Arvados - Story #9945

[SDK] Package Python apps as virtualenvs

09/06/2016 01:50 PM - Peter Amstutz

Status:	Resolved	**Start date:**	09/07/2016
Priority:	Normal	**Due date:**	
Assigned To:	Ward Vandewege	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	2019-02-13 Sprint		

Description

Arvados tools and SDKs written in Python often require 3rd party packages that are not available as OS packages, or require a newer version than the OS package. Currently we package these as backports, but this a fairly high maintenance and (when upgrading existing OS packages) runs the risk of breaking the OS. Investigate the alternative of creating deb and rpm packages which use a Python virtualenv to isolate the package dependencies.

On brief survey I've found a couple of tools for doing this:

https://github.com/spotify/dh-virtualenv

https://github.com/kevinconway/rpmvenv

fpm also has support?

https://github.com/jordansissel/fpm/issues/697

https://github.com/jordansissel/fpm/pull/930

Subtasks:

Task # 9963: bring our fpm fork in line with latest head	**Resolved**

Related issues:

Related to Arvados - Bug #9944: [CWL] python-lockfile version conflict	**Resolved**
Related to Arvados - Bug #14326: Our custom-compiled `python-future` and `pyt...	**Resolved**

Associated revisions

Revision 84b30d02 - 02/04/2019 03:35 PM - Ward Vandewege

Merge branch '9945-make-python-package-dependency-free'

closes #9945

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <wvandewege@veritasgenetics.com>

Revision 71263c36 - 02/04/2019 05:46 PM - Ward Vandewege

The libpam-arvados package on Centos7 has a dependency on the python-pam package.

refs #9945

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <wvandewege@veritasgenetics.com>

Revision 2b6be859 - 02/04/2019 10:21 PM - Ward Vandewege

Do not try to test packages that were not built.

refs #9945

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <wvandewege@veritasgenetics.com>

Revision 41d2a628 - 02/05/2019 01:36 PM - Ward Vandewege

We have reset the package iterations to -1, make the
run-build-docker-jobs-image.sh script reflect that.

refs #9945

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <wvandewege@veritasgenetics.com>

Revision 09939997 - 02/05/2019 10:14 PM - Ward Vandewege

Make the python-arvados-cwl-runner package expose the cwltool executable, make
it conflict with the python-cwltool (and cwltool) packages, and add a note to
the upgrade documentation to that effect. Also document the Centos7 specific
upgrade from python33 to rh-python35 as part of story 9945.

refs #9945

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <wvandewege@veritasgenetics.com>

Revision cdd31206 - 02/07/2019 04:52 PM - Ward Vandewege

In the arvados/jobs image, use the Python executable from the
python-arvados-python-client package by default.

refs #9945

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <wvandewege@veritasgenetics.com>

Revision e541e210 - 02/07/2019 05:42 PM - Ward Vandewege

Now python-arvados-cwl-runner includes a virtualenv, make sure to use its
python executable when importing the arvados_cwl python module.

refs #9945

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <wvandewege@veritasgenetics.com>

Revision 305089e2 - 03/12/2019 02:34 PM - Ward Vandewege

Build the cwltest package, which lives out of tree.

refs #9945

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <wvandewege@veritasgenetics.com>

Revision 35faa53a - 05/23/2019 06:45 PM - Ward Vandewege

Make sure that the libpam-arvados package puts its various files in the place
where the pam subsystem will expect them, not inside the virtualenv.

