Solr Staging is Broken
(Tuning Ensemble Notes)

Jan 19, 2024

- New plan
    - Restore all desired staging indexes to the new cluster
        - (note the pulfalight-staging is good, doesn't need to be migrated)
    - switch the lib-solr8-staging DNS to the new cluster
-

Jan 19, 2024

- This is the link that brought back dss-staging1 back up
    - https://stackoverflow.com/questions/3938691/how-to-recover-from-solr-deleted-index-files
    - What we did:
        - ssh to lib-solr-staging6
        - become deploy user
        - cd /solr/data
        - In the solr admin panel identify which shard to recover:
            - go to collections
            - click the collection you want to recover
            - expand all its shards and find the one that's on lib-solr-staging6; it will have status: "down"
        - create a directory in /solr/data that has the same name as the shard name
        - copy a core.properties file from out of one of the existing directories into the new directory
        - edit the core.properties file to reflect the shard you want to recover.
            - the coreNodeName can be taken from the "Replica" heading of the shard (e.g. core_node7)
            - the config name is also on that admin screen, on the left
            - copy and past the "core" value for the shard into the "name" field of the file
            - update the collection value in the file
            - If it's a shard2, update the shard name
        - place the updated core.properties file in that new directory
        - create a `data` directory in your new directory
        - restart the solr service `sudo service solr restart`

        - you will see it creates the index directory inside that new data directory, and starts to populate it from the leader
- Collections with only 1 shard
    - pulmap-staging is only on lib-solr-staging5
    - special-collections-staging1 is only on lib-solr-staging4
- Collections we want to delete (try the delete from zookeeper thing on these??)
    - catalog-alma-staging1
    - catalog-alma-staging2
    - lae-staging2
    - dpul-production (not found in princeton_ansible)
    - pulfalight-staging1 (this one looks weird, it says there's 1 shard but that shard doesn't exist on any machine)
    - cicognara-test (not found in princeton_ansible)
    - pulfalight-staging (collection data is lost on disk and anyway it's currently looking at the new infra)

Jan 18, 2024

- Pul_solr was not deploying to new boxes.
    - Anna was able to remove the solr wrapper
    - Used lando
    -
- Next steps Proposal 1:
    - Point anything that was looking at the new d machines back to the original staging boxes
    - Upgrade to jammy on staging boxes in situ
        - Francis fixed this
    - Upgrade to jammy on prod boxes in situ one at a time.
    - Note that the solr role deletes collections. But if you run them one at a time the collections should re-replicate after each run. Or Francis can update the role so it doesn't do that.
- Have we tried the pdc discovery process on new cluster?
    - No we have not.
    - Bess can set it up right now to see whether it will fail. It may take a while, we would know probably on Monday
    - We currently point to http://lib-solr8-staging.princeton.edu:8983/solr/pdc-discovery-staging – what should it point to?
- Note: we want to add another replica to DSS-staging
- Q: If we decide we want the new boxes, why do we have to rename them so they don't have 'd' in them?
    - reduce confusion
- Next steps Proposal 2 :

    - Shut down lib-solr-staging6
    - rename lib-solr-staging6d to lib-solr-staging6
    - wait for collections to replicate
    - do the same for each of the other boxes.
- Old cluster is lib-solr8-staging (load balancer DNS name)
    - New cluster is lib-solr8d-staging
- We will move forward with Proposal 1.
- Here's a thing about clusterstate.json https://stackoverflow.com/questions/16579242/solrcloud-delete-collection-bug

Jan 11, 2024

For today, we will address the immediate issue. Some of us (Alicia, Francis, maybe Anna, Robert, Bess) may also dig more deeply into Solr and Zookeeper in general.

