Pulp - Story #5403

As a plugin writer, I have a Serializer I can use for Content creation via upload that accepts either file or an existing Artifact but not both

09/04/2019 05:03 PM - bmbouter

Status:	CLOSED - CURRENTRELEASE	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	mdellweg	**% Done:**	100%
Category:		**Estimated time:**	0:00 hour
Sprint/Milestone:	3.0.0		
Platform Release:		**Tags:**	
Groomed:	No	**Sprint:**	Sprint 59
Sprint Candidate:	No	**Quarter:**	

Description

Motivation

Plugin writers are writing the same serializer over and over. The situation is this:

- The user wants to create a Content unit via upload.
- Sometimes they want to uplod the file directly in that 1 call (for small files).
- Sometimes they want to use the chunked upload API to create the Artifact and then pass a reference to that Artifact for Content unit creation
- Sometimes they want to associate that ContentUnit with a specific repository, other times they don't and just want to create the Content unit

Usage

Plugin writers would be able to use this Serializer for the create() method of their Content unit specifically.

User Impact

1. This would effectively make all "upload APIs" live at the POST of the content unit. This is more RESTful.

2. This would remove the user's ability to specify the metadata for that content unit. This breaks no use case that I know of, as a user would have to hand over the binary data either way and Pulp "just handles it".

Subtasks:

Task # 5453: Use new upload serializer in pulp_rpm	**CLOSED - CURRENTR**
Task # 5462: Use new upload serializer in pulp_ansible	**CLOSED - CURRENTR**
Task # 5463: Use new upload serializer in pulp_docker	**CLOSED - WONTFIX**
Task # 5464: Use new upload serializer in pulp_python	**MODIFIED**
Task # 5487: Use new upload serializer in pulp_deb	**CLOSED - CURRENTR**

Related issues:

Blocks Ansible Plugin - Issue #5281: Ansible collection upload URL missing /p...	**NEW**

Associated revisions

Revision e5511dc2 - 09/18/2019 09:50 PM - Fabricio Aguiar

add new upload serializer

Required PR: https://github.com/pulp/pulpcore-plugin/pull/122 closes #5403 https://pulp.plan.io/issues/5403

Revision e7a03931 - 09/20/2019 05:41 PM - mdellweg

Add context to the general_create task

re #5403 https://pulp.plan.io/issues/5403

Required PR: https://github.com/pulp/pulpcore-plugin/pull/122

Revision 4a64190e - 09/20/2019 05:42 PM - mdellweg

File upload

Required PR: https://github.com/pulp/pulpcore-plugin/pull/122

re #5403 https://pulp.plan.io/issues/5403

Revision 2ae13eb7 - 09/23/2019 06:18 PM - mdellweg

Add file upload functionality to content types

re #5403 https://pulp.plan.io/issues/5403 fixes #5487 https://pulp.plan.io/issues/5487 fixes #5371 https://pulp.plan.io/issues/5371 fixes #5376 https://pulp.plan.io/issues/5376 fixes #5379 https://pulp.plan.io/issues/5379

Required PR: https://github.com/pulp/pulpcore-plugin/pull/122

Revision 38ca5ef9 - 09/23/2019 09:18 PM - bmbouter

Add upload docs

re #5403 https://pulp.plan.io/issues/5403

History

#1 - 09/05/2019 02:28 PM - mdellweg

I took one shot at this topic (pun intended):

https://github.com/ATIX-AG/pulp_deb/tree/upload_create

Still missing are tests and the ability to do the association with a repository.

I believe, creating a new repo version must be done in a task, which raises the question, whether the create should always trigger a task, or the endpoint might be able to return different objects based on conditions.

#2 - 09/09/2019 11:53 AM - mdellweg

- Assignee set to mdellweg

More questions that came up.
1. Do the base content serializers belog into the pulpcore repository, or should they rather be in the pulpcore-plugin?
2. Do the names _artifact and _relative_path (with leading underscore) make sense? In plugins using those, we take a lot of effort to rename them back.

#3 - 09/09/2019 11:59 AM - ipanova@redhat.com

mdellweg wrote:

> I took one shot at this topic (pun intended):
>
> https://github.com/ATIX-AG/pulp_deb/tree/upload_create
>
> Still missing are tests and the ability to do the association with a repository.
>
> I believe, creating a new repo version must be done in a task, which raises the question, whether the create should always trigger a task, or the endpoint might be able to return different objects based on conditions.

our current one shot uploads optionally add created content unit to the repo. So we should lock on artifact and optionally on the repo, if specified, new repo version will be created within the task. tldr - endpoint should always return a task.

