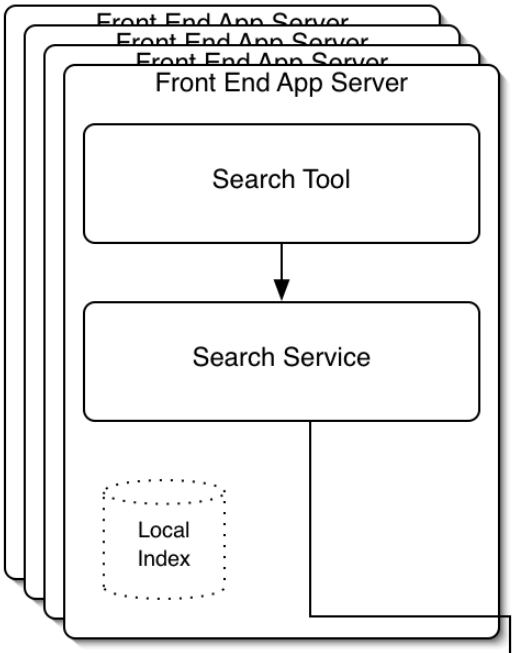
SearchServerImpl

Search Server Implimentation

In large deployments there is a need to have a sedicated number of nodes operating as search servers so as to reduce local disk requirements per node. If there is 1TB of content, and 100G of index, ten each node would require 100G of local storage and a futher 100G of shared storage. To eliminate this requirement a search node can be configured to use a Search Service (become a Search Client) provided by annother Sakai instance operating as a Search Service.

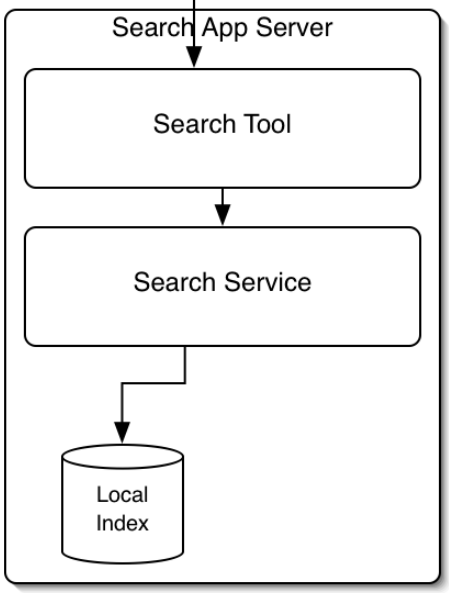


Central Search Service

Index On One Search Node
Front End Nodes use
Search Service which redirects search via HTTP REST to Search App Server. Search App Server returns Hits and index information.
Content and further processing performed on Front End App server.

If there is 1 Search App Server, Local Filesystem Index can be used.

If there are more than one Search App server, a clustered Index can be used.



Implementation

Front End:
Change the Search Service unser the API to redirect to HTTP REST and decode the result. (4 Hours)
Back End:
Generate a new XML View of the RSS based Feed with an additional User/Context parameter. (4 Hours)

Search Client

- The search client is configured to use a search server.
- It has no local index and does not take part in any indexing operations
- When a user performs a search, the tool operates the same.
- The Search Service Component serializes the request into an HTTP POST which it sends to the Search Service.
    - This includes the current user id and a secure hash of the request.
- The Search Tool on the Search Server takes the request and processes it locally serializing the raw results back out as XML.
    - The XML only contains the references and index information, no content or highlighting
- The Search Service Component on the client adds the content and highlighting and displays the results to the user.

Search Server

- The search server is configured with an index and the HTTP POST url exposed to accept request from search clients.
- The search server recieves request and passes the information to the Search Service Component.
- The search Service Component decodes the POST content, performin a secure hash to check the request.
    - This may optionally contain a shared secret.
- The Search Service Component to performs the search as the user specified in the request, performing security filtering.
- The result (which is a sublist) is then serialized to XML and sent back as the response.

Error Conditions.

- Any errors are converted to a stack trace and then serialized to XML and sent back to the Search Client for processing and reporting.

This feature is available post SVN r20036 configruation settings are detailed on the search Home page, duplicated here

Configuration

Search Server/Search Client

A new feature post r20035 (2.4) allows you to specify if a node is a search server or a search client. Search Servers have a copy of the index and provide a HTTP-XML Search web service to Search clients. Search Clients do not have a copy of the index (for search purposes) and use the configured Search Service to perform the search operation. For mere details of the impl see SearchServerImpl

- searchServer@org.sakaiproject.search.api.SearchService=true

will make the node in question a search service and expose an HTTP-XML Web service to other clients. This service is relatively insecure and can be made more secure using a shared Ket, but should not be exposed outside the cluster.

- sharedKey@org.sakaiproject.search.api.SearchService=<some shared secret>

will set the shared key which must be the same on both the search server and its clients.

On the client you must indicate which search service to use with a URL.

- searchServerUrl@org.sakaiproject.search.api.SearchService=http://<host>:<port>/sakai-search-tool/xmlsearch/

Additionally you will want to stop the client from participating in index building with

- search.indexbuild=false

So the server config might be

```
searchServer@org.sakaiproject.search.api.SearchService=true
sharedKey@org.sakaiproject.search.api.SearchService=SharedKetToStopSecurity6423134Leaks

# If you only have 1 node acting as a dedicated search server you may also want to
# disable Shared Segments (but read the warnings above first)
#
# localSegmentsOnly@org.sakaiproject.search.api.SearchService.SegmentStore=true
```

and the client(s)

```
searchServerUrl@org.sakaiproject.search.api.SearchService=http://<host>:<port>/sakai-search-tool/xmlsearch/
search.indexbuild=false
sharedKey@org.sakaiproject.search.api.SearchService=SharedKetToStopSecurity6423134Leaks
```