# Settings File

The settings.json file holds environmental information (server addresses, passwords, system properties, repository settings, etc.) for your Aspire installation.

What goes in the Settings File? This is the information to include in the settings.xml Configuration File:

- Node configuration
- Security configuration
- Maven repositories
- App Bundle properties
- Applications to launch on startup

## Settings File Location

On startup, the Aspire application will automatically attempt to load the settings from the configured provider. Aspire by default uses Elasticsearch as a provider, and the settings are expected to be in the aspire-settings index.

The settings file can be uploaded by using the aspire.bat/aspire.sh file with the command "-upload_settings <absolute_settings_path>" or "-us<absolute_settings_path>" or in the case of using containers, these will be uploaded automatically on the startup.

## Structure of settings.json

The settings.json file contains sections for automatically starting system configuration files, setting Aspire system properties, and setting Apache Felix system properties. The structure is as follows:

```
{
  "settings": {
    "authentication": {
      "tokenExpiration": "30m",
      "refreshExpiration": "1h",
      "type": "Ldap",
      "ldap": {
        "server": "ldap://oldap:389",
        "authentication": "simple",
        "bindDN": "cn=admin,dc=accenture,dc=com",
        "searchBase": "dc=accenture,dc=com",
        "userDNQuery": "(uid={user})",
        "groupsHoldMembers": "true",
        "memberAttr": "uniqueMember",
        "connectTimeout": "3000",
        "roles": [
          {
            "dn": "cn=administrators,ou=Groups,dc=accenture,dc=com",
            "group": "true",
            "roles": [
              "ADMINISTRATOR"
            ]
          },
          {
            "dn": "cn=operators,ou=Groups,dc=accenture,dc=com",
            "group": "true",
            "roles": [
              "OPERATOR"
            ]
          }
        ]
      }
    },
    "configAdmin": {
      "properties": {
        "@pid": "org.apache.felix.webconsole.internal.servlet.OsgiManager",
        "property": [
          {
            "@name": "username",
            "$": "admin"
          },
```

```json
        {
          "@name": "password",
          "$": "admin"
        },
        {
          "@name": "manager.root",
          "$": "/osgi"
        }
      ]
    }
  },
  "repositories": {
    "defaultVersion": "5.0-SNAPSHOT",
    "allowAutoUpdate": "true",
    "repository": [
      {
        "@type": "distribution",
        "directory": "bundles/aspire"
      },
      {
        "@type": "maven",
        "remoteRepositories": {
          "remoteRepository": {
            "id": "stPublic",
            "url": "https://repository.sca.accenture.com/artifactory/st-snapshot/"
          }
        }
      }
    ]
  },
  "encryptionProvider": {
    "implementation": "com.accenture.aspire:aspire-encryption-provider",
    "jarName": "aspire-encryption-provider-5.0-SNAPSHOT.jar",
    "jarPath": "/bundles/aspire/",
    "className": "com.accenture.aspire.encryption.providers.AspireEncryptionProvider"
  },
  "properties": {
    "property": [
      {
        "@name": "sampleProperty1",
        "$": "http://localhost:8983"
      },
      {
        "@name": "sampleProperty2",
        "$": "false"
      },
      {
        "@name": "sampleProperty3",
        "$": "data/crawler"
      },
      {
        "@name": "sampleProperty4",
        "$": "data"
      }
    ]
  },
  "nodesProperties": {
    "worker": {
      "maxMemQueueSize": "1000",
      "queueSizeThreshold": "0.75",
      "cleanUpWaitTime": "300000",
      "cleanUpThreshold": "3600000",
      "maxEnqueueRetries": "5",
      "debug": "false",
      "appCleanUpWaitTime": "60000",
      "appCleanUpThreshold": "3600000",
      "tags": "",
      "entryProcessorBaseSleep": "200",
      "entryProcessorMaxSleep": "10000",
      "entryProcessorMaxIterations": "5",
      "entryProcessorMultiplier": "2",
```

```json
          "batchLoaderBaseSleep": "200",
          "batchLoaderMaxSleep": "10000",
          "batchLoaderMaxIterations": "5",
          "batchLoaderMultiplier": "2",
          "connectionTimeout": "60000",
          "socketTimeout": "60000",
          "maxRetries": "3",
          "proxyHost": "",
          "proxyPort": "0",
          "proxyUser": "",
          "proxyPassword": "",
          "pingFrequency": "15000",
          "nodeFailureTimeout": "30000"
        },
        "manager": {
          "scanBatchCreatorBaseSleep": "200",
          "scanBatchCreatorMaxSleep": "10000",
          "scanBatchCreatorMaxIterations": "10",
          "scanBatchCreatorMultiplier": "2",
          "processBatchCreatorBaseSleep": "200",
          "processBatchCreatorMaxSleep": "10000",
          "processBatchCreatorMaxIterations": "10",
          "processBatchCreatorMultiplier": "2",
          "crawlProgressManagerBaseSleep": "500",
          "schedulerBaseSleep": "10000",
          "maxBatches": "1000",
          "maxBatchItems": "100",
          "connectionTimeout": "60000",
          "socketTimeout": "60000",
          "maxRetries": "3",
          "proxyHost": "",
          "proxyPort": "0",
          "proxyUser": "",
          "proxyPassword": "",
          "pingFrequency": "15000",
          "nodeFailureTimeout": "30000",
          "tags": "",
          "workerRoundRobin": "false",
          "workerRoundRobinTimeout": "600000"
        }
      },
      "autoStart": {
        "application": [
          {
            "@config": "com.accenture.aspire:app-cf-bootloader"
          },
          {
            "@enable": false,
            "@config": "com.accenture.aspire:app-admin-ui"
          }
        ]
      }
    }
  }
}
```