Also, remove the code from sdk/pam/setup.py that installed a copy of the
libpam-arvados.py file under /usr/data/lib/security/libpam_arvados.py for
legacy reasons.

refs #9945
refs #15186

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <wvandewege@veritasgenetics.com>

Revision 1fd7b708 - 06/05/2019 04:49 PM - Ward Vandewege

Crunch-job needs to call the python executable from our sdk package for
collating job output.

refs #9945

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <wvandewege@veritasgenetics.com>

Revision 4c63c682 - 06/05/2019 05:53 PM - Ward Vandewege

Merge branch '9945-crunch-job-python-path'

refs #9945

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <wvandewege@veritasgenetics.com>

Revision a8fe18b2 - 06/05/2019 05:55 PM - Ward Vandewege

Crunch-job needs to call the python executable from our sdk package for
collating job output.

refs #9945

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <wvandewege@veritasgenetics.com>

Revision 85577ffe - 06/07/2019 01:16 AM - Ward Vandewege

Fix the build by disabling an old crunch-job (jobs api) test that now
depends on the presence of a post 1.3 version of the
python-arvados-python-client package being present.

refs #9945

Arvados-DCO-1.1-Signed-off-by: Ward Vandewege <wvandewege@veritasgenetics.com>

History

#1 - 09/06/2016 01:57 PM - Peter Amstutz

- Description updated

#2 - 09/06/2016 02:08 PM - Peter Amstutz

- Description updated

#3 - 09/06/2016 02:09 PM - Peter Amstutz

- Description updated

#4 - 09/07/2016 02:23 AM - Peter Amstutz

```
root@e9972d7acb01:~# cat > after-inst <<EOF
> #!/bin/sh
> cd /usr/bin
> ln -s /usr/share/python/arvados-cwl-runner/bin/arvados-cwl-runner
> ln -s /usr/share/python/arvados-cwl-runner/bin/arvados-cwl-runner cwl-runner
> EOF
root@e9972d7acb01:~# fpm --after-install=after-inst -s virtualenv -t deb -n arvados-cwl-runner arvados-cwl-run
ner
```

#5 - 09/07/2016 02:31 AM - Peter Amstutz

Related, using fpm+gem to produce self-contained Ruby packages:
http://blog.gemnasium.com/post/60091718748/package-your-ruby-based-tool-the-safe-and-easy-way

#6 - 08/29/2017 01:31 PM - Tom Morris

- Target version set to Arvados Future Sprints

#7 - 12/12/2018 04:36 PM - Ward Vandewege

- Status changed from New to In Progress

- Assigned To set to Ward Vandewege

- Target version changed from Arvados Future Sprints to 2018-12-21 Sprint

#8 - 01/02/2019 04:19 PM - Tom Morris

- Target version changed from 2018-12-21 Sprint to 2019-01-16 Sprint

#9 - 01/02/2019 04:21 PM - Ward Vandewege

- Related to Bug #14326: Our custom-compiled `python-future` and `python3-future` packages can't be installed together and have precedence added

#10 - 01/16/2019 04:24 PM - Tom Morris

- Target version changed from 2019-01-16 Sprint to 2019-01-30 Sprint

#11 - 01/22/2019 02:37 AM - Ward Vandewege

Ready for (initial?) review at 5a09ff5d42b0f8b71ca6775813e0844067363e12

This still builds our backported libcloud package.

#12 - 01/22/2019 07:51 PM - Tom Clegg

I'm not sure why this

```
set -e
arvados-node-manager --version
set +e

PYTHON=`ls /usr/share/python2.7/dist/arvados-node-manager/bin/python2.7`

if [ "$PYTHON" != "" ]; then
  set -e
  exec $PYTHON <<EOF
...
else
  exit 1
fi
```

...isn't it equivalent to just using an explicit python binary like this?

```
set -e
arvados-node-manager --version
exec /usr/share/python2.7/dist/arvados-node-manager/bin/python2.7 <<EOF
...
```

Of course, the whole idea of an "import" test seems a bit weird here. Mostly this is scope creep, but I'll observe anyway...

- In the arvados-node-manager case, it tests some libcloud thing that (I'm guessing) isn't tested by "arvados-node-manager --version", which seems fine.
- In the arvados-cwl-runner case, I'm not sure what the inline script tells us that `arvados-cwl-runner --version` doesn't.
- In the arvados-fuse case, it looks like we test a script that imports arvados_fuse (which nobody does, afaik) but we don't try running "arv-mount --version", which seems more relevant. This seems like a pre-existing bug, but now might be an appropriate time to fix this one, since "run arv-mount from PATH and expect it to import all its things, including llfuse C libs" is the sort of thing we could be breaking here.
- Likewise, in arvados-python-client, we should add something like "arv-put --help".

