Arvados - Feature #7328

[Crunch] [UX] Standard excepthook to help debug job problems

09/14/2015 03:36 PM - Bryan Cosca

Status:	New	**Start date:**	09/14/2015
Priority:	Normal	**Due date:**	
Assigned To:		**% Done:**	0%
Category:	Crunch	**Estimated time:**	0.00 hour
Target version:	Arvados Future Sprints		

Description

Original report

for example, a simple "ls" in the directory that they are currently in or looking into would help. Working with collections is a new concept, so users can easily run into file not found issues. Also, if they are piping commands and lose track half way, it would be great to get more insight into what is being created/deleted/etc.

Implementation

Overview: Add an excepthook function to the Python SDK that saves the contents of the $TASK_WORK directory as the task's output, and mark it failed. The SDK automatically installs this function as sys.excepthook when it's running under a job that has a debugging runtime constraint set.

- Add a new function to the arvados.crunch Python SDK module.
    - It will be installed as sys.excepthook and must take the same signature: (exc_type, exc_value, traceback)
    - It creates a new collection from the contents of the $TASK_WORK directory, and updates arvados.current_task() so that the task's output is the manifest of that collection, and success is false.
        - run-command already has most of the code to do this.
    - No matter what happens in the above step(s), the last thing it does is always call sys.__excepthook__ to get the usual exception handling behavior. Ensure this happens using a try/finally block.
- Add this code or the functional equivalent to __init__.py:

```
if os.environ.get('TASK_UUID'):
    try:
        _want_debughook = current_job()['runtime_constraints']['debug_exceptions']
    except (errors.ApiError, KeyError):
        # The job doesn't exist or doesn't define the constraint.
        pass
    else:
        if _want_debughook:
            sys.excepthook = arvados.crunch.your_new_function
        del _want_debughook
```

- Add documentation for the new runtime constraint to the Job schema page.

History

#1 - 09/14/2015 03:52 PM - Bryan Cosca

I realize its probably hard (if not impossible) to keep track of files that are getting piped. so it would be great if we could log the paths of files when they get deleted/created, as well as a live view of what files are getting written to keep, what files are getting used by keep, that would be great.

#2 - 09/16/2015 02:57 PM - Brett Smith

Bryan,

I think I understand the basics of what you're getting at, but a little more detail would help. It sounds like this suggestion is specifically for the FUSE mount. Is that right? And the idea is that, if the user runs a command inside FUSE that doesn't find files, FUSE should show them what files are available in the collection?

#3 - 09/16/2015 03:05 PM - Bryan Cosca

Brett Smith wrote:

> Bryan,
>
> I think I understand the basics of what you're getting at, but a little more detail would help. It sounds like this suggestion is specifically for the FUSE mount. Is that right? And the idea is that, if the user runs a command inside FUSE that doesn't find files, FUSE should show them what files are available in the collection?

Yes, that's one of my ideas.

In my comment, I would also like a logging mechanism of showing what files get created within the temporary directory or any directory before getting uploaded into keep. So for example, if you have a command: grep 'foo' file.txt > file2.txt, I would want to know if file2.txt got created.

#4 - 09/30/2015 02:34 PM - Brett Smith

Unfortunately, Unix doesn't let us do this.

arv-mount is running in one process, handling all kinds of requests for files in Keep: listing them, opening them, reading them, writing them, closing them. Meanwhile, the process sending all these requests—whether that's your shell, commands in your shell like ls or grep, a Python script that's manipulating files in a mounted directory, whatever—is completely separate.

In order to make suggestions or log information like you describe, arv-mount would have to write to the second process' stdout or stderr. This might theoretically be doable in some cases, but it would be a very dirty hack and generally a bad idea: it might introduce corrupted output or even bugs to the second process. And in a lot of other cases, arv-mount won't even have permission to write to those file descriptors at all.

We already have some mechanisms to log to files in arv-mount. We could clean up the logging to make it more consistent, and maybe turn it on by default for very basic information about changes to collections. Would that still be interesting?

#5 - 09/30/2015 08:14 PM - Bryan Cosca

> We already have some mechanisms to log to files in arv-mount. We could clean up the logging to make it more consistent, and maybe turn it on by default for very basic information about changes to collections. Would that still be interesting?

Hmm... I don't think so. Generally, my scripts don't change collections. They just create a new one at the end and if that doesn't ahve the file I needed then I'll look in the logs.

#6 - 11/11/2015 03:55 PM - Bryan Cosca

- Target version set to Arvados Future Sprints

#7 - 11/17/2015 02:53 PM - Brett Smith

Notes from discussion: This mostly comes up in the context of jobs and their temporary directory. It happens in various ways that a file the user is expecting to be in the directory (e.g., output from a previous subprocess) isn't there.

Idea: When a task fails, list the contents of its temporary directory in the logs. This is relatively cheap, and can still give the user reasonably good insight about what happened.

#8 - 11/17/2015 02:57 PM - Brett Smith

- Subject changed from [UX] If a user runs into a file not found problem, help them to [Crunch] [UX] Help user debug missing file problems in running jobs

- Category set to Crunch

#9 - 11/17/2015 06:58 PM - Tom Clegg

My main reservation is that a temp dir is somewhat likely to contain extracted git repositories, tarballs, and so on, which could overwhelm the useful parts of the log.

Perhaps we can make it so it can be enabled/disabled from the outside (via job record) or from the inside (via Python SDK)?

Perhaps the entire feature can be in the Python SDK -- arvados.crunch.list_my_temp_dir_contents_with_the_stack_trace_if_this_program_crashes() or something a bit more succinct?

#10 - 12/01/2015 07:52 PM - Brett Smith

- Story points set to 1.0

#11 - 12/02/2015 05:36 PM - Brett Smith

- Subject changed from [Crunch] [UX] Help user debug missing file problems in running jobs to [Crunch] [UX] Standard excepthook to help debug job problems

- Description updated

#12 - 12/02/2015 05:42 PM - Brett Smith

- Description updated

#13 - 12/02/2015 05:43 PM - Brett Smith

- Description updated

#14 - 12/02/2015 07:32 PM - Tom Clegg

Make sure that, in the event the excepthook itself crashes, we still log the stack trace from the original exception.

I think we should try writing to Keep even if we get arvados.errors.KeepRequestError. For example, "block not found" and "signature expired" don't mean we can't write to Keep. I haven't thought of a really good example of "can write block A but not block B" other than "timeout before giving up was a little too short"... but trying anyway seems mostly harmless.

#15 - 12/02/2015 07:46 PM - Brett Smith

- Description updated

Updates per Tom's comments.