# Auto Start Section

The "autoStart" section will automatically load applications when Aspire is initialized. It contains a simple list of application files to load, for example:

```
"autoStart": {
  "application": [
    {
      "@config": "com.accenture.aspire:app-cf-bootloader"
    },
    {
      "@enable": false,
      "@config": "com.accenture.aspire:app-admin-ui"
    }
  ]
}
```

Applications are loaded in the order specified. However, since Aspire has component-dependency checking built-in, the order of load is usually not that important.

## Both Application XML/JSON Files and App Bundles

Each application can be launched either from an application XML file or an App Bundle.

- For application XML files: The *@config* attribute should hold the file name of the Application XML/JSON file to load.

- For App Bundles: The *@config* attribute should hold the Maven coordinates of the App Bundle to start.

## Rename Auto-Started Applications

In general, the name of the application will be taken as the "default name" as specified at the top of the application.xml file.

However, you can specify other names for the configuration file using the *@name* attribute, as shown below:

```
{
  "@name": "RDBConnector2",
  "@config": "com.searchtechnologies.appbundles:cs-rdbms-connector:2.0",
  "properties": {
    "property": [
      {
        "@name": "rdbmsHasDefaults",
        "$": "false"
      },
      {
        "@name": "debug",
        "$": "true"
      }
    ]
  }
}
```

This lets you install the same App Bundle multiple times, but with different top-level names.

## Application Properties

Finally, as shown above, applications can have a nested "properties" section which holds properties that are defined just for that application. These properties can then be used with the ${propName} substitution pattern within the application.xml file.

# Repositories Section

The "repositories" identifies where to find component code to load into Aspire.

```
"repositories": {
  "defaultVersion": "5.0",
  "allowAutoUpdate": "true",
  "maxVersion": "5.0.2",
  "repository": [
    {
      "@type": "distribution",
      "directory": "bundles/aspire"
    },
      {
      "@type": "S3distribution",
      "bucketName":"s3-repo",
      "directory": "bundles/",
            "regionName":"us-east-1",
      "accessKey":"optional",
      "secretKey":"optional"
    },
    {
      "@type": "maven",
      "offline": "false",
      "localRepository": "~search/.m2/repository",
      "remoteRepositories": {
        "remoteRepository": {
          "id": "stPublic",
          "url": "https://repository.sca.accenture.com/artifactory/st-snapshot/"
        }
      }
    }
  ]
}
```

The following options are available for the repositories section:

| Property | Type | Default | Description |
| --- | --- | --- | --- |
| defaultVersion | string | LATEST | (Strongly Recommended) Specifies the default version for all artifacts for which no version is specified. Note that this defaults to a version of "LATEST" - but unfortunately, this has some odd behavior between the local and remote repositories (it only checks the local repository if the version is available on the remote repository, and the remote repository has been "scanned"). |
| allowAutoUpdate | boolean | true | (Optional) Enables updating the artifacts to the latest minor version available. The latest version depends on the version configured in the maxVersion option. |
| maxVersion | string | none | (Optional) The max version supported for the artifacts. |

There are three types of repositories that can be configured in the "repository" section:

# Distribution Repository

The Distribution Repository will load the component Jar files in a directory within your Aspire distribution, typically the "bundles/aspire" directory.

It is configured as follows:

```
{
  "@type": "distribution",
  "directory": "bundles/aspire"
}
```

On startup, Aspire will scan through the entire directory looking for bundles to load. If at any time you add new bundles (or update bundles) in this directory, then click on "Check for Updates" on the Aspire application home page. This will cause Aspire to re-scan the directory so that the new files are available. The "directory" tag identifies the directory where the bundles can be located.

