Pulp - Task #2449

Figure out current and expected performance for applicability calculation

11/30/2016 07:18 PM - ttereshc

Status:	CLOSED - COMPLETE	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	ttereshc	**% Done:**	0%
Category:		**Estimated time:**	0:00 hour
Sprint/Milestone:	3.0.0		
Platform Release:		**Tags:**	
Groomed:	Yes	**Sprint:**	Sprint 21
Sprint Candidate:	Yes	**Quarter:**	

Description

Compare applicability calculation performance for:

- the latest upstream Pulp
- the latest downstream Katello
- the latest downstream Spacewalk

Ask for the expectations or required performance improvements for the downstream Katello.

Figure out if we can just convert applicability mostly as-is to Pulp 3 or if there is a need for re-design.

Related issues:

Blocks Pulp - Task #2450: Create a plan for applicability calculation impleme...	**CLOSED - WONTFIX**

History

#1 - 11/30/2016 07:29 PM - ttereshc

- Blocks Task #2450: Create a plan for applicability calculation implementation added

#2 - 11/30/2016 08:01 PM - semyers

How do we plan to quantify "applicability calculation performance" to make those numbers comparable across the different implementations? If that isn't well-defined then I think that this story's acceptance criteria are also not well-defined. Clearly defining what we're measuring and how should make the story unambiguous. For example, is it the time it takes a service to complete the entire task, from asking it perform the applicability calculation to returning the result? Is it the time that the single function responsible for performing the applicability calculation (assuming there is a single function) to run, without involving any tasking systems? (and so on...)

It's possible that we can't define this without first digging into the three different projects under test, but I think that we'll want to be especially clear about what we're measuring with this task so that we can reliably replicate the test in pulp 3 for comparison later.

#3 - 11/30/2016 08:17 PM - ttereshc

I think it is mostly about time. But I assume it will be clear after discussion what current issues or customer complains in this area are.

One can create a test setup, multiple consumers/repos (maybe @dkliban has a script for it already, iirc) and ask/test it in various projects.
is it a bad idea?

#4 - 11/30/2016 09:08 PM - semyers

- Groomed changed from No to Yes

I don't think it's a bad idea, I just think it's important that we make it very clear how we measured the performance so that we can reproduce the tests fairly across all platforms, including the pulp 3 implementation. I don't think we can know what method is best until we've looked at all 3 existing platforms, which means I think this is good enough for someone to get started, and the comments on the issue should help with the acceptance upon completion; we can refine the story as-needed later.

#5 - 12/01/2016 06:01 AM - mhrivnak

Agreed. A critical part of this work is to define a representative test that can be applied directly to the pulp and katello scenarios, and that at least makes sense to a spacewalk dev. If we can directly test with spacewalk, that's great, and we should. But in case it's too difficult to get an apples-to-apples comparison, starting with a sanity check from their dev team: "we did $this in $time on $hardware; would customers complain about that?" would also be quite valuable.

#6 - 12/01/2016 04:54 PM - mhrivnak

- Sprint/Milestone set to 30

#7 - 03/16/2017 04:06 PM - mhrivnak

- Sprint/Milestone changed from 30 to 36

#8 - 04/04/2017 12:23 AM - ttereshc

- Status changed from NEW to ASSIGNED

- Assignee set to ttereshc

#9 - 04/10/2017 02:59 PM - mhrivnak

- Sprint/Milestone changed from 36 to 37

#10 - 05/01/2017 04:15 PM - jortel@redhat.com

- Sprint/Milestone changed from 37 to 38

#11 - 05/17/2017 02:37 AM - ttereshc

Profile - set of packages installed on a consumer system
Applicability - set of packages and errata applicable (aka should be updated) to a particular consumer profile.

