Registering Search Interface to SAS® Content as Google OneBox Module

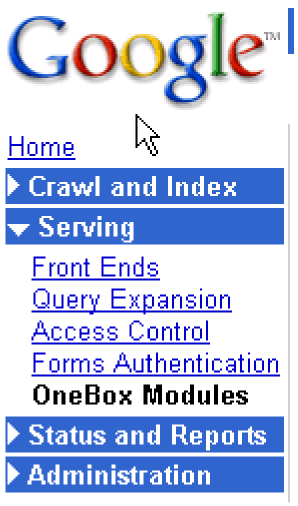
Search Interface to SAS Content supports two kinds of search results:

- Reports search - supports searching of SAS BI Dashboard 4.3 (and later), SAS Web Report Studio reports, Stored Process reports, and images.
- Data search - supports searching of Information Maps

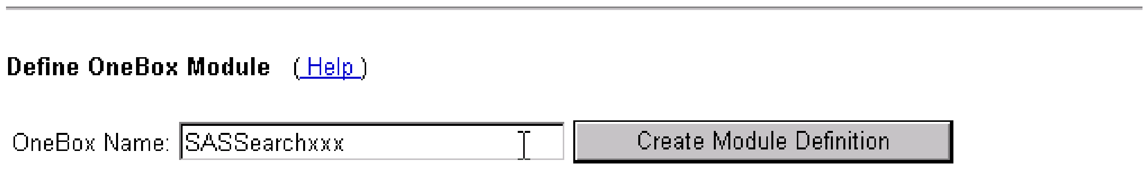
To search both types of content together, Search Interface to SAS Content will have to be registered as two separate OneBox modules. The sections below describe the steps required to use the Google Search Appliance (GSA) Administrative User Interface to perform the registration.

Define the OneBox Module to the GSA

Go to the OneBox Module definition screen in the Serving section of the interface.



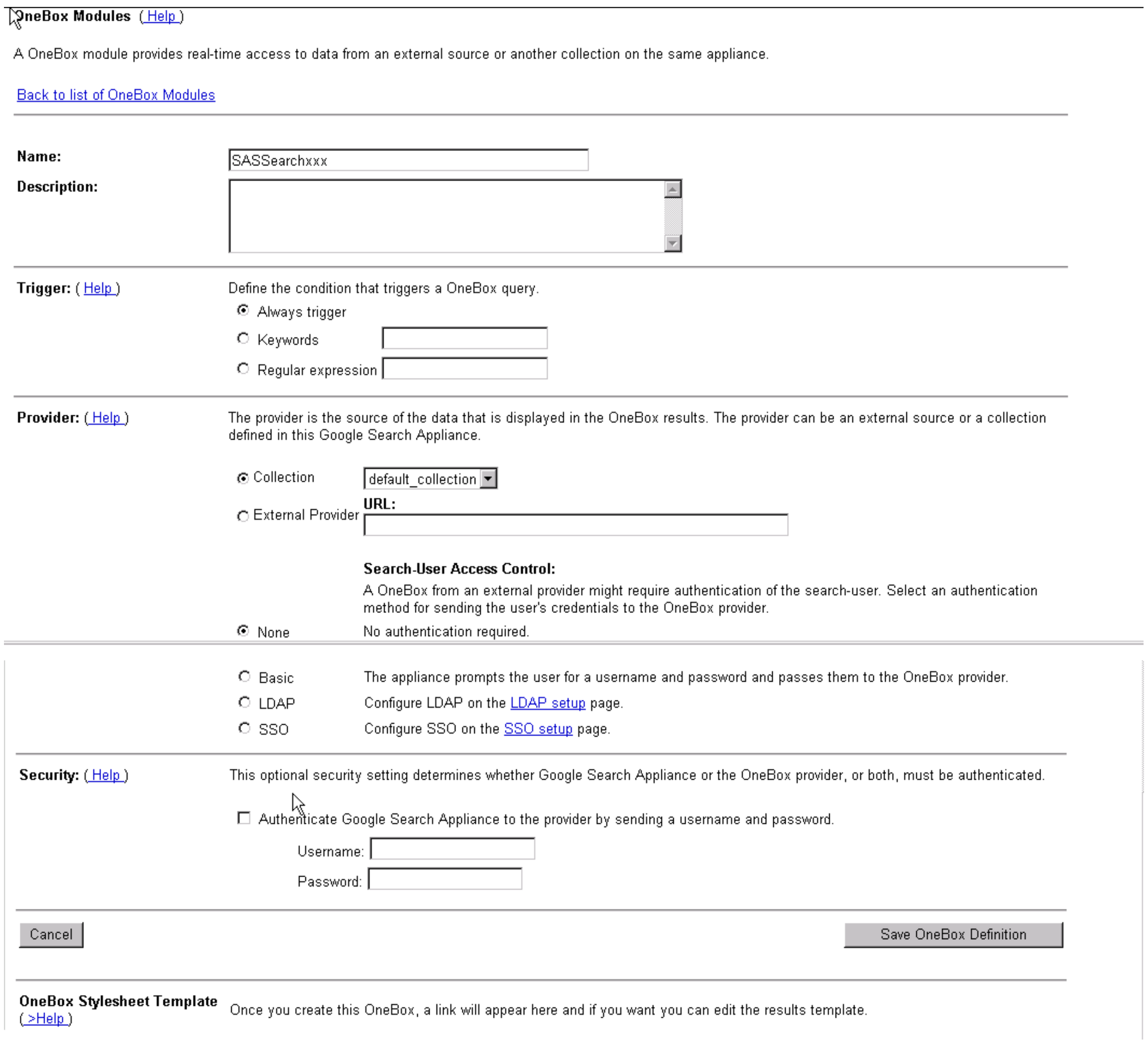
At the bottom of this page, there will be an entry form to create a new OneBox module definition. Enter any name.



