SOLR Statistics Maintenance

DSpace Log Converter and DSPACE Log Importer

With the release of DSpace 1.6, new statistics software component was added. The use of Solr for statistics in DSpace makes it possible to have a database of statistics. With this in mind, there is the issue of the older log files and how a site can use them. The following command process is able to convert the existing log files and then import them for Solr use. The user will need to perform this conversion only once.

The Log Converter program converts log files from dspace.log into an intermediate format that can be inserted into Solr.

Command used:	`[dspace]/bin/dspace stats-log-converter`
Java class:	org.dspace.statistics.util.ClassicDSpaceLogConverter
Arguments short and long forms):	Description
`-i` or `--in`	Input file
`-o` or `--out`	Output file
`-m` or `--multiple`	Adds a wildcard at the end of input and output, so it would mean if `-i dspace.log -m` was specified, dspace.log* would be converted. (i.e. all of the following: *dspace.log, dspace.log.1, dspace.log.2, dspace.log.3,* etc.)
`-n` or `--newformat`	If the log files have been created with DSpace 1.6 or newer
`-v` or `--verbose`	Display verbose output (helpful for debugging)
`-h` or `--help`	Help

The command loads the intermediate log files that have been created by the aforementioned script into Solr. Please note that after importing event data, you need to update bitstream view events in the solr index to include the bundleName with `[dspace]/bin/dspace stats-util -b`

Command used:	`[dspace]/bin/dspace stats-log-importer`
Java class:	org.dspace.statistics.util.StatisticsImporter
Arguments (short and long forms):	Description
`-i` or `--in`	input file
`-m` or `--multiple`	Adds a wildcard at the end of the input, so it would mean dspace.log* would be imported
`-s` or `--skipdns`	To skip the reverse DNS lookups that work out where a user is from. (The DNS lookup finds the information about the host from its IP address, such as geographical location, etc. This can be slow, and wouldn't work on a server not connected to the internet.)
`-v` or `--verbose`	Display verbose ouput (helpful for debugging)
`-l` or `--local`	For developers: allows you to import a log file from another system, so because the handles won't exist, it looks up random items in your local system to add hits to instead.
`-h` or `--help`	Help

Although the DSpace Log Convertor applies basic spider filtering (googlebot, yahoo slurp, msnbot), it is far from complete. Please refer to Filtering and Pruning Spiders for spider removal operations, after converting your old logs.

Filtering and Pruning Spiders

Command used:	`[dspace]/bin/dspace stats-util`
Java class:	org.dspace.statistics.util.StatisticsClient
Arguments (short and long forms):	Description

`-b` or `--reindex-bitstreams`	Reindex the bitstreams to ensure we have the bundle name
`-r` or `--remove-deleted-bitstreams`	While indexing the bundle names remove the statistics about deleted bitstreams
`-u` or `--update-spider-files`	Update Spider IP Files from internet into `[dspace]/config/spiders`. Downloads Spider files identified in `dspace.cfg` under property `solr.spiderips.urls`. See Configuration settings for Statistics
`-f` or `--delete-spiders-by-flag`	Delete Spiders in Solr By isBot Flag. Will prune out all records that have `isBot:true`
`-i` or `--delete-spiders-by-ip`	Delete Spiders in Solr By IP Address, DNS name, or Agent name. Will prune out all records that match spider identification patterns.
`-m` or `--mark-spiders`	Update isBot Flag in Solr. Marks any records currently stored in statistics that have IP addresses matched in spiders files
`-h` or `--help`	Calls up this brief help table at command line.

Notes:

The usage of these options is open for the user to choose. If you want to keep spider entries in your repository, you can just mark them using "`-m`" and they will be excluded from statistics queries when "`solr.statistics.query.filter.isBot = true`" in the `dspace.cfg`. If you want to keep the spiders out of the solr repository, just use the "`-i`" option and they will be removed immediately.

Spider IPs are specified in files containing one pattern per line. A line may be a comment (starting with "#" in column 1), empty, or a single IP address or DNS name. If a name is given, it will be resolved to an address. Unresolvable names are discarded and will be noted in the log.