# S3 Distribution Repository

The S3 bucket Distribution Repository will load the component Jar files to your Aspire distribution, typically from S3 "bundles/" directory. S3 directory must end with "/".

It is configured as follows:

```json
{
  "@type": "S3distribution",
  "bucketName":"s3-repo",
  "directory": "bundles/",
  "regionName":"us-east-1",
  "accessKey":"optional",
  "secretKey":"optional"
}
```

Upon startup, Aspire will scan the entire S3 directory, similarly to the Distribution repository, to look for bundles to load. If you add new bundles (or update existing ones) to this directory at any time, then click on "Check for Updates" on the Aspire application home page. This action will prompt Aspire to rescan the directory, making the new files available. The "directory" tag specifies the location where the bundles can be found.

# Maven Repository

The Maven Repository loads the component Jar files directly from Maven. The Maven Repository allows Aspire to share the same Jars as Eclipse and the Maven command-line program. Therefore, any newly 'install'ed or 'deploy'ed Jar file artifacts will be automatically available to Aspire.

It is configured as follows:

```json
{
  "@type": "maven",
  "offline": "false",
  "localRepository": "~search/.m2/repository",
  "remoteRepositories": {
    "remoteRepository": {
      "id": "stPublic",
      "url": "https://repository.sca.accenture.com/artifactory/st-snapshot/"
    }
  }
}
```

The following options are available for the maven repository:

| Property | Type | Default | Description |
|---|---|---|---|
| localReposi tory | string | (user home directory)/. m2 /repository | (Optional) Specifies the location of the Maven local repository, where jars will reside locally once they are downloaded from the remote repository. This is also the location where Maven "install" will install new or updated artifacts. |
| defaultVersi on | string | LATEST | (Strongly Recommended) Specifies the default version for all artifacts for which no version is specified. Note that this defaults to a version of "LATEST". Unfortunately, this has some odd behavior between the local and remote repositories: it only checks the local repository if the version is available on the remote repository, and the remote repository has been "scanned". |
| offline | boolean | false | (Optional) Specifies if the system is "offline" - in which case the Maven repository will only ever look to the local repository for artifacts, and never the remote repositories. |

## Use Specific Versions of Bundle

If required, you can force the Maven repository to give you a specific version of a bundle. This is if you don't specify it in the *factoryName* in application. xml files or in the *config* attribute in the *autoStart* section of the settings file. In the settings.json file, the bundleVersions section **must be created** within the **Repositories** section.

Normally in Aspire, if references to Maven artifacts do not give the *version*, then the *defaultVersion* (see above) is used. However, you may add a *bund leVersions* section to the settings file to give more precise control over the versions of bundles loaded. The parameters are shown below:

| Property | Type | Default | Description |
|---|---|---|---|
| bundleVersions\bundle\@groupId | String | com.accenture.aspire | (Optional) The group ID for the bundle to version |
| bundleVersions\bundle\@artifactId | String | | **Required**: The artifact ID for the bundle to version |
| bundleVersions\bundle\@version | String | | **Required**: The version of the bundle to request from Maven |

⚠

> ⚠ If a requested bundle is not configured in the *bundleVersions* section, then the *defaultVersion* (as configured above) of that bundle will be requested.

> ⚠ If the version specified is not located in Maven, an error will occur.

Example:

The following snippet will load all requested bundles at version *5.0*, except the three specified, which will be loaded at the requested version

```
"repositories": {
    "defaultVersion" : "5.2.2",
    "allowAutoUpdate" : "false",
    "bundleVersions": {
        "bundle": [
            {
                "@artifactId": "aspire-tools",
                "@groupId": "com.accenture.aspire",
                "@version": "5.0.0.2-SNAPSHOT"
            },
            {
                "@artifactId": "aspire-dbserver-source",
                "@groupId": "com.accenture.aspire",
                "@version": "5.0.0.1-SNAPSHOT"
            },
            {
                "@artifactId": "aspire-adobe-experience-source",
                "@groupId": "com.accenture.aspire",
                "@version": "5.0.0.1-SNAPSHOT"
            }
        ]
    }
}
```

## Proxy Settings

You can configure Maven remote repositories to use an HTTP proxy for outgoing communications. This is useful when your Aspire server has restricted access to the Internet, and you want to be able to fetch bundles as normal from the configured repository. To do this, add a <proxy> section to your remote repository and set the following properties:

| Property | Type | Default | Description |
|----------|------|---------|-------------|
| host | string | null | (Optional) Proxy server hostname or IP address. |
| port | string | 0 | (Optional) Proxy server port number. |
| user | string | null | (Optional) User required for authentication against the proxy server. |
| password | string | null | (Optional) Password for authenticating against the proxy server. You can encrypt it by following the instructions here. |