OneBox Name: SASSearchxxx

Suggestion: Select a name that is generic enough to cover the triggers and items to be searched, but unique enough to be able to easily identify this module later when selecting it from a list.

After creating the OneBox module definition, you will be brought to a screen to enter the details about the module.


OneBox Modules (Help)

A OneBox module provides real-time access to data from an external source or another collection on the same appliance.

Back to list of OneBox Modules

Name: SASSearchxxx

Description:

Trigger: (Help)
Define the condition that triggers a OneBox query.
[✓ Always trigger]
Keywords
Regular expression

Provider: (Help)
The provider is the source of the data that is displayed in the OneBox results. The provider can be an external source or a collection defined in this Google Search Appliance.
[✓ Collection] default_collection
External Provider **URL:**

Search-User Access Control:
A OneBox from an external provider might require authentication of the search-user. Select an authentication method for sending the user's credentials to the OneBox provider.

[✓ None]	No authentication required.
Basic	The appliance prompts the user for a username and password and passes them to the OneBox provider.
LDAP	Configure LDAP on the LDAP setup page.
SSO	Configure SSO on the SSO setup page.

Security: (Help)
This optional security setting determines whether Google Search Appliance or the OneBox provider, or both, must be authenticated.
☐ Authenticate Google Search Appliance to the provider by sending a username and password.
Username:
Password:

Cancel | Save OneBox Definition

OneBox Stylesheet Template (>Help)
Once you create this OneBox, a link will appear here and if you want you can edit the results template.

Fill in the following fields.

Trigger

The only supported option at this point is Regular expression. In the **Regular expression** field, a regular expression should be entered. When the input search matches the regular expression, this module will be called. The regular expression must contain a "parenthesized" field which will be passed as the search term to the module.

Note: *Regular expression processing can be expensive, so try to keep the regular expression simple.*

Examples:

`(.*) report` – would match any search phrase that has the word report in it, and it would pass as the search string the phrase before that word. "sales report" would search for reports that have the word "sales" in it. "new sales report" would search for reports that have the phrase "new sales" in it.

`^report (.*)` – would match any search phrase where the first word in the search is "report", and it would pass as the search string the phrase after that word. "report sales" would search for reports that have the word "sales" in it. "report new sales" would search for reports that have the phrase "new sales" in it. Note that this definition is equivalent to using the keyword trigger option; however, the keyword trigger option does not pass the appropriate phrase to the module, and thus cannot be used as a replacement for this syntax.

`^(.*sales.*)` – would match any search phrase that has the word "sales" in it. In addition, it will pass the entire entered search string to the module. Thus, if the user enters "new sales in US", that entire phrase will be passed, and only results that match the entire phrase will be returned.

`^promotions|promotion(.*)` – would match any search phrase where the first word in the search is either "promotion" or "promotions", and it would pass the search string the phrase after that word.

