Pulp - Story #6134

[EPIC] Pulp import/export

02/11/2020 10:13 PM - daviddavis

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

Description

An epic for the next batch of importer/exporter stories for Katello.

After an import, the destination should have a repo version that is exactly the same as the exported repo version.

Collaboration on the design is happening here: https://hackmd.io/@ggainey/HyfXU_648

Subtasks:

Story # 6135: As a user, I can export a set of repository versions to a file, and have ...	**CLOSED - CURRENTR**
Issue # 6457: PulpExporter serializer should display hrefs for repositories[]	**CLOSED - CURRENTR**
Issue # 6466: PulpExport needs to cast() repositories it is exporting	**CLOSED - NOTABUG**
Story # 6136: As a user, I will receive only the new artifacts if I export using a Pulp...	**CLOSED - CURRENTR**
Story # 6137: As a user, I can import an export and have its contents be added to exist...	**CLOSED - CURRENTR**
Story # 6138: As a user, I can upload an incremental export and specify a repository ve...	**CLOSED - WONTFIX**
Story # 6328: As a user, I can create/read/update/delete PulpExporters	**CLOSED - CURRENTR**
Story # 6329: As a user, I can create/read/update/delete PulpImporters	**CLOSED - CURRENTR**
Task # 6364: Add docs for performing imports/exports	**CLOSED - COMPLETE**
Story # 6436: As a plugin writer, I can write custom resources to customize what conten...	**CLOSED - CURRENTR**
Task # 6454: Document that pulp import/export is provided as a tech preview	**CLOSED - CURRENTR**
Story # 6456: As a user, I can import/export Pulp content while using S3 or other stora...	**CLOSED - CURRENTR**
Story # 6472: Add model-resource for pulp_file	**CLOSED - CURRENTR**
Story # 6473: Add model-resource for pulp_rpm	**CLOSED - CURRENTR**
Story # 6483: As a user, I can import content that might already exist in the database	**CLOSED - CURRENTR**
Task # 6484: Have the import code import repo versions using child tasks	**CLOSED - COMPLETE**
Issue # 6514: Rehome QueryModelResource to pulpcore.plugin	**CLOSED - CURRENTR**
Task # 6515: Investigate/reduce the fields being exported	**CLOSED - CURRENTR**
Task # 6532: Check repo type during import	**CLOSED - COMPLETE**
Task # 6539: Add functional tests for PulpExport/PulpExporter	**CLOSED - COMPLETE**
Task # 6541: Add more/any reporting to the export process	**CLOSED - CURRENTR**
Task # 6542: Add tests for importing	**CLOSED - COMPLETE**
Issue # 6544: export needs to validate and persist passed-in params	**CLOSED - CURRENTR**
Issue # 6555: Investigate/decide whether "set last_export to null explicitly before all...	**CLOSED - CURRENTR**
Issue # 6556: Export requires exporter-UUID in bindings instead of HREF - can we fix this?	**CLOSED - CURRENTR**
Story # 6558: As a user, I receive an error message if I try to import an export from a...	**CLOSED - CURRENTR**
Issue # 6564: Export filename for pulp exports has dupe slashes	**CLOSED - CURRENTR**
Story # 6566: As a User, I can create an Exporter to export a specific set of Repositor...	**CLOSED - CURRENTR**
Story # 6572: As a User/Importer, I can know the versions of pulpcore/plugins that were...	**CLOSED - CURRENTR**
Story # 6736: As a user, I can export into a series of files of a particular size	**CLOSED - CURRENTR**
Story # 6737: As a user, I can import a split export	**CLOSED - CURRENTR**
Story # 6739: As a user, I can export and import kickstart trees	**CLOSED - CURRENTR**