Example:

```
"remoteRepository": {
  "id": "stPublic",
  "url": "https://repository.sca.accenture.com/artifactory/st-snapshot/",
  "proxy": {
    "host": "127.0.0.1",
    "port": 8888,
    "user": "PROXY-USER",
    "password": "PROXY-PASSWORD"
  }
}
```

# Properties

Properties are specified as name/value pairs. For example:

```
"properties": {
  "property": [
    {
      "@name": "sampleProperty1",
      "$": "http://localhost:8983"
    },
    {
      "@name": "sampleProperty2",
      "$": "false"
    },
    {
      "@name": "sampleProperty3",
      "$": "data/crawler"
    },
    {
      "@name": "sampleProperty4",
      "$": "data"
    }
  ]
}
```

Once specified in the settings.xml file, these properties become available for use in Application XML files. Careful use of such properties will make your system configuration files portable to multiple Aspire installations without modification.

You can use these properties from the UI when configuring content sources, services, and workflow applications.

For example, you might use "http://localhost:8080" as your SOLR server on your personal laptop, but then use "http://customer.searchtechnologies.com:8983" for the production site. Using a property allows the same system configuration file to be tested on one machine and then installed on another machine without modification.

# Properties & Environment Variables in Application XML Files

Properties declared in the settings.xml file can be used in application XML files with the ${propertyName} syntax. As an example:

```
<component name="feed2Solr" subType="default" factoryName="aspire-post-xml">
  <postXsl>config/aspire2solr.xsl</solrXsl>
  <postUrl>${solrServer}/solr/update</postUrl>
</component>
```

In the above example, the "solrServer" property was defined in the settings.xml file and then referenced with ${solrServer} in the application XML file.

This property value substitution occurs automatically on the component configurations by the Component Manager. It does not require any further intervention or programming for any individual component.

The *${XXX}* syntax can also be used for substitution of environment variables and Java system properties (i.e., those defined on the command line with -Dxxx=yyy). Substitution prefers properties defined in the settings.json file. If the property is not found in the settings.json file, the system properties are checked and if still not found, the system environment is checked. Furthermore, in these versions, properties may be defined from other properties:

```
"properties": {
  "property": [
    {
      "@name": "baseDir",
      "$": "/home/user/aspire"
    },
    {
      "@name": "configDir",
      "$": "${baseDir}/cfg"
    }
  ]
}
```

**Note:** Property references for properties that are *not* in the settings.xml file will be left as-is. This allows for other configurations that use the same syntax (specifically, the Groovy Scripting component) to continue to operate properly.

## Property Escaping (for Groovy Scripts)

If you need to insert a property into a Groovy script, assigning it to a string that contains the **\\** character (such as *${aspire.home}* would), will cause Groovy to raise an error as it sees invalid escaped characters. To avoid this, you can prefix the property name with *escape:* and any **\\** characters in the contents of the property will be replaced with **\\\\**.

For example:

```
"properties": {
  "property": [
    {
      "@name": "file",
      "$": "c:\top-directory\directory\file.html"
    }
  ]
}
```

and

```
<config>
 <file>${file}</file>
 <escapefile>${escape:file}</escapefile>
 <fileattr attr="${file}">somevalue</fileattr>
 <escapefileattr escapeattr="${escape:file}">somevalue</escapefileattr>
</config>
```

expands to

```
<config>
 <file>c:\top-directory\directory\file.html</file>
 <escapefile>c:\\top-directory\\directory\\file.html</escapefile>
 <fileattr attr="c:\top-directory\directory\file.html">somevalue</fileattr>
 <escapefileattr escapeattr="c:\\top-directory\\directory\\file.html">somevalue</escapefileattr>
</config>
```

# Properties for Applications

You can specify properties that apply to a specific application (rather than the properties above, which apply to all components).

```
<autoStart>
  <application config="config/system.xml">
    <properties>
      <property name="debug">true</property>
      <property name="managerExternalRDB">false</property>
      <property name="managerRDB">CSRDB</property>
    </properties>
  </application>
  <application config="com.searchtechnologies.appbundles:cs-manager:4.0">
    <properties>
      <property name="debug">true</property>
      <property name="managerExternalRDB">false</property>
      <property name="managerRDB">CSRDB</property>
      <property name="managerExternalJDBCUrl"></property>
      <property name="managerExternalJDBCDriverJar"></property>
      <property name="managerExternalJDBCUser"></property>
      <property name="managerExternalJDBCPassword"></property>
    </properties>
  </application>
</autoStart>
```

These properties are passed to all components (and only those components) that exist "under" the component manager. If the same property names are used at both the global level and the component manager level, the component manager definition will be used for components "under" that manager, whilst the global value would be used for other components.