Provider

In the provider section, select **External Provider** and enter a URL that points to the location where you deployed the `sas.searchsas.ear` file.

For example, if you have deployed the EAR file to a Web server on yyy.mycompany.com that is listening on port 8080, you would enter

- for reports:

  ```
  http://yyy.mycompany.com:8080/SASSearchService/Controller?forward
  =Search&searchType=reports
  ```

- for information maps:

  ```
  http://yyy.mycompany.com:8080/SASSearchService/Controller?forward
  =Search&searchType=data
  ```

The HTTP protocol should only be used if you are using None for the Authentication value. If you are using Basic Authentication, then it is preferable to use HTTPS (HTTP would also work but the credentials would be visible in the URL), such as:

```
https://yyy.mycompany.com:8443/SASSearchService/Controller?forward=Sear
ch&searchType=data
```

Authentication

The GSA supports multiple types of authentication for OneBox providers. At this time, only None and Basic are supported for the Search Interface to SAS OneBox provider.

None - No Authentication is required to use this module. Thus, when a user does a search with only "Public Content" selected, this module will be called. If you choose this option, then Search Interface to SAS Content provides search results for user SAS Web Anonymous user. If you want to change the user to some other user, then you will have to append the username and password in the parameter as below:

```
http://yyy.mycompany.com:8080/SASSearchService/Controller?forward=Searc
h&searchType=reports&userName=sasguest@saspw&password=xxxxxxx
```

Note: *If you provide the user name and password in the URL, all users will have the access to see all the results accessible for this user.*

Basic – The GSA will prompt the user for a user ID and password and will pass these values to the provider. When a user is doing a search, the user must have selected "Public and Secure content" to be searched for this module to be invoked. The authentication scheme allows for the results to be limited to just what that individual user is authorized to access.

Once done with the 'Security settings', click **Save OneBox Definition**. Now you land on the main page, where you can find your newly added module in the list. If you want to change/edit the 'OneBox Stylesheet Template', click on the 'Edit' link corresponding to the module listing. At the bottom of the configuration page, you will see a link of 'Edit XSL'.

Note: *When using Basic Authentication, Search Interface to SAS Content should use the HTTPS protocol to protect the user ID and password that is being passed to the provider.*