What is wrong? What behavior are we seeing? We have three current problems:

1. pdc-discovery repopulates its solr index every half-hour by creating a new collection, reindexing, then switching Solr to point to the newly reindexed collection. The process worked fine in production and in staging for many months. Without any changes anyone can remember or find, staging then broke, first breakage was August 19, 2023. Since then staging has been consistently broken; prod is still fine. When staging is broken, you can query solr, but you cannot make any changes to collections (add or delete).
2. we're seeing solr connection errors on the new pufalight-staging - not sure if this is a separate issue or connected to problem #1
3. lib-solr-staging6 is broken as a side effect of the attempt to upgrade the OS to Jammy; the other staging solr boxes are still on bionic

Next Steps:
1. ~~Turn off Solr on lib-solr-staging6; this should remove it from the solr cluster from zookeeper's viewpoint - this may fix problem #2~~ - **DONE**, investigation showed that pulfalight staging ONLY had collections on lib-solr-staging6; Anna will reindex on one or both of the other two boxes (lib-solr-staging4, lib-solr-staging6)
2. Build 6 new staging VMs based on Jammy and install Zookeeper then Solr on them, test as much as we can (keep existing Bionic machines running while we do this)
3. Optionally migrate (or copy) existing staging indexes from the Bionic machines to the Jammy machines
4. Swap the new Jammy VMs over so the staging Solr and Zookeeper services are running on those

Clarifying the status of staging solr: We all agree - staging should be “breakable”, but our current aim is to create a reliable, stable staging system that we can roll back to (after breaking it with experimentation). Staging solr shouldn’t be permanently, consistently broken.

Nov. 28, 2023

The pdc_discovery application has two collections with an alias in Solr and reindexes every 30 minutes. The job starts by deleting the inactive index, then reindexing from scratch, then changing the alias to point to the freshly reindexed collection. On the staging solr boxes, the delete job fails, triggering a Honeybadger error.

When we looked at the boxes, all the errors in the logs are related to the delete action. However, the errors only happen when the heap usage gets high. It looks like garbage collection is stopping activity on the solr boxes in that situation. We do not understand what is making the memory on the staging servers work this hard - the logs show a lot of pings happening, but it’s not clear if they are related or not. We have also changed some Solr parameters as part of the Search and Race work, but we’re not sure if that’s related either.

Possible culprits

- Network latency
    - All 3 staging boxes are in the same data center (Forestal - the 3rd box was moved this week, but we continued to see the issue)
    - TEST: We could test this by creating the collection with a single shard and a single node (i.e. no replication). If Network latency is at fault then the errors will go away.
    - **We tried this test, then remembered that we should check the location of the Zookeeper boxes. And two of them were in New South! We moved those, re-tested without replication (worked fine), retested with replication (also worked fine). And garbage collection is working well with the newer (G1GC) settings.**
    - **SOLUTION: keep all Solr servers AND Zookeeper servers in the same data center**
- Garbage collection
    - We noticed that the settings are different on staging vs. production. This was experimental in relation to the big outage over the summer, which turned out not to be a memory issue. An experimental change was made to staging but never applied to production and never reverted.
    - TEST: We could test this by changing the staging config to match production (reverting from G1GC to ConcMarkSweep)
    - We created an issue and a PR for reverting the GC changes, but we may want to revisit the question of which config to use for GC, now that we know it was not actually the problem here . . . also need to figure out how the Jammy upgrade relates to the GC setting changes.

- Heap settings
    - There are heap dump instructions on the pul_solr README
    - The Solr documentation recommends heap of 8-16 GB in production, see also the JVM settings docs
    - TEST: Change the heap allocation - maybe xms of 12 GB and xmx of 16 GB?
- Monitoring pings
    - There is no check in Honeybadger. In Datadog there's a check, but it doesn't seem to be the issue. However, we did update the Datadog check recently, see the Datadog Book Club notes.