# Apache Felix Configuration

Some Apache Felix configuration parameters can also be placed in the settings.xml file, as follows:

```
"configAdmin": {
  "properties": {
    "@pid": "org.apache.felix.webconsole.internal.servlet.OsgiManager",
    "property": [
      {
        "@name": "username",
        "$": "admin"
      },
      {
        "@name": "password",
        "$": "admin"
      },
      {
        "@name": "manager.root",
        "$": "/osgi"
      }
    ]
  }
}
```

Although, before using this approach, check to see if these parameters can be stored in the Apache Felix system properties file (called "felix.properties" for most Aspire installations). That may be the better location for these properties.

Inside <configAdmin>, each <property> tag contains a "pid" attribute, which is the "persistent ID" of the configuration element. The nested properties are the OSGi Configuration properties.

See the following for more information about OSGi and Apache Felix configuration properties:

- Apache Felix Framework Usage
- OSGi specifications download page - Go here and download the "Compendium Specification" to get more details on the OSGi configuration server and how it's used.
- Apache Felix Web Console Properties
- Apache Felix HTTP Service Properties

# Security Configuration

This is the configuration to use the Login page.

```
"authentication": {
  "tokenExpiration": "30m",
  "refreshExpiration": "1h",
  "validateRemoteHost" : "true",
  "type": "Ldap",
  "ldap": {
    "server": "ldap://oldap:389",
    "authentication": "simple",
    "bindDN": "cn=admin,dc=accenture,dc=com",
    "bindDNPassword": "password",
    "searchBase": "dc=accenture,dc=com",
    "userDNQuery": "(uid={user})",
    "groupsHoldMembers": "true",
    "memberAttr": "uniqueMember",
    "connectTimeout": "3000",
    "readTimeout": "5000",
    "roles": [
      {
        "dn": "cn=administrators,ou=Groups,dc=accenture,dc=com",
        "group": "true",
        "roles": [
          "ADMINISTRATOR"
        ]
      },
      {
        "dn": "cn=operators,ou=Groups,dc=accenture,dc=com",
        "group": "true",
        "roles": [
          "OPERATOR"
        ]
      }
    ]
  }
}
```

ⓘ **Important Information**

In order to avoid issues of invalid token between nodes when security is enabled, the environment variable named **ASPIRE_JWT_SECRET** needs to be set with a random string, which is 32 characters (combination of numbers, special characters, letters lower and upper case) in length in every aspire node (all with the same value).

Aspire uses a token-based security to access the Rest API. These are the general configuration options.

| Property | Type | Default | Description |
|---|---|---|---|
| tokenExpiration | String | 30m | (Optional) The access token expiration time |
| refreshExpiration | String | 4h | (Optional) The refresh token expiration time. This value is recommended to be greater than the tokenExpiration |
| validateRemoteHost | boolean | true | (Optional) Whether or not the session tokens can only be used for the same node that generated it, and if the client IP should be recorded on the session for validation |
| type | String | | **Required:** Currently only "Ldap" or "OIDC" authentication is supported. |

## LDAP Configuration

LDAP is configured as defined below.

⚠ When configuring Aspire behind a LoadBalancer make sure the health checks point to **GET /aspire/_api/login** endpoint which will always return a valid JSON if Aspire loaded correctly. If you use **GET /aspire** for health checks it will return 401 as it will be restricted as part of the authentication settings.

```
"type": "Ldap",
"ldap": {
  "server": "ldap://oldap:389",
  "authentication": "simple",
  "bindDN": "cn=admin,dc=accenture,dc=com",
  "bindDNPassword": "password",
  "searchBase": "dc=accenture,dc=com",
  "userDNQuery": "(uid={user})",
  "groupsHoldMembers": "true",
  "memberAttr": "uniqueMember",
  "connectTimeout": "3000",
  "readTimeout": "5000",
  "roles": [
    {
      "dn": "cn=administrators,ou=Groups,dc=accenture,dc=com",
      "group": "true",
      "roles": [
        "ADMINISTRATOR"
      ]
    },
    {
      "dn": "cn=operators,ou=Groups,dc=accenture,dc=com",
      "group": "true",
      "roles": [
        "OPERATOR"
      ]
    }
  ]
}
```

