Pulp - Issue #6121

Syncs of very large repositories take longer than Pulp 2

02/09/2020 10:55 PM - dalley

Status:	CLOSED - CURRENTRELEASE	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	lmjachky	**Estimated time:**	0:00 hour
Category:			
Sprint/Milestone:	3.3.0		
Severity:	2. Medium	**Groomed:**	No
Version:		**Sprint Candidate:**	No
Platform Release:		**Tags:**	
OS:		**Sprint:**	Sprint 67
Triaged:	Yes	**Quarter:**	

Description

A Pulp 2 immediate-mode sync of 150,000 files using this repository [0] takes 37.5 minutes

With Pulp 3, the same sync takes about 3x as long (about 2 hours).

An on_demand sync of the same repo in Pulp 3 takes about 90 minutes, still much longer than Pulp 2.

The first 50k files seem sync roughly as fast as they do for Pulp 2, but once the number grows higher to progress starts being made very very noticeably slower than at the beginning.

[0] http://quartet.usersys.redhat.com/pub/fake-repos/very_large_file_150k/

Related issues:

Related to File Support - Issue #6104: pulp_file performance test takes too l...	**NEW**

Associated revisions

Revision fdaa0718 - 02/28/2020 11:36 AM - Lubos Mjachky

Fetch excluded content directly from the database

fixes #6121 https://pulp.plan.io/issues/6121

History

#1 - 02/10/2020 03:06 PM - dalley

- Subject changed from Immediate mode syncs of very large repositories fail to Syncs of very large repositories take longer than Pulp 2

- Description updated

#2 - 02/11/2020 04:29 PM - daviddavis

- Related to Issue #6104: pulp_file performance test takes too long for Travis added

#3 - 02/11/2020 04:30 PM - daviddavis

Once we fix this performance problem, we should increase the number of files in the file-perf fixture. We had to decrease the number to 20K due to timeouts in Travis[0].

[0] https://pulp.plan.io/issues/6104

#4 - 02/11/2020 04:32 PM - fao89

- Triaged changed from No to Yes

- Sprint set to Sprint 66

#5 - 02/14/2020 03:02 PM - fao89

I believe the solution for performance problems is to make StagesAPI customized by plugin users, on the past, batches had a maximum size of 100 and a minimum of 50, with this PR I increased the minimum to 500 and maximum to 1000: https://github.com/pulp/pulpcore/pull/440/files For a repo with 150000 files, we will use StagesAPI machinery about 150 times, so I think the batch size should be larger. I think in 2 options:

- Make it adjustable by the user/plugin
- Define batch size as a fraction of the total content, like 10%

#6 - 02/18/2020 08:05 PM - bmbouter

Is my understanding correct that policy='on_demand' or policy='streamed' performance of Pulp3 is faster than Pulp2? And this ticket is for policy='immediate' only?

#7 - 02/18/2020 08:29 PM - bmbouter

As a next step on this I believe we need to return to profiling a pre-built test. If we needd to load Pulp up with a lot of units first we can do that too, but we need to understand with cprofile where the time is going. Is the increase in units increasing the postgresql query runtime. If so which queries are taking the longest and can we optimize those?

In terms of batching, currently the batch size is configurable so I think it's mostly around setting the right default. The larger the batch the larger the memory usage of that stage so we need to be careful to not load it up too much. This is why I think we should set the value and not make it variable based on the unit counts in the entire repo.

#8 - 02/19/2020 11:08 AM - lmjachky

- Status changed from NEW to ASSIGNED

- Assignee set to lmjachky

#9 - 02/21/2020 03:34 PM - rchan

- Sprint changed from Sprint 66 to Sprint 67

#10 - 02/21/2020 04:27 PM - lmjachky

- File sync-file-perf-cprofile.txt added

- File sync-file-perf-cprofile.out added

I have attached the output of cprofile. I ran performance tests for the plugin pulp_file on my local machine and profiled the sync. There were loaded 20,000 files into Pulp where each of the files had a size of 50 bytes.

This is just a status update. I will continue working on the issue and I will try to increase a number of files to see how the performance changes over time with the additional load.

#11 - 02/28/2020 11:43 AM - lmjachky

- File feb28-fix.png added

- File feb28-fix.out added

- File feb28-master.png added

- File feb28-master.out added

I concur that syncs of very large repositories take a tremendous amount of time. I tried to sync a repository which contains 200k units. With the current version of pulpcore, the sync took more than 130 minutes even on a powerful machine.

```
-> Sync tasks => Waiting time (s): 6.965342 | Service time (s): 8330.099087

-> Resync tasks => Waiting time (s): 0.048282 | Service time (s): 2413.934992

-> Publication tasks => Waiting time (s): 0.047476 | Service time (s): 230.289603

-> Distribution tasks => Waiting time (s): 0.046366 | Service time (s): 0.103166
```

I observed that the most time is consumed by the method __iter__ of the class FlatValuesListIterable. The problematic part is shown below (https://github.com/pulp/pulpcore/blob/9614b16b795fa3aa0f6ad7ccbc1002b2119a2590/pulpcore/app/models/repository.py#L584-L586):

```
to_add = set(content.values_list('pk', flat=True))
for existing in batch_qs(self.content.order_by('pk').values_list('pk', flat=True)):
    to_add = to_add - set(existing.all())
```

I replaced these lines by one database call (https://github.com/pulp/pulpcore/pull/565) and the performance improved drastically.

```
-> Sync tasks => Waiting time (s): 0.052631 | Service time (s): 2304.212409

-> Resync tasks => Waiting time (s): 0.051633 | Service time (s): 2425.380134

-> Publication tasks => Waiting time (s): 0.047535 | Service time (s): 229.94097

-> Distribution tasks => Waiting time (s): 0.047981 | Service time (s): 0.103984
```

#12 - 02/28/2020 03:27 PM - dalley

Great work, that is a huge improvement! It's interesting that resyncs take a little longer than syncs, maybe there's still room to improve that (as part of a separate issue)

#13 - 03/03/2020 03:41 PM - Anonymous

- Status changed from ASSIGNED to MODIFIED

Applied in changeset pulpcore|fdaa07183843e40654e765d25a77a9897740bbd9.

#14 - 04/15/2020 09:56 PM - ttereshc

- Status changed from MODIFIED to CLOSED - CURRENTRELEASE

- Sprint/Milestone set to 3.3.0

Files

sync-file-perf-cprofile.txt	195 KB	02/21/2020	lmjachky
sync-file-perf-cprofile.out	244 KB	02/21/2020	lmjachky
feb28-fix.out	240 KB	02/28/2020	lmjachky
feb28-fix.png	1.04 MB	02/28/2020	lmjachky
feb28-master.png	585 KB	02/28/2020	lmjachky
feb28-master.out	239 KB	02/28/2020	lmjachky