Currently the following applicability calculation requests are available:

- for repositories:

- applicability is calculated from scratch for every unique consumer profile for every specified repo to which profile is bound
- can be calculated in parallel by multiple workers without resource reservation

- for consumers

- applicability is calculated for every new profile (out of one or more specified consumer profiles) for every repo to which profile is bound
- can be calculated in parallel by multiple workers with resource reservation for individual consumers only

The variables of interest:
- the number of unique consumers' profiles
- the size of a consumer's profile
- the number of errata and packages in a consumer's bound repository
- the number of the changes in consumer's bound repositories
- the applicability calculation type - for consumers or for repositories

Common cases:
- consumer just registered and got bound to a set of repositories. Applicability calculation is triggered for this particular consumer only.
- the number of consumers just registered and got bound to a set of repositories. Applicability calculation is triggered for them.
- there were few updates in a big repo. Applicability calculation is triggered for updated repo. That means re-calculation will be done for the entire repo for every consumer bound to this repo.

Average deployment:
- 4000 consumers (close to 700 unique consumer profiles)
- 1000 average number of packages in consumer profile
- 3 bound repositories
- 60000 RPMs across those repositories (20000 RPMs in a single repo)
- 4000 errata across those repositories (most errata in a single repo)

Large deployment:
- 40000 consumers (close to 10000 unique consumer profiles)
- 1000 average number of packages in consumer profile
- 10 bound repositories
- 170000 RPMs across those repositories (20000 RPMs in a single repo)
- 20000 errata across those repositories (most errata in a single repo)

Suggested tests (common cases + some additional ones):

- for consumers:

- one new consumer with new profile bound to 2 repositories, big and small one
- one new consumer with existing profile bound to 2 repositories, big and small one
- N new consumers with mostly new profiles bound to 2 repositories, big and small one
- N new consumers with mostly existing profiles bound to 2 repositories, big and small one

- for repositories

- average deployment, small number of updates in a large repo (the most common case)
- average deployment, large number of updates in a large repo
- large deployment, small number of updates in a large repo
- large deployment, large number of updates in a large repo

Comparison to Spacewalk:
- ***(updated)*** they re-calculate applicability for all consumers and all repositories in case of any changes on consumer or repository side and not on request
- ***(updated)*** they update consumer profiles periodically (on every interaction with Spacewalk or every 1-4 hours) and not on request
- ***(updated)*** they re-calculate applicability incrementally (in certain cases) but most of the time from scratch
- I am not sure if they make calculation for a unique profile or for every consumer
- I was told that with large number of updates in a large repo applicability re-calculation can take up to ~30 mins
- I was told that usually applicability calculation does not take longer than few mins.
- ***(updated)*** Applicability may not be up to date becasue of possible delays in applicability re-calculation, Spacewalk does not communicate it to the user.

So it should be easy to compare with them cases when consumers were just registered.
The case when there are few changes in a repo is really fast for them to re-calculate since they are doing it incrementally.

Suggested next steps:
- provide a script to grab some statistics for average deployment by calculating all the variables of interest
- provide a script to generate consumer profiles for suggested tests based on the synced repos in db
- run tests on upstream Pulp in the most optimized way possible (in parallel without resource reservation)
- run tests on Katello downstream
- ask Spacewalk downstream to test at least the cases when consumers were just registered
- decide on minimum acceptable performance for Pulp 3 based on those tests

Any feedback, suggestions and objections are welcome!

Thanks to Katello (@jsherrill) and Spacewalk (@tlestach, @dyordano) for their time and input on this topic.

#13 - 05/17/2017 08:27 PM - mhrivnak

That all sounds great.

One question that could be helpful: How does Spacewalk communicate to the user how "fresh" its applicability data is? Do they have something in the UI that shows it? If so, a screenshot might be very useful to share around. One key lesson we got from Grant when he told us about their applicability solution is that setting expectations is key. Showing the user how fresh the calculations are, and perhaps even what the state of in-progress calculations are, would be a way to do that.

#14 - 05/22/2017 10:48 PM - mhrivnak

- Sprint/Milestone changed from 38 to 39

#15 - 05/26/2017 12:29 AM - ttereshc

Spacewalk does not communicate to user how fresh applicability is.
I've added some info to the previous comment , modified lines are marked as ***(updated)***.