Story # 6763: As a User, I can create an Exporter to export a specific set of Repositor...	**CLOSED - CURRENTR**
Task # 6807: Teach import/export to use 'natural keys' instead of pulp_id/uuids	**CLOSED - CURRENTR**
Issue # 6815: Not all advisory models are defined for import/export	**CLOSED - CURRENTR**
Issue # 6919: Import/Export docs page typos	**CLOSED - CURRENTR**
Task # 6936: Rehome Content ModelResource classes to use new BaseContentResource	**CLOSED - CURRENTR**
Task # 6937: Rehome Content ModelResource classes to use new BaseContentResource	**CLOSED - CURRENTR**
Issue # 7137: pulp_rpm needs tests for import/export	**CLOSED - DUPLICAT**
Task # 7221: Add 'toc' info to the core export task	**CLOSED - CURRENTR**
Issue # 7246: Failed pulp exports leave behind an export file	**CLOSED - CURRENTR**
Story # 7252: As a plugin writer, I have a way to map Content to Repositories in Pulp e...	**CLOSED - CURRENTR**
Task # 7277: Have QueryModelResource exclude pulp_id, pulp_created, and pulp_last_updat...	**CLOSED - CURRENTR**
Task # 7296: Update documentation to mention/use BaseContentResource	**CLOSED - CURRENTR**
Issue # 7403: PulpExport full= can fail	**CLOSED - CURRENTR**
Test # 7422: Add tests for export/import of kickstarts	**MODIFIED**
Related issues:	
Related to Pulp - Story #5096: [epic] As a user, I can export the content of ...	**CLOSED - DUPLICATE**

History

#1 - 02/11/2020 10:13 PM - daviddavis

- Subject changed from Importers/Exporters to [EPIC] Importers/Exporters

- Description updated

#2 - 02/14/2020 05:57 PM - daviddavis

- Description updated

#3 - 02/14/2020 06:25 PM - ggainey

Notes from initial design meeting 2020-02-14:

Pulp3 exporters explanation

- only for filesystem
  - exports to disk somewhere
- talked about rsync exporter
  - same but uses rsync to elsewhere
- not for re-importing, for external consumption
- file repo
  - can't export version - list-of-content, but no metadata
  - current exporters only know how to export publications (which is what adds metadata)
  - master/detail (pulpcore/plugin) used for exporters
- maybe use publish first?
  - publish == create-a-publication
  - publish code doesn't exist in pulpcore? - code is in plugins
- not all plugins differentiate between version/publication
- FileSystemVersion vs ...PublicationExporter
  - PubExp includes metadata we don't need
  - FS doesn't have data-from-db only from filesystem
- Therefore - need a third-kind of Exporter?
  - RepositoryVersionExporter

two approaches

- Master baseclass to handle grunt filesystem work
  - Plugins extend to expose/provide the API
  - this is The Pulp3 Way
  - could have most of the 'heavy lifting' happening in Master, even with API controlled/exposed by Detail
- expose API at pulpcore level
  - plugins only define model info
  - see https://pulp.plan.io/issues/5096

django import/export

- just handles models
- however, export needs two pieces

    - dataset in db for all content-items in repo-version
    - artifacts
    - example:
        - RPM - errata from DB, and the RPMs themselves from filesystem
- need not just content-units but also relationships
    - relationship between content and artifact
    - cross-content-unit relationships
    - certain content-types have relationships to other content types
    - can we rely on uuids-as-keys for db export/import work?
    - or, do we need to export by 'natural key' (eg, NEVRA or NSVCA or errata-name etc)

general notes

- will prob want to apply to existing FileSystemExporter (once we know what we're doing)
- diff-exports - export diff-metadata or full-metadata?
    - prob full - puts onus of set-theory on importer, and gives enough info to make that possible
- 3 questions to be emailed to list:
    - master/detail vs core
    - natural key or uuid?
    - incremental export
        - dump 'all' db-metadata? (importer does set-theory to handle added/updated/removed)
        - dump just the differences? (exporter does set theory)
        - always dump just the incremental artifacts

AIs:

- ggainey to add notes to epic
- ddavis to send note to pulp-dev w/questions and pointer