Nov. 29, 2023

Things we might want to investigate further

- Look closely at zookeeper
    - Zookeeper machines are
        - lib-zk-staging1, ip is 128.112.203.153
        - lib-zk-staging2, ip is 128.112.203.154
        - lib-zk-staging3, ip is 128.112.203.155
    - Zookeeper logs are in `/var/log/zookeeper`
        - try grepping in zookeeper.log for the solr ip addresses like `ag 128.112.200 zookeeper.log`
        - resulting gist: https://gist.github.com/hackartisan/fc0c5d689533ea774e5775d9e8a129cf
        - example lines:
            - 2781:2023-11-28 18:23:11,452 [myid:3] - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:ZooKeeperServer@942] - Client attempting to establish new session at /128.112.200.92:38174
            - 2782:2023-11-28 18:23:11,479 [myid:3] - INFO [CommitProcessor:3:ZooKeeperServer@687] - Established session 0x38bee51c4960008 with negotiated timeout 30000 for client /128.112.200.92:38174
            - 2784:2023-11-28 18:23:11,543 [myid:3] - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxn@1044] - Closed socket connection for client /128.112.200.92:38174 which had sessionid 0x38bee51c4960008
            - 2780:2023-11-28 18:23:11,429 [myid:3] - INFO [NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCnxnFactory@192] - Accepted socket connection from /128.112.200.92:38174
        - We don't see log lines like this for the prod solr machines on the prod zk machines, which makes us think these are symptomatic log lines.

                - Note that the prod zk machines seem to loc mac addresses rather than IPs?
                - to the extent we see closed socket connections they look like they're coming locally.
                - e.g. 64200:2023-11-25 10:59:54,040 [myid:3] - INFO [Thread-2474476:NIOServerCnxn@1044] - Closed socket connection for client /0:0:0:0:0:0:0:1:33486 (no session established for client)
                - But if we do `ag closed.socket.connection | grep -v 0:0:0:0:0:0:0:1` on staging it comes back empty.
            - Given that there is this difference in logging (IP vs MAC address?) between, could there be configuration differences between prod and staging that are relevant to our bug?
                - We looked at ansible role for zk and it doesn't seem like there are configuration differences
                - Maybe distinctions are due to different operating systems (focal vs jammy)
                - Maybe differences are due to different versions of zookeeper?
                -
    - A zookeeper troubleshooting link: https://docs.apigee.com/api-platform/troubleshoot/zookeeper/zookeeper-connection-loss-errors
- Maybe also look more at solr logs?
    - Solr machines are
        - lib-solr-staging4, ip is 128.112.200.92,
        - lib-solr-staging5, ip is 128.112.200.116
        - lib-solr-staging6, ip is 128.112.200.153
    - Solr logs are in `/solr/logs`
        - try grepping for our servers like `ag 128.112 solr.log`
    - Solr prod IPs
        - lib-solr-prod4 ip is 128.112.200.170
        - lib-solr-prod5 ip is 128.112.200.169
        - lib-solr-prod6 ip is 128.112.200.168
    - Zookeeper machines are
        - lib-zk-staging1, running Ubuntu 18.04 (Bionic), zk 3.4.10
        - lib-zk-staging2, running Ubuntu 18.04 (Bionic), zk 3.4.10
        - lib-zk-staging3, running Ubuntu 18.04 (Bionic), zk 3.4.10
        - lib-zk-prod1, running Ubuntu 18.04 (Bionic),
        - lib-zk-prod2, running Ubuntu 18.04 (Bionic)
        - lib-zk-prod3, running Ubuntu 18.04 (Bionic), zk 3.4.10
    - `echo "status" | nc localhost 2181 | head -n 1`
    - All zookeeper staging boxes run version 3.4.10-3–1

- Action taken: Moved the zk-staging1, 2, 3 boxes to the same VMs as the solr staging boxes (A cluster). Put the zk-staging 4, 5, 6 boxes back where they were before we moved them yesterday.
- Next steps: Alicia will confirm the solr staging 4, 5, 6 and zk staging 1, 2, 3 boxes are co-located, and stop / start all the services. (Done)
- We will check tomorrow.

Nov. 30, 2023

It's still broken this morning. New round of ideas.