There are guards in place to control what can be defined as an IP range for a bot. In `[dspace]/config/spiders`, spider IP address ranges have to be at least 3 subnet sections in length 123.123.123 and IP Ranges can only be on the smallest subnet [123.123.123.0 - 123.123.123.255]. If not, loading that row will cause exceptions in the dspace logs and exclude that IP entry.

Spiders may also be excluded by DNS name or Agent header value. Place one or more files of patterns in the directories `[dspace]/config/spiders/domains` and/or `[dspace]/config/spiders/agents`. Each line in a pattern file should be either empty, a comment starting with "#" in column 1, or a regular expression which matches some names to be recognized as spiders.

Export SOLR records to intermediate format for import into Elastic Search

Command used:	`[dspace]/bin/dspace stats-util`
Java class:	org.dspace.statistics.util.StatisticsClient
Arguments (short and long forms):	Description
`-e` or `--export`	Export SOLR view statistics data to usage statistics intermediate format

This exports the records to dspace / temp / usagestats_0.csv. This will chunk the files at 10,000 records to new files. This can be imported with stats-log-importer to SOLR or stats-log-importer-elasticsearch to Elastic Search.

Export SOLR statistics, for backup and moving to another server

Command used:	`[dspace]/bin/dspace solr-export-statistics`
Java class:	org.dspace.util.SolrImportExport
Arguments (short and long forms):	Description
- i or - -index-name	optional, the name of the index to process. "statistics" is the default
`-l or --last` *`integer`*	optionally export only *integer* many days worth of statistics
`-d or --directory`	optional, directory to use for storing the exported files. By default, `[dspace]/solr-export` is used. If that is not appropriate (due to storage concerns), we recommend you use this option to specify a more appropriate location.
- f or - -force-overwrite	optional, overwrite export file if it exists (DSpace 5.7 and later)

Import SOLR statistics, for restoring lost data or moving to another server

Command used:	`[dspace]/bin/dspace solr-import-statistics`
Java class:	org.dspace.util.SolrImportExport
Arguments (short and long forms):	Description
- i or - -index-name	optional, the name of the index to process. "statistics" is the default

`-c or --clear`	optional, clears the contents of the existing stats core before importing
`-d or --directory`	optional, directory which contains the files for importing. By default, `[dspace]/solr-export` is used. If that is not appropriate (due to storage concerns), we recommend you use this option to specify a more appropriate location.

Reindex SOLR statistics, for upgrades or whenever the Solr schema for statistics is changed

Command used:	`[dspace]/bin/dspace solr-reindex-statistics`
Java class:	org.dspace.util.SolrImportExport
Arguments (short and long forms):	Description
-i or --index-name	optional, the name of the index to process. "statistics" is the default
`-k or --keep`	optional, tells the script to keep the intermediate export files for possible later use (by default all exported files are removed at the end of the reindex process).
`-d or --directory`	optional, directory to use for storing the exported files (temporarily, unless you also specify `--keep`, see above). By default, `[dspace]/solr-export` is used. If that is not appropriate (due to storage concerns), we recommend you use this option to specify a more appropriate location. Not sure about your space requirements? You can estimate the space required by looking at the current size of `[dspace]/solr/statistics`
-f or --force-overwrite	optional, overwrite export file if it exists (DSpace 5.7 and later)

NOTE: `solr-reindex-statistics` is safe to run on a live site. The script stores incoming usage data in a temporary SOLR core, and then merges that new data into the reindexed data when the reindex process completes.

Routine Solr Index Maintenance

Command used:	`[dspace]/bin/dspace stats-util`
Java class:	org.dspace.statistics.util.StatisticsClient
Arguments (short and long forms):	Description
`-o` or `--optimize`	Run maintenance on the SOLR index. Recommended to run daily, to prevent your servlet container from running out of memory

Notes:

The usage of this this option is strongly recommended, you should run this script daily (from crontab or your system's scheduler), to prevent your servlet container from running out of memory.