OneBox Stylesheet Template

When the results are returned from the module to the GSA, the GSA can apply an XSL style sheet to the results to format them for display. You should provide a style sheet to make the user experience more robust than is provided by the GSA's default XSL. To edit the XSL, use the **Edit XSL** link at the bottom of this area of the administrative page. The following is a sample XSL:

```
<xsl:template name="results">
 <xsl:variable name="searchLink"><xsl:value-of
select="title/urlLink"/></xsl:variable>
    <table border="0">
      <tr>
      <td>
      <xsl:if test="//IMAGE_SOURCE">
      <xsl:variable name="imageSource"><xsl:value-of
select="//IMAGE_SOURCE[1]"/></xsl:variable>
      <xsl:variable name="imageData">
      <xsl:for-each select="MODULE_RESULT">
            <xsl:choose>
            <xsl:when test="Field[@name='URI'] = $imageSource">
            <xsl:variable name="link"><xsl:value-of
select="U"/></xsl:variable>
            <xsl:choose>
                  <xsl:when test="$link=''">
                  ||||||||<xsl:value-of select="Field[@name='Name']"/>
                  </xsl:when>
                  <xsl:otherwise>
                  ||||<xsl:value-of select="$link"/>||||<xsl:value-of
select="Field[@name='Name']"/>
                  </xsl:otherwise>
            </xsl:choose>
            </xsl:when>
            <xsl:otherwise></xsl:otherwise>
            </xsl:choose>
      </xsl:for-each>
      </xsl:variable>
      <xsl:choose>
      <xsl:when test="$imageData!=''">
            <xsl:choose>
                  <xsl:when test="starts-with(normalize-
space($imageData),'||||||||')">
                        <xsl:variable name="title">
                              <xsl:choose>
                                    <xsl:when test="contains(substring-
after($imageData,'||||||||'),'||||')">
```

```
                                     <xsl:value-of select="substring-
before(substring-after($imageData,'||||||||'),'||||')"/>
                                    </xsl:when>
                                    <xsl:otherwise>
                                    <xsl:value-of select="substring-
after($imageData,'||||||||')"/>
                                    </xsl:otherwise>
                              </xsl:choose>
                        </xsl:variable>
                  <xsl:element name="img">
                              <xsl:attribute name="src"><xsl:value-of
select="$imageSource"/></xsl:attribute>
                              <xsl:attribute name="alt"><xsl:value-of
select="$title"/></xsl:attribute>
                              <xsl:attribute name="title"><xsl:value-of
select="$title"/></xsl:attribute>
                        </xsl:element>
                  </xsl:when>
                  <xsl:otherwise>
                        <xsl:variable name="link">
                        <xsl:value-of select="substring-before(substring-
after($imageData,'||||'),'||||')"/>
                        </xsl:variable>
                        <xsl:variable name="title">
                        <xsl:choose>
                              <xsl:when test="contains(substring-
after(substring-after($imageData,$link),'||||'),'||||')">
                                     <xsl:value-of select="substring-
before(substring-after(substring-
after($imageData,$link),'||||'),'||||')"/>
                              </xsl:when>
                              <xsl:otherwise>
                                    <xsl:value-of select="substring-
after(substring-after($imageData,$link),'||||')"/>
                              </xsl:otherwise>
                        </xsl:choose>
                        </xsl:variable>
                        <a><xsl:attribute name="href">
                              <xsl:value-of select="$link"/>
                              </xsl:attribute><xsl:attribute
name="target">_parent</xsl:attribute>
                              <xsl:element name="img">
                                          <xsl:attribute
name="src"><xsl:value-of select="$imageSource"/></xsl:attribute>
                                          <xsl:attribute
name="alt"><xsl:value-of select="$title"/></xsl:attribute>
                                          <xsl:attribute
name="title"><xsl:value-of select="$title"/></xsl:attribute>
                                    </xsl:element>
                        </a>
                  </xsl:otherwise>
            </xsl:choose>
            </xsl:when>
      <xsl:otherwise>
            <xsl:element name="img">
                  <xsl:attribute name="src"><xsl:value-of
select="$imageSource"/></xsl:attribute>
```

```
                  <xsl:attribute name="alt">Powered by SAS</xsl:attribute>
                  <xsl:attribute name="title">Powered by
SAS</xsl:attribute>
            </xsl:element>
      </xsl:otherwise>
      </xsl:choose>
      </xsl:if>
      </td>
                  <td>
               <!-- The oneBox style guide says to only show 3 responses
at most -->
             <xsl:for-each select="MODULE_RESULT[position() &lt;=3]">
             <xsl:variable name="Type"><xsl:value-of
select="Field[@name='Type']"/></xsl:variable>
             <xsl:choose>
                  <xsl:when test="$Type!='Document'">
                  <a><xsl:attribute name="href"><xsl:value-of
select="U"/></xsl:attribute><xsl:attribute
name="target">_parent</xsl:attribute><b> <xsl:value-of
select="Field[@name='Name']"/></b></a><br/>
                  <!-- If there is a RELATED field, then add the Similar
Reports hyperlink -->

                  <font size="-1"><xsl:value-of
select="Field[@name='Desc']"/>
                  <xsl:if test="Field[@name='RELATED']">
                  <a class="fl"><xsl:attribute name="href"><xsl:value-of
select="Field[@name='RELATED']"/></xsl:attribute><xsl:attribute
name="target">_parent</xsl:attribute> <xsl:value-of
select="Field[@name='SimilarLabel']"/></a>
                  </xsl:if>

                  </font>

                  <br/>
            </xsl:when>
            <xsl:otherwise>
            </xsl:otherwise>
            </xsl:choose>
            </xsl:for-each>
            <xsl:if test="count(MODULE_RESULT) > 3">
                         <!-- Since there are more results than can fit on
one OneBox result, issue the same query and show all the results -->
                 <!-- The search criteria is the same in all
MODULE_RESULTS, so just get it from the first one -->
                  <xsl:if test="$searchLink != ''">
                  <font size="-1"><a><xsl:attribute name="href">
                  <xsl:value-of
select="$searchLink"/></xsl:attribute><xsl:attribute
name="target">_parent</xsl:attribute><xsl:value-of
select="MODULE_RESULT[1]/Field[@name='MoreLabel']"/></a></font>
                  <br/>
                  </xsl:if>
        </xsl:if>
        </td>
      </tr>
</table>
```

```
</xsl:template>
```

Note: *There are limitations imposed by the GSA on the XSL used here. For more information, see the Google Enterprise Web site,* `www.google.com/Enterprise`.