#4 - 09/09/2019 07:14 PM - bmbouter

mdellweg wrote:

> More questions that came up.
> 1. Do the base content serializers belog into the pulpcore repository, or should they rather be in the pulpcore-plugin?

I think this serializer can start to live in pulpcore-plugin until we have a reason to move it to pulpcore. I think during the switching of machinery onto it we'll find out for sure.

> 2. Do the names _artifact and _relative_path (with leading underscore) make sense? In plugins using those, we take a lot of effort to rename them back.

I agree the plugins take extra effort to handle this "convention" we put in place earlier in pulp3's development. At this point it's feeling out of place for a variety of reasons. First of which is that these field names strangely are becoming user facing as an attribute they are settings. Second because of the issues these field names create in the bindings. For these reasons, I'm in favor of switching the _artifact to be _artifact and _relative_path to be relative_path.

#5 - 09/09/2019 07:16 PM - bmbouter

mdellweg I think bringing up the switching of _artifacts to artifact and _relative_path to relative_path on pulp-dev's mailing list would be productive to bring more visibility to this issue.

#6 - 09/10/2019 08:10 PM - daviddavis

- Blocks Issue #5281: Ansible collection upload URL missing /pulp/api/v3/ added

#7 - 09/12/2019 10:08 PM - daviddavis

- Sprint/Milestone set to 71

#8 - 09/13/2019 05:42 PM - bmbouter

- Sprint/Milestone changed from 71 to 3.0.0

#9 - 09/16/2019 07:06 AM - dalley

The ideas expressed here are all good, but I have some concerns about the mechanics of this proposal.

(I'm going to combine some details from the email thread with this quote, so it's not a direct quote)

```
1. Make all uploads of a specific content type live at POST /pulp/api/v3/content/<plugin_name>/<content_name>/
.  This would effectively make all "upload APIs" live at the POST of the content unit. This is more RESTful.
2. Have it accept either binary data (to create an Artifact from before the Content unit) OR a reference to an
 existing Artifact (allowing the chunked upload API to be used) but not both. This would remove the user's abi
lity to specify the metadata for that content unit. This breaks no use case that I know of, as a user would ha
ve to hand over the binary data either way and Pulp "just handles it".
```

I'm on board with point 1 in theory. It would definitely be nice to have it all in one place. I have a couple of concerns about point 2. Possibly they can be cleared up by making the proposal more specific since it is currently very light on details.

I'm unclear on whether the proposed API would return a task or not. Ina and Matthias have pointed out that it is necessary if we want to add to a new repository version at the same time as upload, but it wasn't really acknowledged by anyone else and it's not in the current prototype or in the description or in the email thread. I didn't really get the impression that it was part of the design proposal, but maybe I missed some IRC discussion.

I **do** think that we should return a task from our upload endpoint regardless of what the endpoint is, but mostly for a different reason. Basically, we probably should not be parsing user-provided arbitrary files synchronously inside of the web server [0]. We are and have been doing that in some of our prototype upload implementations but I definitely don't think that should continue into our GA.

A) security implications?

- sure, our users are sysadmins and we have to trust them somewhat, but still, "parsing user-provided arbitrary files inside a webserver" is the sort of sentence that should probably give us pause

B) reliability implications?

- let's say createrepo_c (or some other native library for one of the many many plugins that we may have in the future) throws a segfault -- what happens? is it just a 500 server error or can it do worse things?
- there's no way to manually cancel it if it's taking too long or using too many resources. but maybe the webserver can be configured to deal with that -- I have no experience there.

C) API implications?

- parsing a user-provided arbitrary file is at least somewhat likely to fail occasionally, and we should make sure we can provide more useful information than "500 Server Error" when that happens

So I think we definitely do want it to return a task and it would be good to note this in the description. Let me know if I'm off-base with any of this.

I would also like some clarification on "specifies that it would "remove the user's ability to specify the metadata for that content unit", since it wasn't specified whether it would rule that out permanently for all content types. I don't think that is what you meant, but I can imagine that surely there's some kind of artifact-less content type out there that we might want to be able to just create with a single request w/ metadata.

Lastly, I'm not sure whether we would be making things more RESTful by having one HTTP verb for an endpoint take completely different parameters than the others, vs having a separate endpoint with different parameters. I don't personally care very much about that, I just want to point it out.