- Could it be running out of temp space?
- Could we have some ports closed, so that Solr can connect to Zookeeper but the Zookeepers can't talk to each other??? According to this post, we need ports 2181, 2888, and 3888 open.
    - documentation of these ports? (see link above - it's Person on the Internet, so may not be reliable, but that's where I found it)
- I found some kind of interesting solr logs when searching for `closing socket` and other related things, but they aren't frequent enough to explain all the outages. may only be related to data center moves, e.g.
    - https://app.datadoghq.com/logs?query=service%3Asolr%20host%3A%28lib-solr-staging6%20OR%20lib-solr-staging4%20OR%20lib-solr-staging5%29%20zookeeperconnection%20&cols=host%2Cservice&index=%2A&messageDisplay=inline&refresh_mode=sliding&stream_sort=desc&viz=stream&from_ts=1698763026604&to_ts=1701355026604&live=true
- Try changing solr staging GC settings to match prod settings? -- I'm not sure if we went through with this or aborted
- Try going down to one node for the collection, longer-term
    - I think we tried this briefly but we got hung up on deleting a collection that had multiple nodes and maybe didn't get past that part.
- Look into timeout settings for solr's connection to zk?
- If theory continues to be that GC (maybe GC is triggered by deletion) is stopping the world, and then the solr box's connection to zookeeper is timing out, then reducing the nodes, changing GC, and/or adjusting the timeout might all be relevant options.
- Look at system logs, see if there's anything there that suggests a cause?

Hector looked at this issue and theorized that the problem is caused by the failure of the previous reindex job to finish, which in turn is caused by a failure in fetching data from DataSpace. See https://github.com/pulibrary/pdc_discovery/issues/485. See notes below.

Dec 1, 2023

The DataSpace indexing was not the culprit. The issue recurs even when we disable the DataSpace reindex.

I ran a bunch of other commands manually from the PDC Discovery box to try to see what commands we could use to force the reindex and it looks like any command that tries to modify the Solr collections (DELETE, CREATE) fails, but commands that just read the collections (LIST, STATUS) work OK.

Below are the details of the commands that I tried yesterday and today:

```
Looks like the reindex gets stuck and then subsequence reindex never really
start:

deploy@pdc-discovery-staging1:/opt/pdc_discovery/current$ grep "Indexing"
log/staging.log
I, [2023-11-30T00:00:36.057221 #738840]  INFO -- : Indexing: Created a new
collection for writing:
http://lib-solr8-staging.princeton.edu:8983/solr/pdc-discovery-staging-1
I, [2023-11-30T00:00:36.057376 #738840]  INFO -- : Indexing: Fetching PDC
Describe records
I, [2023-11-30T00:00:36.057775 #738840]  INFO -- : Indexing: Harvesting and
indexing PDC Describe data started
I, [2023-11-30T00:06:59.830116 #738840]  INFO -- : Indexing: Harvesting and
indexing PDC Describe data completed
I, [2023-11-30T00:06:59.830258 #738840]  INFO -- : Indexing: Fetching DataSpace
records
I, [2023-11-30T00:06:59.830488 #738840]  INFO -- : Indexing: Harvesting and
indexing DataSpace research data collections started

Notice that the DataSpace reindex started at 00:06:59 never finished.
A complete reindex will also log:

      Indexing: Harvesting and indexing DataSpace research data collections
      completed
      Indexing: Fetching completed
      Indexing: Updated Solr to read from the new collection:
      pdc-discovery-staging/ -> ...

and then the reindex kicked in at 00:30:03 but I guess since the previous one
has not finished Solr is blocking our second reindex.

I, [2023-11-30T00:30:03.209589 #745144]  INFO -- : Indexing: Research Data
indexing started
```

```
I, [2023-11-30T01:00:03.163854 #751255]  INFO -- : Indexing: Research Data
indexing started
I, [2023-11-30T01:30:03.117385 #757441]  INFO -- : Indexing: Research Data
indexing started
I, [2023-11-30T02:00:03.120285 #763553]  INFO -- : Indexing: Research Data
indexing started
I, [2023-11-30T02:30:03.176146 #769739]  INFO -- : Indexing: Research Data
indexing started
I, [2023-11-30T03:00:03.164096 #775846]  INFO -- : Indexing: Research Data
indexing started
I, [2023-11-30T03:30:03.101533 #781968]  INFO -- : Indexing: Research Data
indexing started
I, [2023-11-30T04:00:03.161018 #788080]  INFO -- : Indexing: Research Data
indexing started
I, [2023-11-30T04:30:03.088005 #794343]  INFO -- : Indexing: Research Data
indexing started
I, [2023-11-30T05:00:03.144873 #801437]  INFO -- : Indexing: Research Data
indexing started
I, [2023-11-30T05:30:03.274566 #807689]  INFO -- : Indexing: Research Data
indexing started
I, [2023-11-30T06:00:03.328160 #813801]  INFO -- : Indexing: Research Data
indexing started
```