Additional Authentication Steps

If you used a value of Basic Authentication for the Authentication option, you will need to perform an additional step in setting up a Google One-Box to use Basic Authentication.

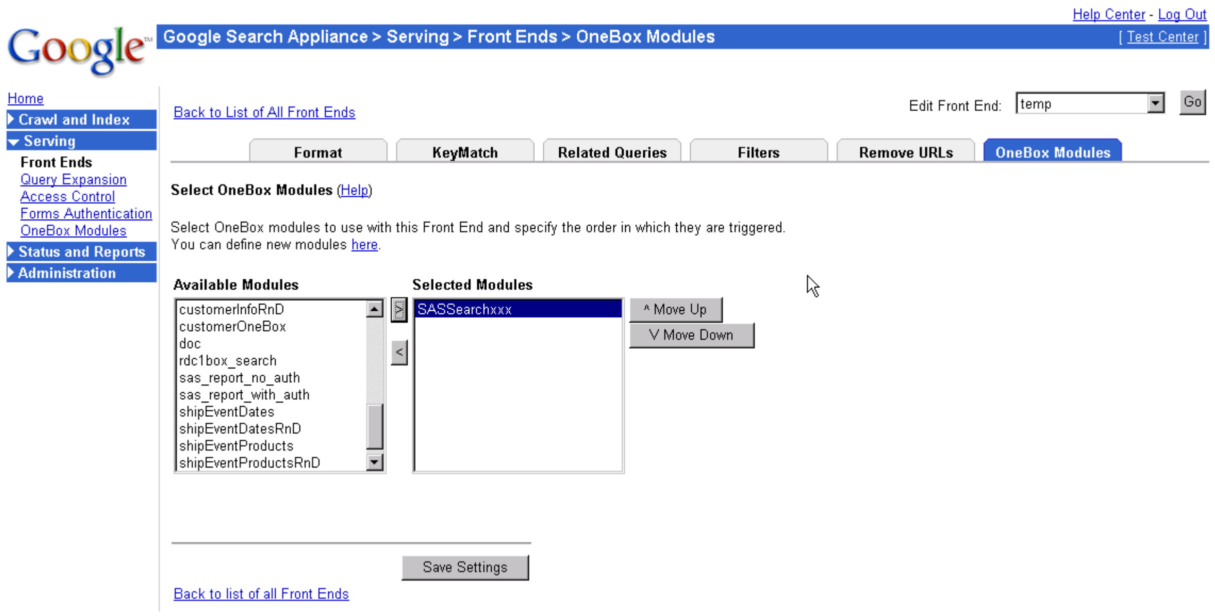
In the Google Admin UI→**Crawl and Index**→**Crawler Access**.

Make sure that the URL pattern of the One Box provider resides there, with some user ID and the "public" checkbox NOT checked. For example, if the web application is deployed to `http://yyy.mycompany.com:8080/SASSearchService`, then this same string needs to be used in the URL pattern line.

Add the OneBox Module Definition to a Front End

For the OneBox module to be called when a user does a query, the OneBox module must be attached to a Front End (for more information on what a Front End is, see `www.google.com/Enterprise`).

In the Google Admin UI→**Serving**→**Front Ends,** edit the Front End you want to add this OneBox Module to by going to the **OneBox Modules** tab, and selecting your OneBox Module from the list.



Edit Front End: temp

Select OneBox Modules (Help)

Select OneBox modules to use with this Front End and specify the order in which they are triggered.
You can define new modules here.

Available Modules

customerInfoRnD
customerOneBox
doc
rdc1box_search
sas_report_no_auth
sas_report_with_auth
shipEventDates
shipEventDatesRnD
shipEventProducts
shipEventProductsRnD

Selected Modules

SASSearchxxx

Feeding SAS Content to the Index of Google Search Appliance using Search Interface to SAS Content

Search Interface to SAS Content supports feeding SAS contents to the index of Google Search Appliance. To feed the contents follow the steps below.

Step 1 - Revise the Configuration Information in url_list.txt

To load SAS Contents to the index of Google Search Appliance, the index loading application requires configuring the Search Interface to SAS Content URL in the url_list.txt file available in the Search Interface to SAS Content installation home directory.