These are the configuration properties for the LDAP authentication:

| Property | Type | Default | Description |
|---|---|---|---|
| server | String | | **Required:** The LDAP server to use |
| authentication | String | anonymous | (Optional) The authentication types, "simple" and "anonymous" are supported |
| bindDN | String | | **Required:** User DN to authenticate with. Not required if the authentication is anonymous. |
| bindDNPassword | String | | **Required:** The password of the User DN. This value is recommended to be passed using the property aspire.ldap.bind.dn. password as a JVM parameter or as an environment variable instead of using the settings file. Not required if the authentication is anonymous. |
| searchBase | String | | **Required:** The base used to search the users to log in |
| userDNQuery | String | | **Required:** The query used to search for the user. |
| groupsHoldMembers | String | false | (Optional) If true, the groups in LDAP contain the members |
| memberAttr | String | memberOf | (Optional) If groupsHoldMembers is true, this is the group attribute that contains the members. If groupsHoldMembers is false, this is the user attribute that contains the groups. |
| connectTimeout | String | 1m | (Optional) LDAP server timeout in ms or using the ms, s, m, h units notation |
| readTimeout | String | 5m | (Optional) LDAP read timeout in ms or using the ms, s, m, h units notation |
| roles | Array | | **Required:** List of groups or users associated with roles. |
| roles/dn | String | | **Required:** The user or group DN |
| roles/group | String | false | (Optional) Flag to determine if it is a user or a group |
| roles/roles | Array | | **Required:** The roles for this user or group. The supported roles ADMINISTRATOR or OPERATOR. |

> ⓘ **Important Information**
>
> In order to avoid issues of invalid token between nodes when security is enabled, the environment variable named **ASPIRE_JWT_SECRET** needs to be set with a random string, which is 32 characters (combination of numbers, special characters, letters lower and upper case) in length in every aspire node (all with the same value).

# OIDC Configuration (SSO)

OIDC (SSO) is configured as bellow

> ⊘ When configuring Aspire behind a LoadBalancer make sure the health checks point to **GET /aspire/_api/login** endpoint which will always return a valid JSON if Aspire loaded correctly. If you use **GET /aspire** for health checks it will return 401 as it will be restricted as part of the authentication settings.

```
"type": "OIDC",
"tokenExpiration": "150000",
"clientId": "7a908761-123f-65g7c3fca7b3",
"discoveryURI": "https://login.microsoftonline.com/{tenant}/v2.0/.well-known/openid-configuration",
"logoutURI": "https://login.microsoftonline.com/common/oauth2/v2.0/logout",
"rolesClaim": "roles",
"scope": "openid email profile",
"userNameClaim": "name",
"preferred_jws_algorithm" : "RS256",
"roleMapping": [
  {
    "original": "Administrator",
    "role": "ADMINISTRATOR"
  },
  {
    "original": "Operator",
    "role": "OPERATOR"
  }
]
```

These are configuration properties for the OIDC authentication:

| Property | Type | Default | Description |
|---|---|---|---|
| clientId | String | | **Required:** Your app registration's Application (client) ID |
| discoveryURI | String | | **Required:** OpenID Configuration URI. If using Azure, it should look like: https://login.microsoftonline.com/{tenant}/v2.0/.well-known/openid-configuration |
| logoutURI | String | | Optional: URL to call once the logout is complete |
| redirectURI | String | | Optional: If the Aspire Manager node is accessed via a proxy or load balancer, add the proxy or load balancer URL here, and append the /aspire/_api/login/sso path to it. |
| rolesClaim | String | | **Required:** Unique roles for both internal and external users. |
| scope | String | | **Required:** a space-separated list of identifiers used to specify what access privileges are being requested, "openid" is a required scope |
| userNameClaim | String | | **Required:** Unique username for both internal and external users. |
| preferred_jws_algorithm | String | | **Optional:**  OIDC JWS algorithm use by default for the provider, one of the most common is RS256. |
| roleMapping | Array | | **Required:** List of groups or users associated with roles. |
| roleMapping/original | String | | **Required:** The original user or group to map to the role as it comes in the ID token |
| roleMapping/role | String | | **Required:** The roles for this user or group. The supported roles ADMINISTRATOR or OPERATOR. |

> ⓘ **Important Information**
>
> In order to avoid issues of invalid token between nodes when security is enabled, the environment variable named **ASPIRE_JWT_SECRET** needs to be set with a random string, which is 32 characters (combination of numbers, special characters, letters lower and upper case) in length in every aspire node (all with the same value).