Solr Sharding By Year

Command used:	`[dspace]/bin/dspace stats-util`
Java class:	org.dspace.statistics.util.StatisticsClient
Arguments (short and long forms):	Description
`-s` or `--shard-solr-index`	Splits the data in the main core up into a separate solr core for each year, this will upgrade the performance of the solr.

Notes:

Yearly Solr sharding is a routine that can drastically improve the performance of your DSpace SOLR statistics. It was introduced in DSpace 3.0 and is not backwards compatible. The routine decreases the load created by the logging of new usage events by reducing the size of the SOLR Core in which new usage data are being logged. By running the script, you effectively split your current SOLR core, containing all of your usage events, into different SOLR cores that each contain the data for one year. In case your DSpace has been logging usage events for less than one year, you will see no notable performance improvements until you run the script after the start of a new year. Both writing new usage events as well as read operations should be more performant over several smaller SOLR Shards instead of one monolithic one.

It is highly recommended that you execute this script once at the start of every year. To ensure this is not forgotten, you can include it in your crontab or other system scheduling software. Here's an example cron entry (just replace [dspace] with the full path of your DSpace installation):

```
# At 12:00AM on January 1, "shard" the DSpace Statistics Solr index.  Ensures each year has its own Solr
index - this improves performance.
0 0 1 1 * [dspace]/bin/dspace stats-util -s
```

You MUST restart Tomcat after sharding

After running the statistics shard process, the "View Usage Statistics" page(s) in DSpace will **not** automatically recognize the new shard.

Restart tomcat to ensure that the new shard is recognized & included in usage statistics queries.

Repair of Shards Created Before DSpace 5.7

If you ran the shard process before upgrading to DSpace 5.7 or DSpace 6.1, the multi-value fields such as owningComm and onwningColl are likely be corrupted. Previous versions of the shard process lost the multi-valued nature of these fields. Without the multi-valued nature of these fields, it is difficult to query for statistics records by community / collection / bundle.

You can verify this problem in the solr admin console by looking at the owningComm field on existing records and looking for the presence of "\\," within that field.

The following process may be used to repair these records.

1. Backup your solr statistics-xxxx directories while tomcat is down.
2. Backup and delete the contents of the dspace-install/solr-export directory
3. For each "statistics-xxxx" shard that exists, export the repository

```
dspace solr-export-statistics -i statistics-xxxx -f
```

4. Run the following to repair records in the dspace-install/solr-export directory

```
for file in *
do
sed -E -e "s/[\\]+,/,/g" -i $file
done
```

5. For each shard that was exported, run the following import

```
dspace solr-import-statistics -i statistics-xxxx -f
```

If you repeat the query that was run previously, the fields containing "\\," should now contain an array of owning community ids.

Shard Naming

Prior to the release of DSpace 5.7, the shard names created were off by one year in timezones with a positive offset from GMT.

Shards created subsequent to this release may appear to skip by one year.

See blocked URLDS-3437 - When sharding statistics, the destination shard name is off by one year CLOSED

Technical implementation details

After sharding, the SOLR data cores are located in the [dspace.dir]/solr directory. There is no need to define the location of each individual core in solr.xml because they are automatically retrieved at runtime. This retrieval happens in the *static* method located in the*org.dspace.statistics.SolrLogger* class. These cores are stored in the *statisticYearCores* list each time a query is made to the solr these cores are added as shards by the *addAdditionalSolrYearCores* method. The cores share a common configuration copied from your original *statistics* core. Therefore, no issues should be resulting from subsequent ant updates.

The actual sharding of the of the original solr core into individual cores by year is done in the *shardSolrIndex* method in the *org.dspace.statistics.SolrLogger* class. The sharding is done by first running a facet on the time to get the facets split by year. Once we have our years from our logs we query the main solr data server for all information on each year & download these as csv's. When we have all data for one year we upload it to the newly created core of that year by using the update csvhandler. One all data of one year has been uploaded that data is removed from the main solr (by doing it this way if our solr crashes we do not need to start from scratch).