#4 - 02/24/2020 09:47 PM - ggainey

Notes from design discussion 2020-02-21

Attendees

- ggainey
- daviddavis

Use django import/export as basis

- django import/export - https://django-import-export.readthedocs.io/en/latest/getting_started.html
- old issue RE django-import-export : https://pulp.plan.io/issues/5096

Ownership and workflow

- core starts from repo-version, which 'knows about' all the artifacts - so core can be responsible for packaging up the physical on-disk entities
- plugins will need ModelResources to define how to export/import the database metadata that matches a repo-version
- who owns RAR-ing resulting exported filefile? katello? us? P^3I?
    - katello/caller would own this part of the process (and the re-creating at import as well)
- what if plugin can't handle export-import?
    - core needs to be able to call a specific per-repo-version-type method to export/import
    - on error/exception/NotImplemented, return error to the caller
- need to think about pre-export-sanity-checks (eg, disk space)

What does the API look like?

- Possible /import /export endpoints
    - just a repository-version
    - a list of repository-versions
    - latest for a repository * what about "everything* for a repo (all versions/distributions/publications)
        - is this a real use case?
- probable first cut is "you specify a specific repo-version" (export) and "specify a repository" (import)
- need to know/talk about 'natural' keys for things
    - if downstream can be relied on to never create content-artifacts 'on its own' , can we rely on uuid-to-natural-key being "the same" between up and downstream?

Artifact transfers

- how to insure a given file/artifact only gets transferred once in the presence of multi-version-apis?
- start with single, but make sure we don't architect-out multi-version/content-once approach next
- exporting distributions - relative-path in pulp - needs to be exported
- publications - export? or publish downstream? *pulpcore doesn't know about publishing

edge cases

- There will be several/many - ponder on workflow/complicated plugins and start thinking about general answers

Notes from design discussion 2020-02-24

Attendees

- ggainey
- daviddavis

Django import/export discussion

- how does it handle complicated FK relationships, esp at import-time?
- how is import-order defined?
- Need to look at real-world cases and prototype

Design doc draft

- collaborating in Pulp3 Import/Export design doc in team gdrive

#5 - 02/27/2020 09:35 PM - ggainey

Notes from design discussion 2020-02-27

daviddavis tried out import/export

- (JSON example)[https://gist.github.com/daviddavis/f35ec8f0225585e4f137cf4e3aad9cc2]
- foreign-key: exports the FKID
- many-to-many: string of comma-separated-list of FKIDs (ie, "associated-foos": "foo-id-1,foo-id-2,foo-id-3")
- as long as FKs are uuids, and not an internal-to-0this-db-only entity, should work for us

katello export use-cases

- katello needs us to export multiple-repo-versions in order to export All The Things in a Content View at once
- downstream not-just-like upstream
  - repos have diff names, for example (pulp-uuids all in same content-view being imported)
- how do we handle mapping export-repo-version to appropriate destination-repo?
  - import-api **must** allow a destination-repo to be specified, for all repo-versions in the import-file
  - requires a way to have/generate a mapping for each repo-version in the export
  - export-side shouldn't have to 'know' this in advance - so must be handled on import-side
- katello's usecase has enough info to fill in this info
- for non-katello pulp user, must have a way for the user to define this mapping and hand it to the import side
- regardless, import/ sides needs two params: tarfile-path, mapping

import-side discussion

- dryrun could generate a mapping of all repo-versions being imported into repos with the same names as they had on the export-side.
  - User can manipulate/update/change that mapping to their hearts' content, then use it to do the 'real' import
- import needs a dry run that **accepts** a mapping as a test vehicle and spits out what it would do with that mapping,
  - for sanity- and error-checking prior to actual-import errors (bad json, dest-repo doesn't exist, whatever)

triple-disk-problem

- artifacts/content can be taking up disk space three times
  - in pulp
  - in temp-export-dir
  - in tarfile
- user needing three times current pulp-usage to do exports is Painful
- how do we avoid this?
  - linked streams?
  - holding entire tarfile in memory at once is a nonstarter

naming discussion

- RE this question:

  Currently, Pulp has the concept of Exporters (filesystem, rsync, etc) which are implemented as Master/Detail. This was done to accommodate the fact that some plugins will need to export publications while others might export repository versions. Do we divorce the concept of import/export and Exporters? Or bring the Exporters inline with import/export by looking into having core handle Exporters?

