Upgrading Solr Server for DSpace

DSpace 6_x is still using Solr server 4.10, which is no longer maintained. We need to upgrade. As of 14-Sep-2018, Solr 7 is STABLE, 6.4 is LTS (bug fixes only), and all previous versions are EOL. So we should move up at least to 6.4, and arguably to 7. This will enforce some changes in the way that Solr is deployed for DSpace.

The need

Starting in Solr 5, the server is shipped as a stand-alone application with an internal Jetty container. DSpace has been building and installing its own copy to run alongside itself in the same container (typically Tomcat). We can no longer do that. Solr must become a DSpace prerequisite, like Java or the DBMS. A new site will need to have a Solr instance (perhaps empty, perhaps already used for other things) up and running before deploying DSpace. An existing site will need to designate or create a stand-alone instance of Solr and make its existing DSpace cores available to that instance, when upgrading DSpace to a version which requires post-4x Solr. It is not yet known whether existing 4x cores can be migrated as-is.

Setting up stand-alone Solr is rather easy. Get a copy, unpack it in a suitable location, and write a startup script for it that is compatible with your OS.

DSpace has been adding two small classes to its Solr artifact: one to start up Log4J 1.x and one to accept only connections from the local host. The Log4J starter will no longer be needed, as stand-alone Solr will have its own logging setup separate from DSpace's logging configuration. The localhost filter is a mere convenience and can be abandoned, or we could make that single class into a JAR artifact and continue to provide it so that sites can choose to drop it into their Solr instances. It may be better to use the host's firewall support or the container's own access controls to screen Solr from hosts which have no legitimate need to contact it. Also, recent Solr server has finer-grained access control features of its own which might be more appropriate. We need to decide whether to keep or drop the localhost filter.

Recent Solr versions also offer APIs to manage schemas. Do we want to take advantage of that in our deployment tooling, try to automate schema installation/upgrade ourselves, or just tell installers what files they must give to Solr and how to make any necessary edits?

What is our policy concerning supported Solr versions?

What should DSpace do if Solr is not listening when DSpace starts up?

So, to summarize:

- We need to upgrade to a supported version of Solr server.
- We need to drop the dspace-solr artifact, and replace it with a requirement to provide a Solr server instance by other means.
- We need to decide whether to continue to maintain the localhost filter, now that it will no longer be used by anything that DSpace installs.
- We need to explain the nature of these changes, and any migration process, to existing DSpace sites.
- We need to provide new sites with some advice about setting up a Solr instance if they need one.
- Documentation should refer the reader to the Solr community for full information.
- We need to code DSpace to deal gracefully and informatively with cases in which Solr is not ready at the first instant that DSpace wants it.

Challenges

We may need to revise our schemas. A number of field types have been replaced over three major upgrades, and the types that we use may no longer be provided.

We need to find out what, if anything, must be done to prepare existing cores for use by a vastly newer server version.

- The search core might be dropped and the repository re-indexed.
- The statistics core could be rebuilt **if** a site has kept its DSpace logs (or the extracts prepared by `bin/dspace stats-log-converter`). We also have a dump/restore tool.
- The authority core contains information that is not easily reproduced, so it may be best to dump and reload it (which may require building or adapting a tool).
- How might the other cores (oai, ???) be migrated, if need be?

This change complicates development and maintenance in cases where one wishes to use the same index content across different versions of DSpace. How can we facilitate this?

We should coordinate the Solr upgrade work with the DSpace 7 work, because of its impact on development environments.

Opportunities

Solr can be set up in a "cloud mode" which supports redundancy and scaling-out. SolrCloud also activates new APIs which we might leverage in place of code that we now provide for manipulating Solr cores. We need to decide whether the new APIs are useful enough to **require** the use of cloud mode by all sites. A Solr cloud can consist of a single instance with an internal copy of Apache ZooKeeper (which is used to orchestrate multiple instances), so it may be relatively simple for a site with modest requirements to do that and have support for APIs that we choose to use. It does mean more moving parts, including new ports to be secured. We currently use an older mode of sharding which is now considered "legacy" and probably won't get much attention in the future, so we might choose to take advantage of the Collections API now as an attempt at future-proofing DSpace's use of Solr. On the other hand, advice about running SolrCloud mostly assumes that you are running a large installation with multiple instances and multiple external ZooKeeper nodes to manage them, so there may not be much help out there for single-instance production SolrCloud sites. We should find out why the legacy mode still exists, whether it is intended to disappear some day, and whether a minimal cloud setup is significantly harder to manage than stand-alone.

- If multiple shards are already in use, how should those be migrated into the new version of Solr?
- As a Solr instance grows (specifically statistics), what scaling options exist? If Solr Cloud is the solution, how difficult will it be to make that migration later?

Our use of sharding is, well, a bit eccentric. Sharding was introduced into Solr to spread the work of searching a large index across multiple storage drives and/or host nodes, and most support for it is aimed at randomly distributing records across shards. DSpace defines shards by clumping records timestamped with the same year into a single shard, expecting the administrator to create new yearly shards as needed. Recent Solr versions implement Time Routed Aliases which we should consider as a replacement.