---

# Simple (username & password)

Simple authentication with a given username and password via environmental variables can be configured with the following configuration:

⊘

⚠ When configuring Aspire behind a LoadBalancer make sure the health checks point to **GET /aspire/_api/login** endpoint which will always return a valid JSON if Aspire loaded correctly. If you use **GET /aspire** for health checks it will return 401 as it will be restricted as part of the authentication settings.

```
"type": "simple"
```

The **username** can be provided with the environmental variable called: **ASPIRE_USER**

The **password** must be provided with the environmental variable called: **ASPIRE_PASSWORD**

ⓘ **Important Information**

In order to avoid issues of invalid token between nodes when security is enabled, the environment variable named **ASPIRE_JWT_SECRET** needs to be set with a random string, which is 32 characters (combination of numbers, special characters, letters lower and upper case) in length in every aspire node (all with the same value).

# Encryption provider

Aspire encryption tasks are managed with a plug-able provider. Clients can now have their own encryption methods, if they wish to do so, by providing an implementation and configuring the settings file accordingly.

## Default Encryption Provider

A default encryption provider is configured with the Aspire installation. Additional configuration is only required if a different provider is to be used.

```
"encryptionProvider": {
    "implementation": "com.accenture.aspire:aspire-encryption-provider",
    "masterKeyFilePath": "config/encryptionKey"
}
```

The default povider uses AES-256 with a key of 32bytes for encryption. Theses are the properties used to configure the default encryption provider.

| Parameter | Type | Default | Description |
|---|---|---|---|
| implementation | String | com.accenture. aspire:aspire-encryption-provider | **Required:** The maven coordinates to the encryption provider implementation |
| masterKeyFilePath | String | | (Optional) Path (including file name) where the encryption key is located, if not provided, a default in-memory key will be used, for **production** installations it must be always provided. This can also be passed as a JVM parameter or as the environment variable **aspire_encryption_key_file** (see Encryption properties) |
| | | | This should be a 32 byte file, if longer, the first 32 bytes will be used as the encryption key. |
| | | | Grant read access to the Aspire user only (chmod 400 <file>) |
| | | | This file could be generated randomly |
| | | | `$ head -c 32 /dev/urandom > encryption.key` |

## AWS KMS Encryption Provider

If the AMS KMS Encryption Provider should be used, change the settings file to:

```
"encryptionProvider": {
    "implementation": "com.accenture.aspire:aspire-aws-kms-encryption-provider",
    "roleARN": "arn:aws:iam:[account_id]:role/[role_id]",
    "keyARN" : "arn:aws:kms:[region]:key/[key_id]",
    "region" : "us-east-1",
    "accessKey" : "[ACCESS_KEY]",
    "secretKey" : "[SECRET_KEY]"
}
```

The AWS KMS Encryption Provider uses KMS to hold the encryption keys and to encrypt/decrypt secrets. More information at AWS KMS Encryption.

| Parameter | Required | Default | Description |
|-----------|----------|---------|-------------|
| roleARN | no | null | (Optional) If the KMS service must be accessed through the assumption of an IAM role, specify the role ARN. |
| keyARN | yes | N/A | The KMS key ARN. See Aspire KMS encryption for more information about creating a KMS key for Aspire. |
| region | yes | N/A | The AWS region on which the KMS service will be used |
| accessKey | no | null | (Optional) Specify the access key if static credentials must be used. If this is not specified, the Default Credential Provider Chain will be used. |
| secretKey | no | null | (Optional) Specify the secret key if static credentials must be used. If this is not specified, the Default Credential Provider Chain will be used. |

# Nodes Properties

The nodes properties are the configuration parameter to use for the worker and manager nodes.

```
    "nodesProperties": {
        "worker": {
          "maxMemQueueSize": "1000",
          "queueSizeThreshold": "0.75",
          "cleanUpWaitTime": "300000",
          "cleanUpThreshold": "3600000",
          "maxEnqueueRetries": "5",
          "debug": "false",
          "appCleanUpWaitTime": "60000",
          "appCleanUpThreshold": "3600000",
          "tags" : "",
          "entryProcessorBaseSleep" : "200",
          "entryProcessorMaxSleep" : "10000",
          "entryProcessorMaxIterations" : "5",
          "entryProcessorMultiplier" : "2",
          "batchLoaderBaseSleep" : "200",
          "batchLoaderMaxSleep" : "10000",
          "batchLoaderMaxIterations" : "5",
          "batchLoaderMultiplier" : "2",
          "connectionTimeout" : "60000",
          "socketTimeout" : "60000",
          "maxRetries" : "3",
          "proxyHost" : "",
          "proxyPort" : "0",
          "proxyUser" : "",
          "proxyPassword" : "",
          "pingFrequency" : "15000",
          "nodeFailureTimeout" : "30000"
        },
        "manager": {
          "scanBatchCreatorBaseSleep" : "200",
          "scanBatchCreatorMaxSleep" : "10000",
          "scanBatchCreatorMaxIterations" : "10",
          "scanBatchCreatorMultiplier" : "2",
          "processBatchCreatorBaseSleep" : "200",
          "processBatchCreatorMaxSleep" : "10000",
          "processBatchCreatorMaxIterations" : "10",
          "processBatchCreatorMultiplier" : "2",
          "crawlProgressManagerBaseSleep" : "500",
          "schedulerBaseSleep" : "10000",
          "maxBatches" : "1000",
          "maxBatchItems" : "100",
          "connectionTimeout" : "60000",
          "socketTimeout" : "60000",
          "maxRetries" : "3",
          "proxyHost" : "",
          "proxyPort" : "0",
          "proxyUser" : "",
          "proxyPassword" : "",
          "pingFrequency" : "15000",
          "nodeFailureTimeout" : "30000",
          "tags": "",
          "workerRoundRobin": "false",
          "workerRoundRobinTimeout": "600000"
        }
    }
```