I disabled the DataSpace indexing which is the one that seems to get stuck and the issue still happens. So it is not that.

DELETE

While the re-index is unable to start running the DELETE collection command manually times out.

From pdc-discovery-staging1

```
$ curl
'http://lib-solr8-staging.princeton.edu:8983/solr/admin/collections?action=DELE
TE&name=pdc-discovery-staging-2'
```


```
{
  "responseHeader":{
      "status":500,
      "QTime":180146},
  "error":{
      "metadata":[
      "error-class","org.apache.solr.common.SolrException",
      "root-error-class","org.apache.solr.common.SolrException"],
      "msg":"delete the collection time out:180s",
      "trace":"org.apache.solr.common.SolrException: delete the collection time
out:180s\n\tat
org.apache.solr.handler.admin.CollectionsHandler.sendToOCPQueue(CollectionsHand
```

ler.java:374)\n\tat
org.apache.solr.handler.admin.CollectionsHandler.invokeAction(CollectionsHandle
r.java:279)\n\
...
org.eclipse.jetty.util.thread.ReservedThreadExecutor$ReservedThread.run(Reserve
dThreadExecutor.java:366)\n\tat
org.eclipse.jetty.util.thread.QueuedThreadPool.runJob(QueuedThreadPool.java:781
)\n\tat
org.eclipse.jetty.util.thread.QueuedThreadPool$Runner.run(QueuedThreadPool.java
:917)\n\tat java.lang.Thread.run(Thread.java:750)\n",
"code":500}}

This Solr issue seems to be relevant:
https://issues.apache.org/jira/browse/SOLR-11301

RELOAD

$ curl
'http://lib-solr8-staging.princeton.edu:8983/solr/admin/collections?**action=RELOAD**&name=pdc-discovery-staging-2'

Reload also times out

LIST works

$ curl
'http://lib-solr8-staging.princeton.edu:8983/solr/admin/collections?action=LIST'

CLUSTERSTATUS works

$ curl
'http://lib-solr8-staging.princeton.edu:8983/solr/admin/collections?**action=CLUSTERSTATUS&collection=pdc-discovery-staging-2**'

CREATE a new collection times out

$ curl
'http://lib-solr8-staging.princeton.edu:8983/solr/admin/collections?**action=CREATE&name=pdc-discovery-staging-3**&collection.configName=pdc-discovery-staging&numShards=1&replicationFactor=2'

It looks like any command that alters the collection information (DELETE, CREATE) fails, but we can still read the list of collections.