Our current schemas contain a lot of material that is not used by DSpace, but was just carried over from the samples provided with Solr. We could trim out anything that we don't use, and replace the voluminous commentary on what to put in a schema with concise documentation of what we have put into **our** schemas and how it is used.

As noted under Challenges, Solr has replaced some of the field types that DSpace uses. Even if not required to adopt the new types, we may benefit from doing so. We should thoroughly review our schemas and consider how they might be improved. For example: some of the defined field types are not used by any of our fields. See also DS-3691 concerning possible improvements in stemming.

Recent Solr provides APIs for schema management. We may want to make use of them. It's been suggested that we could use this for probing the condition of the required cores, and even for future schema updates.

Other issues

We may want to begin work with the "search" core, which should be simplest to work with.

This writer thinks that we should not try to give comprehensive instruction in setting up Solr.

We have existing code for discovering the version of a Solr instance and running upgrades provided by newer Solr versions, which could be adapted. https://github.com/DSpace/DSpace/blob/master/dspace/src/main/config/build.xml#L951

Sharding

DSpace optionally uses sharding to limit the size of the statistics core(s). From Slack discussion, 28-Nov-2018:

> Terry Brady 10:19
>
> I see the following use cases
>
> - One DSpace 6 stats shard
>
> - Multiple DSpace 6 stats shards (uuid migration complete)
>
> - One DSpace 5 stats shard (unmigrated)
>
> - Multiple DSpace 5 stats shards (unmigrated)
>
> - No existing cores (new install)

How do we deal with this?

CLI tools

We need to consider our existing tools related to Solr, including:

- solr-export-statistics and solr-import-statistics
- solr-reindex-statistics
- stats-log-importer
- stats-util
- solr-upgrade-statistics-6x (we will need to provide this for folks upgrading from 4x or 5x to 7x)

Docker

- We will need to customize the docker compose files for DSpace 7 to create an external solr instance

Installing/upgrading DSpace's Solr cores

Before plunging into work to make DSpace use the Solr APIs to manage its cores: What's the Simplest Thing That Could Work? We could simply document where to find the current core configurations in DSpace, and instruct the installer to copy them to a place where Solr will find and use them. We could provide some general hints about how to find the destination of these files. Besides being simple, this handles the case in which the people who run DSpace and the people who run Solr are not the same people and issues of access rights ensue.

TODO (not final)

- [x] Complete upgrade of client code to SolrJ 7_x.
- [x] Remove the dspace-solr artifact.
- [x] Work out manual steps for installing empty cores in a free-standing Solr (for a new installation).
- [ ]

- [ ] See what manual steps can be moved into Ant's `fresh_install` scripts.
- [ ] Determine whether schema updates are **required**.
- [ ] Create dump/restore or migration tools for indexes which cannot be recreated (statistics, authority).
- [x] Work out manual steps for copying/migrating/recreating cores with index records into a free-standing Solr.
- [ ] See what manual steps can be moved into Ant's `update` scripts. This is **only** for transition from our outdated `dspace-solr` artifact to current stock Solr.
- [ ] Document the changes to DSpace fresh installation: set up Solr separately if you don't already have it, install cores.
- [ ] Document the process for moving existing indexes to free-standing Solr during a DSpace upgrade from 6_x.

Solr Deployment Options

Option	DSpace version	Repo content	Features	Installation Process	Migration Process	Schema Update Process	Management	Notes
Deploy Solr as Docker Image	7.preview	New cores only	single server	Core created on container startup Core persisted in docker volumes	N/A	None. A fresh install is required.	N/A	
Standalone Solr	7.preview	New cores only	single server	Ant fresh install script needed	N/A	None. Schema update will not be supported until 7.0	DSpace sysadmin	
	7.0	New cores Migrated cores No shards	single server	Ant fresh install script needed Auto detection of existing core needed	Migration script needed for statistics and authority. Does this run as part of the install process or is this a maintenance script? Is this a migration process or an import process?	Manually deploy schema updates to Solr.	DSpace sysadmin	
	8.0+	New cores Migrated cores No shards	single server					TBD. Note future configuration options.
Solr Cloud	7.0	New cores Migrated cores "Time Routed Aliases" instead of shards	single or multi server	DBA creates cores and installs schemas	Migration script needed for statistics and authority. Does this run as part of the install process or is this a maintenance script?	DBA manually deploys schema updates to Solr.	DBA	
	8.0+	New cores Migrated cores "Time Routed Aliases" instead of shards	single or multi server					TBD. Note future configuration options.

Note that there may be reason to run a "degenerate" SolrCloud on a single server. Some APIs are supported only in cloud mode.

Related Tickets and Pull Requests

- DS-3691 - Consider change to SOLR schema to improve search stemming (text --> textgen) VOLUNTEER NEEDED Improve search stemming
- https://github.com/DSpace/DSpace/pull/2058 - Upgrade Solr Client
- ~~DS-4066~~ - Solr Statistics Schema is invalid/inaccurate after migrating from IDs to UUIDs CLOSED Upgrade statistics from id to uuid