Performance effects of turning off background fetch

Summary

Background fetch as it is right now does not seem to serve well with regards to enabling all container workloads to start faster. This is due to lock contention on accessing spans between background fetcher process and on-demand fetch. Moreover there are questions of how background fetching process is being stopped and resumed. That said, there is a need to see if turning off the background fetcher until it gets re-worked can help.

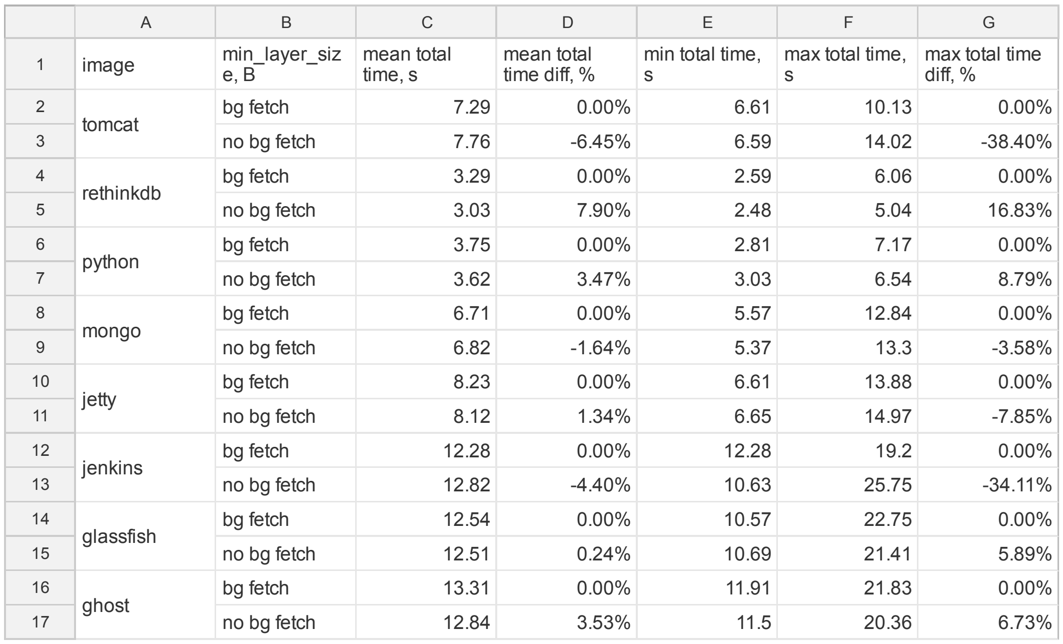
This document provides an overview of the performance effects of turning off the background fetch in snapshotter as of https://github.com/awslabs/soci-snapshotter/commit/3176dc187566ae5c87cfe015293664c168428084.

Looking at benchmarking data across images, the results are contradictory:

1. For images such as: tomcat, mongo, jetty, jenkins, tensorflow and drupal there is a regression in mean time to start the container workload of 1.44-6.45% and max time to start the container workload of 1-38.4%.
2. For other images, the improvement in the mean time to start the container workload is 0.33-7.9%, and the improvement in max time to start the container workload is 4.25-16.83%.

Given the contradictory results it makes sense to turn off the background fetcher until it is re-worked and re-evaluate it afterwards.

Benchmarking results



	A	B	C	D	E	F	G
1	image	min_layer_size, B	mean total time, s	mean total time diff, %	min total time, s	max total time, s	max total time diff, %
2	tomcat	bg fetch	7.29	0.00%	6.61	10.13	0.00%
3		no bg fetch	7.76	-6.45%	6.59	14.02	-38.40%
4	rethinkdb	bg fetch	3.29	0.00%	2.59	6.06	0.00%
5		no bg fetch	3.03	7.90%	2.48	5.04	16.83%
6	python	bg fetch	3.75	0.00%	2.81	7.17	0.00%
7		no bg fetch	3.62	3.47%	3.03	6.54	8.79%
8	mongo	bg fetch	6.71	0.00%	5.57	12.84	0.00%
9		no bg fetch	6.82	-1.64%	5.37	13.3	-3.58%
10	jetty	bg fetch	8.23	0.00%	6.61	13.88	0.00%
11		no bg fetch	8.12	1.34%	6.65	14.97	-7.85%
12	jenkins	bg fetch	12.28	0.00%	12.28	19.2	0.00%
13		no bg fetch	12.82	-4.40%	10.63	25.75	-34.11%
14	glassfish	bg fetch	12.54	0.00%	10.57	22.75	0.00%
15		no bg fetch	12.51	0.24%	10.69	21.41	5.89%
16	ghost	bg fetch	13.31	0.00%	11.91	21.83	0.00%
17		no bg fetch	12.84	3.53%	11.5	20.36	6.73%

18	gcchellocompile	bg fetch	5.56	0.00%	4.47	11.16	0.00%
19		no bg fetch	5.38	3.24%	4.43	9.31	16.58%
20	tensorflow	bg fetch	23.8	0.00%	20.12	42.34	0.00%
21		no bg fetch	24.67	-3.66%	20.63	42.8	-1.09%
22	drupal	bg fetch	5.64	0.00%	4.75	10.5	0.00%
23		no bg fetch	5.81	-3.01%	5.03	10.76	-2.48%
24	redis	bg fetch	2.45	0.00%	1.78	4.63	0.00%
25		no bg fetch	2.13	13.06%	1.73	3.78	18.36%
26	rabbitmq	bg fetch	15.24	0.00%	14.03	19.3	0.00%
27		no bg fetch	15.19	0.33%	14.3	18.48	4.25%

Appendix A. Configuration

The benchmarking was run on the dev desktop with the following configuration:

- Host type: m4.2xlarge
- RAM: 32GiB
- snapshotter’s commit id: 51192d3ae9bca5c178b61c945ad27503fab94d78
- benchmarking framework commit id: 2a2c9165da9cc4cf3da105a6142691cf0e62a304

Contents of soci_config.toml:

```
image_service_path = "/tmp/containerd-grpc/containerd.sock"
[cri_keychain]
enable_keychain = true

no_background_fetch = true
```