#16 - 06/12/2017 04:06 PM - mhrivnak

- Sprint/Milestone changed from 39 to 40

#17 - 06/29/2017 05:59 AM - mhrivnak

Tanya, can you summarize where this work stands, and what you think remains for it to be complete?

#18 - 06/29/2017 10:57 AM - ttereshc

It should be completed this week.
Some work on script to generate test data and getting results for suggested scenarios are remaining parts I think.

#19 - 07/03/2017 02:37 PM - mhrivnak

- Sprint/Milestone changed from 40 to 41

#20 - 07/04/2017 12:17 AM - ttereshc

- File consumer_applicability_stats.py added

- File gen_test_data.py added

#21 - 07/05/2017 02:42 AM - ttereshc

Add "roughly" to almost any number below (hw was slightly different, deployment size was slightly different, not every scenario is possible in all projects, etc).

Hardware (close to minimal requirement for most projects): 2 CPU cores, 2.4 GHz, 4 MB cache, 8GB RAM
Other conditions (for Pulp and Katello only): 4 workers (aka 4 processes for concurrent applicability generation)
Deployment:

- 500 unique consumer profiles
- 1000 number of packages in consumer profile
- 3 bound repositories
- 65000 RPMs across those repositories
- 4000 errata across those repositories (most errata in a single repo)

Applicability generation time, Pulp/Katello/Spacewalk:

- one newly registered consumer - 12 sec/16 sec/up to 50 sec[1]
- 500 newly registered hosts - 1 hour/1 hour[2]/up to 50[1] x 500[3] = it's likely be 30 mins or so and not few hours[1]
- one existing host with few changes in its profile - 12 sec/ 16 sec/1-2 sec
- 500 existing hosts with a few changes in their profile - 1 hour/1 hour[2]/1-2 sec x 500[3] = 10-15 min
- one existing host with many changes in its profile - 12 sec/16 sec/1-2 sec
- 500 existing hosts with many changes in their profile - 1 hour/1 hour[2]/2 sec x 500[3] = 15 min
- one repository (12000 packages, 4000 errata), small number of updates, all 500 hosts are affected - 15 mins/15 mins/10 mins[4]
- one repository (12000 packages, 4000 errata), large number of updates, all 500 hosts are affected - 15 mins/15 mins/15 mins[4]

Conclusion:

- Pulp and Katello have similar performance currently.
- Spacewalk need more time to generate applicability for the first time, almost all subsequent regenerations are quick.
- Pulp/Katello performance depends on number of packages and errata in consumer's bound repos. Spacewalk - not that much.
- For Pulp 3
  - Performance should not be lower than the current one.
  - Version repositories can potentially improve subsequent applicability regeneration (aka not the first one).

1. Spacewalk: "Database-optimizers take time to 'warm up' after restarting, the first few times a given kind-of query is run are going to perform badly". ↩

2. Katello recently started to use API call for single consumer applicability regeneration which improved performance, so currently Pulp and Katello show the same results. ↩

3. I was told that Spacewalk generates applicability sequentially so any time for 1 consumer should be multiplied by the number of consumers. ↩

4. There is no separate case in Spacewalk for requesting applicability regeneration for a repository. For similarity with Pulp workflow (applicability regenerated for existing applicability profiles only), time for existing-consumers-with-updates case is used for Spacewalk. ↩

#22 - 07/05/2017 02:43 AM - ttereshc

- Status changed from ASSIGNED to CLOSED - COMPLETE

- Sprint/Milestone changed from 41 to 40

#23 - 03/09/2018 12:16 AM - bmbouter

- Sprint set to Sprint 21

#24 - 03/09/2018 12:17 AM - bmbouter

- Sprint/Milestone deleted (40)

#25 - 04/25/2019 06:47 PM - daviddavis

- Sprint/Milestone set to 3.0.0

#26 - 04/26/2019 10:39 PM - bmbouter

- Tags deleted (Pulp 3)

Files

consumer_applicability_stats.py	4.93 KB	07/03/2017	ttereshc
gen_test_data.py	4.24 KB	07/03/2017	ttereshc