OVERSEERSTATUS times out

Not sure what this command does, though

$ curl
'http://lib-solr8-staging.princeton.edu:8983/solr/admin/collections?action=OVERSEERSTATUS'

Why are my commands pointing to lib-solr8-staging?
lib-solr8-staging is the load balancer.
Traffic to this address is redirected to lib-solr-staging4/5/6

FORCELEADER
https://solr.apache.org/guide/6_6/collections-api.html#CollectionsAPI-forceleader

/solr/admin/collections?action=FORCELEADER&collection=pdc-discovery-staging-2&shard=shard1'

Returns "The shard already has an active leader. Force leader is not applicable"

Zookeper
https://solr.apache.org/guide/solr/latest/deployment-guide/zookeeper-utilities.html

Our zkcli.sh script is under /opt/solr/server/scripts/cloud-scripts but it is not executable

You can force it with:
$ bash zkcli.sh -zkhost lib-zk-staging1:2181 -cmd ls /

Why does this command show information for Solr 7, 8, and 9 configurations?

Wasn't there an issue in which Zookeeper boxes were mixing Solr versions a few days ago? Could this be related?

I restarted Solr on one of the boxes (lib-solr-staging4) and after I did that, the collections shown on the Solr dashboard ***reflect some of the changes that I made*** (see CREATE and DELETE above) and that had timed out and I had thought did not complete!!!!

For example:

- The pdc-discovery-staging-2 collection is gone since I deleted it
- There is a pdc-discovery-staging-3 collection that I created as a test

December 4, 2023

11:30-ish deployed a new version of PDC Discovery that points directly to a Solr node (`lib-solr-staging4`) rather than using the default load balanced URL (`lib-solr8-staging`).

Ran a few manual reindexes (`bundle exec rake index:research_data`) from `pdc-discovery-staging1` to make sure it works.

I am not sure this new URL will bypass Zookeper but we'll see if we get better or last least different results with this URL.

We'll wait 24 hours to see if something changes. Fingers crossed.

December 5, 2023

Indexing pointing directly to `lib-solr-staging4` also failed after a while, according to Honeybadger around 7:30 PM 12/4/2023.

I do not think this configuration is bypassing Zookeeper.

From our discussion earlier this morning:

- A request to `lib-solr8-staging` is load balanced to `lib-solr-staging4/5/6` by Nginx.
- When does Zookeeper kick in?
    - We think that for writes Zookeeper makes sure the data is written in at least 2 shards.
    - We think that for reads if the box selected by the load balancer does not have a shard for the collection Zookeeper might request the read from another box that does have data for the collection.

December 8, 2023

Restarted Solr in lib-solr-stating4.

The restart got rid of a test collection that I had created (named `hector`) and it also created another test collection (`pdc-discovery-staging-3`) that I had created before but had not shown until after the restart. This is all good and expected, I noted similar behavior on December 1 (see above).

> What is remarkable is that the new collection (`pdc-discovery-staging-3`) although it has no documents whatsoever, has one shard only (on `lib-solr-staging5`) and never has been used for anything it cannot be deleted either!!! WTF???
>
> I guess this adds to our suspicion that the issue is with the way Zookeper is managing the nodes since this collection had no documents and was never used by any system. Yet, it was under the purview of Zookeper and trying to delete it was blocked for what it still is an unknown reason.

I restarted Solr in all 3 boxes lib-solr-stating4/5/6 and then I was able to delete `pdc-discovery-staging-3`

Playing with Zookeeper to see if something pop up :)

SSH to pulsys@lib-solr-staging4

$ cd /opt/solr/server/scripts/cloud-scripts
$ bash zkcli.sh -zkhost lib-zk-staging1:2181 -cmd ls /solr8/
$ bash zkcli.sh -zkhost lib-zk-staging1:2181 -cmd ls /solr8/collections/pdc-discovery-staging-2

This seems to “ls” the files for this collection. The content of some of these files (e.g. default_schema) is also included in this list. Notice that if there are JAR files in the collection (e.g. to handle the CJK indexing) the content of the JAR file is also included.

Alicia and Hector noted that there is a Zookeeper log file on each of the Zookeeper boxes (`lib-sk-staging1/2/3`) and there are some errors logged there:

```
cd /var/log/zookeeper
grep "Error" zookeeper.log
```

Not sure the significance of those errors, but probably worth checking.

There seem to be errors also in the Zookeeper in production, but not sure how similar they are.