Crane - Issue #1708

[RFE] Crane Gives Up After First Failed Attempt At Grabbing Image

02/22/2016 05:05 PM - rhartman

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

Description

Sometimes when we push images to our docker registries, the images
don't always make it in every location they should be (by accident, error, etc).

When this happens, it makes said images appear available, but are actually
still inaccessible. The bigger issue arises when that image is used by another
repo as well (such as a base image).

When this issue occurs, we get unexpected outages. Someone will try to pull
from another repo that shares that layer, Crane goes looking for it, the first
place it sees it should be is in the broken repo, it checks, can't grab the image
and gives up.

I'm requesting that Crane try again, if it doesn't find the image the first time,
look for the next place it exists and try there. It would still be helpful to see
output somewhere that it failed to pull an image so we know there's an issue, but this
would stop errors in one repo affecting the rest of the registry.

History

#1 - 02/24/2016 03:59 PM - mhrivnak

I see the value in what you're asking for, but unfortunately crane doesn't have an opportunity to implement this behavior given its current mode of operation.

When a client requests an image, crane responds with a 302 redirect to the location where that image lives. It is then up to the client (docker) to make a new request to that location. If that request fails, crane has no idea, nor any opportunity to intervene.

One option is to make crane a pass-through proxy. But that would make crane a bottleneck. And it would give crane new responsibilities that a cdn usually handles, like cache headers, deduplicating retrieval of bits from their source, a need to stream bits out as they're accessed, etc. It also removes an opportunity for a cdn to take advantage of geographic proximity to the client. It is much simpler for crane to be a redirect machine that's good at making fast, short responses, and let large files be served by another service designed for that purpose.

Another option is for crane to try pre-fetching each file to make sure each redirect is going to succeed. Even a HEAD request would probably help. But there are two problems with this. One is that crane may not have auth credentials to access the remote content, such as if it's on the Red Hat CDN. The other is more a design inefficiency. Rather than have crane monitor availability of content and try to offer alternate locations (and sometimes there won't be alternate locations available), it would be more productive to do that monitoring somewhere else that is capable of either fixing the problem, or notifying someone who can fix it. Even better, it would be a great idea to deploy the image files first, automatically test their availability, and only then deploy the crane data file if the images are proven to be available.

Does that make sense? Do you have any other ideas I've overlooked?

#2 - 02/29/2016 10:30 PM - mhrivnak

- Status changed from NEW to CLOSED - NOTABUG

#3 - 04/15/2019 10:34 PM - bmbouter

- Tags Pulp 2 added