Pulp - Issue #8839

file content upload performance needs improvement -- currently about 5x slower than rsync

05/28/2021 03:48 PM - vk

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

Description

Following up from IRC #pulp, I've just recently started using pulp 3.12.2, and tried uploading a 1Gbyte file.

I created it with:

```
dd if=/dev/urandom of=./testfile bs=1048576 count=1024
```

That took about 5 seconds (suggesting my local spinning disk based filesystem could handle about 200MB/s, assuming no bottlenecks from /dev/urandom).

Then I did a:

```
pulp file content upload ...
```

It took about 3 mins, which is quite a long time. This is on the same machine where I created the file, to pulp running locally.

I did an rsync over ssh of the same file to a second machine and it took a little over 9s. On the second machine, I set up a replica repo with a remote set to the first pulp instance. The sync took a little over 9s -- i.e. matched rsync.

On the first pulp instance, based on suggestions from IRC, I updated nginx.conf to set client_max_body_size to 1024m for /pulp/api/v3 (and /pulp/content -- though I don't know if the latter was needed).

I then used --chunk-size 1000000000 (1 billion bytes) with the pulp file content upload and got down to 43s. That's still 4.8x slower than rsync.

I realize there are a few seconds of overhead for database operations and checksums (I measured the latter at about 5-6 seconds by running sha256/384/512sum and totaling them). But still, it seems quite slow.

At the moment, this is just a prototype setup. My goal is to have pulp instances globally as replicas for our custom file artifacts. We need to keep the total time taken from uploading an artifact through availability to clients around the world to be as low as possible -- i..e. exploit the best that the underlying infrastructure is capable of. Most of the files I anticipate to be under 100Mbytes, in which case I expect other background operations to take up a greater percent of the total time. I haven't even counted time taken for updating the Publication and Distribution that clients will connect to.

What can be done to improve this?

History

#1 - 06/01/2021 08:08 PM - daviddavis

There's something like a 30-40% overhead that's incurred by using the chunk upload API because it has to handle the upload chunk file/object, validate the chunk(s) as they are uploaded, and create/run a task to combine the chunks into an artifact. And this overhead exists even if the chunk size is bigger than the file. We could add functionality to the CLI to let users bypass chunked uploading and upload to the artifact creation endpoint directly.

We could save another few seconds (10% or so) by not calculating the checksum digests on the client side for the file. We send this digest to pulp to ensure that the upload isn't corrupted. I'm not sure the savings are worthwhile but maybe we could add an option.

Lastly, we could try to see if we can optimize the CLI/server code as well.

#2 - 06/02/2021 01:49 PM - daviddavis

I confirmed with @vkv that this command took 19s (or about half the time):

```
time http --form :24817/pulp/api/v3/artifacts/ file@testfile sha256=$(sha256sum testfile | cut -d' ' -f1)
```

#3 - 06/02/2021 09:07 PM - bmbouter

Am I reading correctly that not using the chunking API, but a straight file-upload is faster (like in Comment 2)?

Also when using the chunking API do the chunks upload in parallel?

I think it's important to have checksum calculation on the client be default, but if an option to disable it existed I'd also be ok with that.

Also how can "the chunk size is bigger than the file" ?

#4 - 06/02/2021 09:26 PM - daviddavis

bmbouter wrote:

> Am I reading correctly that not using the chunking API, but a straight file-upload is faster (like in Comment 2)?

Correct.

> Also when using the chunking API do the chunks upload in parallel?

No but even in the case where the chunk size exceeds or is equal to the file size (and there is only one chunk), it's still rather slow.

> I think it's important to have checksum calculation on the client be default, but if an option to disable it existed I'd also be ok with that.

Agreed

> Also how can "the chunk size is bigger than the file" ?

Chunk size is an option when uploading. You can set it to something greater than the size of the file in which case it uploads the entire file in one chunk. So like if you have a 1 kb file and you have a 1 mb chunk size, it'll just upload the file in one chunk but it still uses the chunk upload api.

#5 - 06/02/2021 10:03 PM - bmbouter

Thanks for the info.

One of the main reason for the chunking API is to get around the file-size maximum of nginx and apache. So given that if the file is < $webserver_size_max then the CLI should use the non-chunking API by default. If >= $webserver_size_max it should use the chunking API. Having an option to force one or the other, and/or to adjust the limit the CLI should use to think of $webserver_size_max I think would be good. For example nginx I believe defaults to 1mb but the installer can configure this see here.

Also the CLI when using the chunking API should upload in parallel. Even if the network isn't allowing speed gains due to the rate limit, having the chunk sha256 calculation happen in parallel on the server side (and locally) would be faster.

I'm not sure if chunk-size is exposed to the CLI, but I think it should not be if possible.

#6 - 06/02/2021 10:24 PM - daviddavis

bmbouter wrote:

> I'm not sure if chunk-size is exposed to the CLI, but I think it should not be if possible.

I'm not sure I follow about why chunk size should not be exposed as an option in the CLI? If a user has a Pulp server set up that accepts a certain size of request (500kb), how do they inform the pulp cli?

#7 - 06/02/2021 10:34 PM - bmbouter

I think we're both wanting the user to specify the max size the server can accept. If that's called chunk-size that's great. Will it default to the default of nginx and apache?

The idea I'm trying to bring forward is a smarter cli where the user doesn't have to make a choice about chunking or not. Have the user either declare a different chunk size, or use the default, but in both cases the CLI knows when to chunk and when to not.

Does this help answer it some?

#8 - 06/04/2021 08:41 PM - daviddavis

- Project changed from File Support to Pulp

I've opened the following against the CLI:

https://github.com/pulp/pulp-cli/pull/262

https://github.com/pulp/pulp-cli/issues/263

I think that should be most of what we can do in the CLI to speed up uploads there.

The rest should of the work will need to happen in the pulpcore upload code.

#9 - 06/08/2021 05:06 PM - dkliban@redhat.com

- Triaged changed from No to Yes