- Exporter is a different , **pulp-to-end-user** functionality, import/export is **pulp-to-pulp**. Is there a better pairing than import/export to avoid this name clash? What if we rename current-exporter to Publisher?

- email sent to ongoing importer/exporter discussion on pulp-dev@

#6 - 02/28/2020 08:22 PM - daviddavis

- Related to Story #5096: [epic] As a user, I can export the content of a RepositoryVersion from one Pulp3 system and import on an air gapped Pulp3 system added

#7 - 02/28/2020 09:21 PM - ggainey

Notes from 2020-02-28 design doc

attendees: ggainey, daviddavis,bmbouters, dkliban

- procedural issue - need to extend more invites - ggainey and daviddavis strongly concur
- how are users going to use the functionality
- what's the workflow?
- how do we support rollback (upstream had five versions, exported one, now want to rollback)
- what about just exporting each of the versions- diffs
- need to import version in the right order
  - if pulp controls multi-version-export, can control multi-version-import - can make sure versions are imported in correct order
- close 5096 (but make sure we've captured all the knowledge there)
- testing of django-import-export
  - csv and json both supported
  - FK/manytomany kind of simple
- what's the easiest impact on the pluginwriter
  - can we dynamically create modelresources if needed?
- how many steps does the user take?
- exporter can't know anything about the downstream
- two types of export
  - repos that pulp can "just sync"
  - trying to keep a replica of a pulp instance
  - some repo-types have no at-rest state
- import use-cases
  - repo doesn't exist
  - repo exists but is the wrong type
  - repo exists and is the right type
  - what if there's a malicious repo on the importer?
    - this has to be fixed by ACLs/RBAC/authorization
- do we require the user to create an ExportEntity?
  - would need to if we care about history
  - "export everything since the last export"
  - need to be able to CRUDL these
  - s that new code? or do we somehow get this 'for free'?
- if we have Exporter object, then this and current-exporters *are* the same kind-of thing
  - PulpExporter and PulpImporter
- dry-run is *really* important, and probably moreso on the Importer
- track sha256 of export-file and at import-time
- Export model relies on a History model
- export-history actually matters to *all* Exporters
  - need to make this stuff happen for all Exporters
- lets make sure we can get contributions from Brno
  - have at least one meeting in AM EST
- ggainey to massage gdoc with output by Monday

#8 - 03/03/2020 11:27 PM - ggainey

2020-03-02

attendees: ggainey davis bmbouters dkliban ttereshc ipanova

- ContentModel vs Just-a-Model
- create an exporter, then call/invoke an exporter
- can thus get incrementals for free
- can override and ask "do the last thing over again"
- restoring last-exported-version - can get complicated?
- need to add labels (what did this mean? gg)

one tarball per repo?

- no - doesn't handle the content-deduplicate issue or multi-file-issue for katello

publications/distributions: do we really need to do this?

- p3 creates publications, and then creates distributions that points to that publication
- leave for 'later' and a 'real' use case
- secure environments - what about when you sign the metadata and the downstream can't do the signing
- some plugins don't have publications (see live-api plugins)

incrementals question:

- if we can import into any repo, do we support additive? or mirror? how does this interact with "import into *this* base-version?"
- export/import wants to leave the 'downstream' as an 'exact copy' of whatever was exported
- can we 'rely on' downstream version-numbers matching the upstream-version-numbers?
- katello doesn't care
- how can we guarantee content is the same, in the presence of incrementals?
- at the end of the day, user has to know?

Publishing design-doc for comments

- write the design
- export as PDF
- attach to epic
- write subtasks referring directly to pdf/hackmd
- wiki page in redmine?