In fpm_build_virtualenv...

The "*" should be quoted or escaped here:

```
+  find build -iname *.pyc -exec rm {} \;
+  find build -iname *.pyo -exec rm {} \;
```

There are some explicit checks for $? so I'm suspecting "set -e" isn't in force in fpm_build_virtualenv, so there are some unchecked errors (including the above "find" commands, "pip install pip", and "cp -f"). I suppose we can't do "set -e" somewhere without accidentally making global changes and having to rewrite Everything...

Suggest

```
-if [[ "${DEBUG:-0}" != "0" ]]; then ...
+if [[ -n "${DEBUG}" ]]; then ...
```

Some typos "Arvadow", "execurin", "sectiontoo"

What's the story behind changing the version discovery? It used to be passed in from run-build-packages ("ARVADOS_BUILDING_VERSION if given, else extract from egg PKG-INFO / ARVADOS_BUILDING_ITERATION if given, else depends on package"); now, run-library runs "setup.py sdist" and parses the resulting tarball filename to get VERSION, and defaults to iteration "-1". (I think setting iteration to -1 is a good move, except that if we want this merge to produce a set of dev packages, we might need to either set iteration to -5 or update the last-change-detection hairball to notice changes in /build. Ugh)

I think source:services/dockercleaner/arvados-docker-cleaner.service needs to be updated: python33→python35, and fix the hardcoded bin path used to detect scl:

```
ExecStart=/bin/sh -c 'if [ -e /opt/rh/python33/root/bin/arvados-docker-cleaner ]; then exec scl enable python3
3 arvados-docker-cleaner; else exec arvados-docker-cleaner; fi'
```

I'm wary of the way the prerm/postinst scripts and the fpm_build function are getting forked here.

- The new prerm/postinst scripts duplicate some stuff from the old ones, and don't seem to enable the systemd units, or check whether $1 indicates the appropriate hook (the old ones do this, so I'm guessing it's needed despite the names of the fpm args)
- The new prerm/postinst scripts get autogenerated seems messy so I'm wondering if there's another way. Could it list files in /usr/share/python*/dist/$pkg/bin/ at runtime? Or are there some fpm args we can use to install our own bin stubs/symlinks as /usr/bin/whatever? Even if we still need to autogenerate the stubs, at least that way yum/apt would know where they came from, avoid installing other packages with conflicting binaries, etc.
- The new fpm_build_virtualenv has a lot of overlap with fpm_build, but does some things differently (e.g., it calls test_package_presence instead of letting the caller do that). Meanwhile the old fpm_build still has lots of (now unused?) python-specific stuff. I don't know which is less messy (two similar funcs vs. lots of if-else). Splitting might be better but it would be good to keep them congruent.

Nit/style thing: I find it easier to follow "if ! foo; then ...; fi" than "foo; if $? != 0 ; then ...; fi". The first form also has an advantage that it works the same way in "set -e" and "set +e" contexts.

The deletion of build.list (and its version compatibility maintenance nightmare) sure makes me happy... this is very encouraging. Thanks!

