CI: https://github.com/ruby/ruby/pull/1980
    - Demerit:
        - 2+ minutes to install OpenSSL 1.1.1 every time is not comfortable.
        - Added new commit for conditional logic with TRAVIS_EVENT_TYPE is not clear from Travis jobs page.
- 2. Use docker image to test OpenSSL 1.1.1: https://github.com/ruby/ruby/pull/1983
    - Merit: It's a scalable way to test on various environment with source built dependencies.
    - Demerit:
        - Total running time 17 minutes is longer than the solution 1.
        - It's harder to maintain the docker image.

And here is 3rd possible solution now.
This is the most recommended and realistic way that I think. :)

Add a conditional job to test new version OpenSSL 1.1.1 to Travis CI.
https://github.com/ruby/ruby/pull/1984

- Merit:
    - The total running time is same with current one.
    - We can check supported test cases regularly when Travis is run on cron mode or we set special environment variable from Travis setting page.
    - The run job is clear from Travis jobs page.
- Demerit:
    - OpenSSL 1.1.1 test case is not run on pull-request.

#6 - 10/17/2018 02:16 PM - shyouhei (Shyouhei Urabe)

Hello,

This might not exactly be what you want but I have just enabled os: osx, whose homebrew accidentally includes openssl 1.1.1.
It seems the builds are seconds slower than linux ones, but at least faster than installing openssl every time from the tarball.
Is this sufficient for you?

#7 - 10/17/2018 04:10 PM - k0kubun (Takashi Kokubun)

As I saw Travis build failure and didn't see this ticket when I read r65122, I somehow changed openssl version at r65124 (sorry for that, not intended to bother this ticket) but I reverted my change at r65131 (and r65138) and so openssl 1.1 CI is there.

Since it has hanged multiple times after r65122, I tentatively added the matrix to allow_failures at r65138, but I know we want to know build failure by openssl 1.1.1, macOS or clang immediately. While I felt it's unstable at first, it also succeeded multiple times recently. Let's monitor how it goes for now.

#8 - 10/17/2018 04:48 PM - jaruga (Jun Aruga)

shyouhei (Shyouhei Urabe) wrote:
...

> This might not exactly be what you want but I have just enabled os: osx, whose homebrew accidentally includes openssl 1.1.1.
> It seems the builds are seconds slower than linux ones, but at least faster than installing openssl every time from the tarball.
> Is this sufficient for you?

Very cool! Thanks for the working.
This is sufficient and great solution for me.

Seeing the history of the openssl@1.1.rb file, after the new OpenSSL 1.1.1 or 1.1.0i were released, the brew package was quickly updated to the latest version by someone within the same day.
https://formulae.brew.sh/formula/openssl@1.1
https://github.com/Homebrew/homebrew-core/blob/master/Formula/openssl%401.1.rb

However I just want to ask you why you do not like my above 3rd solution.
It's not additional cost when doing pull-request and pushing to branches, isn't it?

#9 - 10/18/2018 01:25 AM - shyouhei (Shyouhei Urabe)

jaruga (Jun Aruga) wrote:

> However I just want to ask you why you do not like my above 3rd solution.
> It's not additional cost when doing pull-request and pushing to branches, isn't it?

That pull request is OKish. I especially liked how cron-only build matrix is made possible.
The reason why I didn't press the merge button was that I couldn't immediately understand what's going on, mainly because of the creative use of YAML anchors in middle of actual definitions.
Now I understand that the anchors are introduced there to minimize the patch size. That is a good thing. I don't want to blame on it. Just it is a bit hard to maintain.

Anyway the series of pull request you sent is very informative.
Maybe we can add some cron-only builds (not for OpenSSL) using your technique. Thank you.

#10 - 10/18/2018 09:45 AM - jaruga (Jun Aruga)

shyouhei (Shyouhei Urabe) wrote:
...

> That pull request is OKish. I especially liked how cron-only build matrix is made possible.
> The reason why I didn't press the merge button was that I couldn't immediately understand what's going on, mainly because of the creative use of YAML anchors in middle of actual definitions.
> Now I understand that the anchors are introduced there to minimize the patch size. That is a good thing. I don't want to blame on it. Just it is a bit hard to maintain.

I would admit the anchors are hard to maintain. Especially the .definitions/if part.
The intent for the anchors is like you assumed. I did not want to write same logic to multi parts.

There are 2 cautions to use anchors.

1. The anchor & part needs to be defined before the reference * part in a YAML file.
2. When the reference * refers to the not existed or not loaded anchor &, it is still valid without error at least for Travis CI's YAML parser.

As an example of anchors defined separately, I can introduce you a Shippable CI's document. [1]

> Anyway the series of pull request you sent is very informative.
> Maybe we can add some cron-only builds (not for OpenSSL) using your technique. Thank you.

Alright. That's great. I am looking forward to seeing it.
I can introduce you how to add osx & gcc-8 [2][3] and clang-N case [4][5] on Travis if you want those.cases.

I like that your openssl 1.1.1 patch on trunk branch will be backported to ruby_2_5 branch after "Let's monitor how it goes for now".

Feel free to close my pull-request series when you like it.

Thank you.

[1] Shippable CI template: http://docs.shippable.com/platform/workflow/job/runsh/#shippabletemplatesyml
[2] osx & gcc-8 case: .travis.yml: https://github.com/trinityrnaseq/trinityrnaseq/blob/master/.travis.yml
[3] osx & gcc-8 case: Travis: https://travis-ci.org/trinityrnaseq/trinityrnaseq
[4] clang-N case: .travis.yml: https://github.com/Microsoft/GSL/blob/master/.travis.yml
[5] clang-N case: Travis: https://travis-ci.org/Microsoft/GSL

#11 - 10/25/2018 05:16 AM - shyouhei (Shyouhei Urabe)

- Status changed from Open to Closed

Let me close this now. Thank you for the report!

#12 - 10/25/2018 10:38 AM - jaruga (Jun Aruga)

> Let me close this now. Thank you for the report!

Yeah, thank you for closing. I saw the updated .travis.yaml.
osx & OpenSSL 1.1.1, i686 (32-bit), valgrind, a case to show compiler warnings, and etc.
Great work!