[0] I don't really know whether "inside of the web server" is strictly accurate since I have no idea how WSGI works, but I hope the point still stands

#10 - 09/16/2019 10:20 AM - mdellweg

Thanks for writing this up.
You are right, the pullrequest is a little bit outdated. I will try to update it soon.
I agree on the fact, that this disguised upload should always trigger an async task, and i will try to address your concern about parsing in the webserver frontend.
As for the metadata: It is still possible to pass along metadata to your new contentunit along with the artifact. But the question remains, whether the new serializer should *replace* the SingleContentArtifactSerializer.
The other thing, i am not sure about, is whether the uploaded file *must* be turned into artifact before deferring the rest of the action to the task.

#11 - 09/16/2019 04:58 PM - bmbouter

I agree we should return a Task in all cases and have its locking include the resources as needed. You can see this some in the pulp_ansible implementation of this here for example.

In terms of locks, I was thinking we should always lock on the Artifact that is being associated. We should lock on the repository when it's specified by the user to be associated with that repo.

In terms of when to create the Artifact, I imagined we would create the Artifact in the viewset if we are receiving the binary data, and if a user specifies an existing one, then it already exists. The task implementation would have artifact_pk as its parameter so in all cases it's working from an already-saved Artifact.

mdellweg I'm not entirely sure on if it should replace SingleContentArtifactSerializer. I suspect it should not because the parameters you upload to that and the way you serializer a single content Artifact serve different purposes. I'm not 100% sure on this part though.

#12 - 09/17/2019 04:36 PM - daviddavis

- Sprint set to Sprint 59

Per our triage today, adding this to the sprint. Also, any subtasks involving the use of the serializer in plugins are good to go too.

#13 - 09/17/2019 05:48 PM - mdellweg

- Status changed from NEW to POST

#14 - 09/19/2019 10:01 AM - mdellweg

https://github.com/pulp/pulpcore/pull/305

This PR allows to pass context on to the general_create, which is needed for the ansible plugin, i have been told.

#15 - 09/20/2019 01:41 PM - ipanova@redhat.com

https://github.com/pulp/pulpcore-plugin/pull/122

#16 - 09/20/2019 04:09 PM - dalley

bmbouter wrote:

> I agree the plugins take extra effort to handle this "convention" we put in place earlier in pulp3's development. At this point it's feeling out of place for a variety of reasons. First of which is that these field names strangely are becoming user facing as an attribute they are settings. Second because of the issues these field names create in the bindings. For these reasons, I'm in favor of switching the _artifact to be _artifact and _relative_path to be relative_path.

The problems that this causes for the bindings -- do those not apply to _href, which I think still has to be specified in some places (unless I'm misremembering)?

Of course, _href isn't backed by a real database field.

#17 - 09/20/2019 04:26 PM - mdellweg

dalley wrote:

> The problems that this causes for the bindings -- do those not apply to _href, which I think still has to be specified in some places (unless I'm misremembering)?
>
> Of course, _href isn't backed by a real database field.

Yes, _href might also be a candidate. The difference is, the artifact and relative_path fields are only affecting a Serializer that a conflicting plugin can simply decide not to use. _href is affecting every single serializer, so i would be more cautious.
Anyway, this topic might be better at eiher
https://pulp.plan.io/issues/5428
or

https://pulp.plan.io/issues/5457

#18 - 09/20/2019 08:01 PM - fao89

- pulpcore-plugin - https://github.com/pulp/pulpcore-plugin/pull/122
- pulp_file - https://github.com/pulp/pulp_file/pull/276
- plugin_template - https://github.com/pulp/plugin_template/pull/107
- pulp_ansible - #5462 - https://github.com/pulp/pulp_ansible/pull/213
- pulp_deb - #5487 - https://github.com/pulp/pulp_deb/pull/115
- pulp_rpm - #5453 - https://github.com/pulp/pulp_rpm/pull/1446
- pulp_python - #5464 - https://github.com/pulp/pulp_python/pull/257

#19 - 09/26/2019 07:59 PM - mdellweg

- Status changed from POST to MODIFIED

- % Done changed from 20 to 100

Applied in changeset commit:pulpcore-plugin|fa9404ba980b649758953f91bce2dafc9864801d.

#20 - 12/13/2019 06:10 PM - bmbouter

- Status changed from MODIFIED to CLOSED - CURRENTRELEASE