#13 - 01/24/2019 06:30 PM - Ward Vandewege

Updated branch ready for another look at 89bc5fc945891bec4322fe14b6d11d0cdc1ca267

Tom Clegg wrote:

> I'm not sure why this
>
> [...]
>
> ...isn't it equivalent to just using an explicit python binary like this?
>
> [...]
>
> Of course, the whole idea of an "import" test seems a bit weird here. Mostly this is scope creep, but I'll observe anyway...
>
> - In the arvados-node-manager case, it tests some libcloud thing that (I'm guessing) isn't tested by "arvados-node-manager --version", which seems fine.
> - In the arvados-cwl-runner case, I'm not sure what the inline script tells us that `arvados-cwl-runner --version` doesn't.
> - In the arvados-fuse case, it looks like we test a script that imports arvados_fuse (which nobody does, afaik) but we don't try running "arv-mount --version", which seems more relevant. This seems like a pre-existing bug, but now might be an appropriate time to fix this one, since "run arv-mount from PATH and expect it to import all its things, including llfuse C libs" is the sort of thing we could be breaking here.
> - Likewise, in arvados-python-client, we should add something like "arv-put --help".

Agreed with all that, and all fixed.

> In fpm_build_virtualenv...
>
> The "*" should be quoted or escaped here:
>
> [...]

Done.

> There are some explicit checks for $? so I'm suspecting "set -e" isn't in force in fpm_build_virtualenv, so there are some unchecked errors (including the above "find" commands, "pip install pip", and "cp -f"). I suppose we can't do "set -e" somewhere without accidentally making global changes and having to rewrite Everything...

Yeah. I added some explicit exit code testing to the calls that are important (pip, etc). The find calls don't really matter that much, they just clean up compiled objects, but there shouldn't be any to start with since this typically runs in a clean checkout.

> Suggest
> [...]

The default being set here (0) was superfluous (fixed), we already do that higher in the script. But we can't just use the -n test, because $DEBUG is either 0 or 1.

> Some typos "Arvadow", "execurin", "sectiontoo"

All fixed.

> What's the story behind changing the version discovery? It used to be passed in from run-build-packages ("ARVADOS_BUILDING_VERSION if given, else extract from egg PKG-INFO / ARVADOS_BUILDING_ITERATION if given, else depends on package"); now, run-library runs "setup.py sdist" and parses the resulting tarball filename to get VERSION, and defaults to iteration "-1". (I think setting iteration to -1 is a good move, except that if we want this merge to produce a set of dev packages, we might need to either set iteration to -5 or update the last-change-detection hairball to notice changes in /build. Ugh)

Good call. I changed it to the original behavior, which is more succinct too. I'm not too worried about resetting the iteration to 1, because this branch touches sdk/python we're getting a new version for everything anyway.

> I think source:services/dockercleaner/arvados-docker-cleaner.service needs to be updated: python33→python35, and fix the hardcoded bin path used to detect scl:
> [...]

Yes! fixed.

> I'm wary of the way the prerm/postinst scripts and the fpm_build function are getting forked here.
>
> - The new prerm/postinst scripts duplicate some stuff from the old ones, and don't seem to enable the systemd units, or check whether $1 indicates the appropriate hook (the old ones do this, so I'm guessing it's needed despite the names of the fpm args)
> - The new prerm/postinst scripts get autogenerated seems messy so I'm wondering if there's another way. Could it list files in /usr/share/python*/dist/$pkg/bin/ at runtime? Or are there some fpm args we can use to install our own bin stubs/symlinks as /usr/bin/whatever? Even if we still need to autogenerate the stubs, at least that way yum/apt would know where they came from, avoid installing other packages with conflicting binaries, etc.

Yes. I've removed the new prerm/postinst scripts and done this the right way now. Thanks!

> - The new fpm_build_virtualenv has a lot of overlap with fpm_build, but does some things differently (e.g., it calls test_package_presence instead of letting the caller do that).

I moved that particular test into fpm_build_virtualenv to make run-build-packages simpler. There was a lot of code duplication there running that test.

> Meanwhile the old fpm_build still has lots of (now unused?) python-specific stuff. I don't know which is less messy (two similar funcs vs. lots of if-else). Splitting might be better but it would be good to keep them congruent.

I left the code in place that builds our own copy of libcloud. That uses the old fpm_build, which is why I didn't remove all the python stuff from that yet. I'm kind of tempted to leave things like this until we retire node manager. Unless you think that would take too long?