While configuring the URL, you can either specify the user credential in the URL, or configure to load only public reports to the index of Google Search Appliance. If you specify the user credential in the URL, the index loading application will load all contents accessible for that user to the index. If you configure to load the public reports, index loading application load all contents accessible for SAS Web Anonymous user to the index.

Note: *If you provide the user name and password in the URL, all users will have the access to see all the results accessible for this user.*

To configure the URL for public reports or for a specific user, follow the respective section below.

For public user:

Uncomment the URL in the `url_list.txt` which has `searchClient=gsafeeder` and `authType=none` parameter appended in it. Ensure that all other URLs are commented if you need to provide feed only to Google Search Appliance with public user.

For a specific user:

Uncomment the URL in the `url_list.txt` which has `searchClient=gsafeeder` and `userName=<userName>&password=<password>` parameters appended in it. Ensure that all other URLs are commented if you need to provide feed only to Google Search Appliance for a specific user.

Replace `<userName>` with the metadata username.

Replace `<password>` with the password for the specified user.

Note : *It is recommended that the password is given in the encoded format using the procedure* `pwencode`*. For example, if the password is* `Welcome123`*, execute the following command in a SAS session to encode the password:*

```
proc pwencode IN = 'Welcome123';
run;
```

Use the output produced by this proc for the password.

After you have uncommented the respective URLs for a public user or a specific user, modify the hostname and port in the URL on which Search Interface to SAS Content is deployed.

For example, if you configure the URL for a specific user with username "myuser" and password "Welcome123", and you decided to encode the password, and the hostname for Search Interface to SAS Content is `http://searchsas.mycompany.com` and the port is 8080, then the configured URL will be as follows:

```
http://searchsas.mycompany.com:8080/SASSearchService/Controller?forward
=Search&userName=myuser&password={sas002}835DA53542E057BA07B02C302BDC76
360963D41A&searchClient=gsafeeder
```

Step 3 - Run the loadindex Script with Required Parameters

After the URL has been modified in the `url_list.txt`, you can run the loadindex script available in the installation home directory of Search Interface to SAS Content. The extension of this script file will vary based on the platform on which the search interface to SAS Content is installed: `loadindex.exe` for Windows, `loadindex` for UNIX-based systems and `loadindex.rexx` for z/OS.

The loadindex script accepts the following command line arguments:

`-filename <filename>`: the name of the configuration file containing the URL of Search Interface to SAS Content. This parameter needs to be specified only if you have changed the configuration file name to a name other than `url_list.txt`.

`-gsaurl <gsaurl>:` URL for the Google Search Appliance. The following is a sample URL:

```
http://<APPLIANCE-HOSTNAME>:19900/xmlfeed
```

Replace `<APPLIANCE-HOSTNAME>` with the host name of your Google Search Appliance.

Note: *Refer Google Search Appliance's Feeds Protocol Developer's Guide to get the version specific URL for your Google Search Appliance.*

If the Google Search Appliance URL is `http://gsa.mycompany.com:19900/xmlfeed` and the configuration file is `url_list.txt` (default name), then run the following command in the command line

```
loadindex –gsaurl http://gsa.mycompany.com:19900/xmlfeed
```

Note: *You can schedule a job to load the contents to the Google Search Appliance periodically using Windows Scheduler or in crontab of your UNIX-based systems*

To get the help about the command line options for loadindex, run the following command:

Windows:

```
loadindex.exe –help
```

UNIX-based systems:

```
loadindex –help
```

z/OS:

```
loadindex.rexx –help
```

When the index has been loaded with SAS Content, you can perform a search in Google Search Appliance's Search UI. Make sure that you select a search string which matches some SAS content.

Note: *In case a blank space exists in the path for the file* `url_list.txt`*, provide the path in double quotes. For example, if the path to file is* `C:\my files\url_list.txt`*, then use the following command:*

```
loadindex -filename “C:\my files\url_list.txt”
```

Advanced Configuration Option

- While running the loadindex application, if the contents are more, you may need to increase the java heap size to avoid OutOfMemoryError. To increase the java heap size, open the `loadindex.ini` file located in the Search Interface to SAS Content installation home directory and add the following commands:

  ```
  JavaArgs_7=-Xmx512m
  JavaArgs_8=-Xms512m
  ```

  Note: *To provide more java heap size, you can change the heap size value from 512 to more appropriate value.*

- If Search Interface to SAS Content is deployed in a web-app server configured with https, then the SSL certificate has to be imported in the JRE on which the loadindex application is running.