2020-03-03

attendees: ggainey davis bmbouters dkliban ttereshc ipanova

importers

- what do they look like?
- how they decide "this is an incremental"
  - does it even need to care?
  - has full-db-metadata always
- needs a mapping of my-repo to upstream-repo
- can't find repo?
  - create or error?
  - what about the empty-downstream-case?
  - dry-run to catch errors
  - is there any missing data?
    - how are we going to handle errors?
    - what about plugin-extra-fields? (eg subrepo)
  - can we add this later?
    - sounds like a phase-2 or -3 thing
    - get help from community to add this
  - validation first
    - if anything is wrong, fail the entire operation immediately
    - will need some bad-export-tests
  - validation/import needs to happen
    - needs to lock all affected repos at start?
    - validate-and-lock on repo at a time?
    - is an import an atomic operation or not?
    - what happens if you reimport something you already imported?
      - use stage-api to make it possible to re-import
    - what are some unfixable errors?
      - artifact from a prev export was deleted from downstream - continue, or fail-the-repo?
        - switchable mode? - current default is safety-first, option to report-and-continue
        - question is do we create a repo-version if we have an artifact failure
        - so switch on "create a repo-version on a failure, or not" (like sync)
          - switch is on inside-repo problem, not on entire-version
- how do we define import-order
  - pre-import-per-row hooks?
  - specify the models in an ordered list?
  - we may need to look at how the code works

API and http verbs

- POST - create
- PATCH - update
- GET
- add verb to API for "do the export"?
- POST to pulp_exporters//export - Does The Thing

    - returns a task
    - task-created resource of HREF for specific instance of an-export pulp_exporters//export/
    - create-an-export vs have-an-export vs have an instance-of-an-export
- Export needs to include sha256 of created-archive

#9 - 03/04/2020 09:16 PM - ggainey

- Description updated

#10 - 03/06/2020 08:59 PM - ggainey

Q&A with katello team 2020-03-06

attendees: ttereshc ipanova ggainey bmbouters dkliban daviddavis croberts jsherril jturel

- dryrun discussion
    - maybe don't need 'immediately' - but soon
    - should always do the dry-run equivalent on import **first** ?
    - RE fail-on-error?
        - can't find repo? - fatal
        - can't find artifact? - warn and continue?
        - task will lock all involved repos at start-time
        - can we 'rollback' in case of failure? (no, not really)
            - will import be idempotent?
                - yes should be
            - recoverable ("we have the data but something transient went wrong")
            - non-recoverable - missing data in export?
            - discussion about where we might be able to catch this/validate export/import
            - prob needs more pulp3-dev-discussion, katello seems to be ok with "do your best and report any errors"
- how do we let the user delete the **file(s)** associated with an Export?
- how do we deliver the file to the user? (assume user **does not** have shell-access to pulp server)* how do we cut up the file?
    - thought that was katello's thing? - no, alas
    - are we owning 'iso generation'? - no, but **do** own 'max file size' problem
- Pulp3 needs to own the ability to specify image-size and respond appropriately
    - to avoid triple-storage, pulp has to solve this problem
    - we need to find a dataformat that solves the split/recombine problem
    - initial tech-prev release **does not** need this (as long as we can add)
- history-via-export - actually wants to be per-repo?
    - katello may keep this info so we dont' have to
    - maybe first phase impl is 'immutable exporters' (consensus is 'yes')
- timeline/roadmap
    - pulpcore 3.3 is end of March - want for end-of-May-release for katello
    - katello: prob unable to start integrating before end of April
    - katello use-of requirement would be July
    - can we get katello involved earlier? - jturel says yes
    - so 'tech preview' for 3.3
    - add 'split into multiple files' in April(ish)?

#11 - 04/23/2020 03:07 PM - daviddavis

- Subject changed from [EPIC] Importers/Exporters to [EPIC] Pulp import/export

#12 - 09/15/2020 09:23 PM - daviddavis

- Status changed from NEW to CLOSED - CURRENTRELEASE

Closing out epic. Will file bugs/enhancements as follow up issues.