Libcloud is the only loose end I'm still not clear on. We often need to fork it, and building arvados-node-manager with virtualenv is going to require a new strategy for that, which I haven't figured out yet.

> Nit/style thing: I find it easier to follow "if ! foo; then ...; fi" than "foo; if $? != 0; then ...; fi". The first form also has an advantage that it works the same way in "set -e" and "set +e" contexts.

Yeah, I cleaned that up in the fpm_build_virtualenv function.

> The deletion of build.list (and its version compatibility maintenance nightmare) sure makes me happy... this is very encouraging. Thanks!

Indeed! I like deleting a lot of crufty code.

#14 - 01/28/2019 10:29 PM - Tom Clegg

This looks much better than the previous postinst approach, but shouldn't we be installing to /usr/bin rather than /usr/local/bin?

```
  # Make sure to install all our package binaries in /usr/local/bin.
```

It would be nice not to let node manager's special needs hold this up, but I'm wondering...

- maybe it's only a small amount of work to give it the virtualenv treatment too (I'm hoping so -- is this a matter of find a way to substitute our own fork for the pypi version while building the virtualenv?), or
- otherwise, it seems like we'd be unable to update node manager's dependencies because we no longer build deb/rpm packages that could satisfy them at runtime, which seems like a precarious situation wrt security fixes and cloud API changes. crunch-dispatch-cloud hasn't even started AWS/GCE drivers yet...

#15 - 01/29/2019 03:18 PM - Tom Clegg

Before this branch, with the arvados-python-client debian package installed, you can

1. use "import arvados" without setting up a virtualenv
2. use "import arvados" in a virtualenv where other python packages are installed (perhaps from setup.py / requirements.txt), but arvados is not installed

(The first case is currently our recommended setup -- If you are logged in to an Arvados VM, the Python SDK should be installed. and if not, "option 1" is to install the deb/rpm package.)

I suppose we need to

- provide a workaround -- perhaps change shebang to "#!/usr/share/python2.7/dist/python-arvados-python-client/bin/python" as in the package-testing script? (this only works for the first case)
- help people migrate away from this setup by installing arvados from pypi into a virtualenv (this should be an easy fix in the second case)
- warn about all this on https://doc.arvados.org/admin/upgrading.html

#16 - 01/30/2019 04:05 PM - Ward Vandewege

- Target version changed from 2019-01-30 Sprint to 2019-02-13 Sprint

#17 - 01/30/2019 04:12 PM - Ward Vandewege

See 6e1f3a9a91a694f3ea547f23f924de299f481902

Tom Clegg wrote:

> This looks much better than the previous postinst approach, but shouldn't we be installing to /usr/bin rather than /usr/local/bin?
> [...]

Yes, corrected.

> It would be nice not to let node manager's special needs hold this up, but I'm wondering...
>
> - maybe it's only a small amount of work to give it the virtualenv treatment too (I'm hoping so -- is this a matter of find a way to substitute our own fork for the pypi version while building the virtualenv?), or
> - otherwise, it seems like we'd be unable to update node manager's dependencies because we no longer build deb/rpm packages that could satisfy them at runtime, which seems like a precarious situation wrt security fixes and cloud API changes. crunch-dispatch-cloud hasn't even started AWS/GCE drivers yet...

arvados-node-manager is already using virtualenv. I removed the code that makes a deb/rpm package from our backported libcloud; if we need a backport for libcloud, it will need to be pushed to pypi and listed as a dependency for arvados-node-manager in setup.py, and then it will get pulled into the arvados-node-manager virtualenv on the next package build.

> Before this branch, with the arvados-python-client debian package installed, you can
> use "import arvados" without setting up a virtualenv
> use "import arvados" in a virtualenv where other python packages are installed (perhaps from setup.py / requirements.txt), but arvados is > not installed
> (The first case is currently our recommended setup -- If you are logged in to an Arvados VM, the Python SDK should be installed. and if > not, "option 1" is to install the deb/rpm package.)

I've reordered the options, since using the deb/rpm package is probably not the default workflow for software developers that want to use the Python SDK.

> I suppose we need to
> provide a workaround -- perhaps change shebang to "#!/usr/share/python2.7/dist/python-arvados-python-client/bin/python" as in the package-testing script? (this only works for the first case)

I have done that.

> help people migrate away from this setup by installing arvados from pypi into a virtualenv (this should be an easy fix in the second case)

I'm not 100% sure what you mean here?

> warn about all this on https://doc.arvados.org/admin/upgrading.html

Done.

#18 - 01/30/2019 04:23 PM - Ward Vandewege

- Story points set to 1.0

#19 - 01/30/2019 06:36 PM - Tom Clegg

Ward Vandewege wrote:

> arvados-node-manager is already using virtualenv. I removed the code that makes a deb/rpm package from our backported libcloud; if we need a backport for libcloud, it will need to be pushed to pypi and listed as a dependency for arvados-node-manager in setup.py, and then it will get pulled into the arvados-node-manager virtualenv on the next package build.

Ah, I see. The only issue is that we used to rely on a deb/rpm-based solution for switching between upstream and fork, so we don't have a Python-native way yet. This seems fine.

#20 - 01/30/2019 07:11 PM - Tom Clegg

Ward Vandewege wrote:

> > help people migrate away from this setup by installing arvados from pypi into a virtualenv (this should be an easy fix in the second case)
>
> I'm not 100% sure what you mean here?

On a shell node with our recommended setup, you could do this ("case 1")

1. don't create a virtualenv
2. write a script that imports arvados

...or this ("case 2")

1. create a virtualenv
2. install some unrelated python package "foobar" in the virtualenv
3. write a script that uses the virtualenv and imports both foobar and arvados packages

In both cases, updating the shell node to a new python-arvados-python-client debian package will break the script. I was just making the point that cases 1 and 2 need different fixes.

Fix for case 1: one of

- create a virtualenv, run "pip install arvados" there, and update shebang (or script invocation) to use the virtualenv (this way is surely better)
- update shebang (or script invocation) to use the virtualenv that ships with the new python-arvados-python-client package (this way might be a quicker emergency fix so it seems nice to mention it)

Fix for case 2:

- run "$virtualenv/bin/pip install arvados" on existing virtualenv.

Comments on the doc updates at 6e1f3a9a9:

- Should we really recommend "git clone; setup.py install" rather than "pip install arvados-python-client"? (Avoiding pip seems to imply there's some reason not to use it, which would mean there's a reason not to use setup.py, requirements.txt, etc. as well.)
- I think the upgrade notes should give specific instructions for bringing broken scripts back to life.
- Perhaps the SDK install page should say that option 1 is the recommended way to make the SDK available for use in your own python programs, while option 3 is the recommended way to make the CLI tools available system-wide.

(It might actually be less confusing if we had two different package names -- "arvados-python-cli-tools" (normally installed via deb/rpm, comes with its own virtualenv under the hood) and "arvados-python-client" (normally installed via pypi into your own virtualenv) -- but we should probably save that discussion for another day.)

#21 - 01/30/2019 09:15 PM - Ward Vandewege

See 7995d73085a0fd459edafa679e85c08c9c2b5605

Tom Clegg wrote:

> Ward Vandewege wrote:
>
> > help people migrate away from this setup by installing arvados from pypi into a virtualenv (this should be an easy fix in the second case)
>
> I'm not 100% sure what you mean here?
>
> On a shell node with our recommended setup, you could do this ("case 1")
>
> 1. don't create a virtualenv
> 2. write a script that imports arvados
>
> ...or this ("case 2")
>
> 1. create a virtualenv
> 2. install some unrelated python package "foobar" in the virtualenv
> 3. write a script that uses the virtualenv and imports both foobar and arvados packages
>
> In both cases, updating the shell node to a new python-arvados-python-client debian package will break the script. I was just making the point that cases 1 and 2 need different fixes.

Case 2 isn't real, because the virtualenv will not pull in the Arvados package that was installed system-wide:

```
~$ dpkg -l |grep arvados
```

```
ii  arvados-src                          1.3.0-1                                 all          The Arvados
source code
ii  libpam-arvados                       1.3.0-1                                 all          PAM module f
or authenticating shell logins using Arvados API tokens
ii  python-arvados-cwl-runner            1.3.0-1                                 all          The Arvados
CWL runner
ii  python-arvados-fuse                  1.3.0-1                                 all          The Keep FUS
E driver
ii  python-arvados-python-client         1.3.0-1                                 all          The Arvados
Python SDK
~$ virtualenv x
Running virtualenv with interpreter /usr/bin/python2
New python executable in /data-sdc/home/wardv/x/bin/python2
Also creating executable in /data-sdc/home/wardv/x/bin/python
Installing setuptools, pkg_resources, pip, wheel...source xdone.
~$ source x/bin/activate
(x) ~$ python
Python 2.7.12 (default, Dec  4 2017, 14:50:18)
[GCC 5.4.0 20160609] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import arvados
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named arvados
>>>
```

> Comments on the doc updates at 6e1f3a9a9:
>
> - Should we really recommend "git clone; setup.py install" rather than "pip install arvados-python-client"? (Avoiding pip seems to imply there's some reason not to use it, which would mean there's a reason not to use setup.py, requirements.txt, etc. as well.)

I have no opinion on this so I agree with you. I have deleted that installation option from our documentation.

- I think the upgrade notes should give specific instructions for bringing broken scripts back to life.

Yes, done.

- Perhaps the SDK install page should say that option 1 is the recommended way to make the SDK available for use in your own python programs, while option 3 is the recommended way to make the CLI tools available system-wide.

Yes, also done.

> (It might actually be less confusing if we had two different package names -- "arvados-python-cli-tools" (normally installed via deb/rpm, comes with its own virtualenv under the hood) and "arvados-python-client" (normally installed via pypi into your own virtualenv) -- but we should probably save that discussion for another day.)

Agree, and let's postpone that like you say :)

#22 - 01/31/2019 06:48 PM - Tom Clegg

Ward Vandewege wrote:

> Case 2 isn't real, because the virtualenv will not pull in the Arvados package that was installed system-wide:

Ah, that's a relief.

Doc updates LGTM.

#23 - 02/04/2019 03:45 PM - Ward Vandewege

- Status changed from In Progress to Resolved

Applied in changeset arvados|84b30d024ee757e32af7e05d00bf4324513c388c.

#24 - 03/01/2019 07:33 PM - Tom Morris

- Release set to 15

#25 - 06/05/2019 05:35 PM - Ward Vandewege

1fd7b7089dd0027d36e46c2e1fbb44aee4cc1e66 on branch 9945-crunch-job-python-path is ready for review

#26 - 06/05/2019 05:46 PM - Tom Clegg

There's another use of python ("Using Arvados SDK version...") -- does this also need the same change?

Another thought -- would it be better to prepend the virtualenv bin dir to PATH in the run script (see https://doc.arvados.org/install/install-crunch-dispatch.html)?

#27 - 06/05/2019 05:48 PM - Tom Clegg

Tom Clegg wrote:

> There's another use of python ("Using Arvados SDK version...") -- does this also need the same change?

This looks like an "ancient API server compatibility" thing so probably not.

> Another thought -- would it be better to prepend the virtualenv bin dir to PATH in the run script (see https://doc.arvados.org/install/install-crunch-dispatch.html)?

If there's only one use, let's stick to the current way.

#28 - 06/05/2019 05:51 PM - Ward Vandewege

Tom Clegg wrote:

> There's another use of python ("Using Arvados SDK version...") -- does this also need the same change?

No; that other use of python executes inside the docker container. Our standard 1.4 jobs image already makes sure that running 'python' just works.