## Worker properties

These properties will be used by all worker nodes in the cluster.

| Parameter | Type | Default | Description |
|-----------|------|---------|-------------|
| maxMemQueueSize | String | 1000 | **(Required)** The maximum number of items to keep in the in memory queue |
| queueSizeThreshold | String | 0.75 | **(Required)** The capacity threshold of the in memory queue before requesting more items to the managers |
| cleanUpWaitTime | String | 300000 | **(Required)** The wait time in ms for the thread that checks the connectors clean up threshold |

| cleanUpThreshold | String | 3600000 | **(Required)** The time in ms for a connector to be idle before being removed from memory |
|---|---|---|---|
| maxEnqueueRetries | String | 5 | **(Required)** The number of retries to enqueue a item into the framework pipeline |
| debug | String | false | **(Required)** Enables the debug mode for the node |
| appCleanUpWaitTime | String | 60000 | **(Required)** The wait time in ms for the thread that checks the workflow application's clean-up threshold |
| appCleanUpThreshold | String | 3600000 | **(Required)** The time in ms for a workflow application to be idle before being removed from memory |
| tags | String | | (Optional) The tags of the worker node. These tags will determine which items this node can process. |
| entryProcessorBaseSleep | String | 200 | **(Required)** The base sleep time in ms for the thread in charge of queuing received items into the connector framework pipelines |
| entryProcessorMaxSleep | String | 10000 | **(Required)** The maximum sleep in ms for the thread in charge of queuing received items into the connector framework pipelines |
| entryProcessorMaxIterations | String | 5 | **(Required)** The number of iterations without queuing items before in increasing the sleep time |
| entryProcessorMultiplier | String | 2 | **(Required)** The multiplier used to increase the sleep time after the specified iterations without queuing items |
| batchLoaderBaseSleep | String | 200 | **(Required)** The base sleep time in ms for the thread in charge of requesting batches to the manager nodes |
| batchLoaderMaxSleep | String | 10000 | **(Required)** The maximum sleep in ms for the thread in charge of requesting batches to the manager nodes |
| batchLoaderMaxIterations | String | 5 | **(Required)** The number of iterations without receiving batches from the managers nodes before in increasing the sleep time |
| batchLoaderMultiplier | String | 2 | **(Required)** The multiplier used to increase the sleep time after the specified iterations without receiving batches from the managers nodes |
| connectionTimeout | String | 20000 | **(Required)** The connection timeout for requests to other aspire nodes |
| socketTimeout | String | 20000 | **(Required)** The socket timeout for requests to other aspire nodes |
| maxRetries | String | 3 | **(Required)** The number of retries for requests to other aspire nodes |
| proxyHost | String | | (Optional) The proxy host to use for requests to other aspire nodes |
| proxyPort | String | | (Optional) The proxy port to use for requests to other aspire nodes. Must be provided if the proxyHost is configured. |
| proxyUser | String | | (Optional) The proxy user to use for requests to other aspire nodes |
| proxyPassword | String | | (Optional) The proxy password to use for requests to other aspire nodes. |
| pingFrequency | String | 15000 | **(Required)** The ping timeout used to determine if a node is not working. The node will be marked as failed in this case and the node eventually will shut down itself. |
| nodeFailureTimeout | String | 30000 | **(Required)** The frequency for the node to ping to Elasticsearch. The pings are used to determine if a node is alive and working properly. |

## Manager properties

These properties will be used by all manager nodes in the cluster.

| Parameter | Type | Default | Description |
|---|---|---|---|
| scanBatchCreatorBaseSleep | String | 200 | **(Required)** The base sleep time in ms for the thread in charge of creating batches from the scan queue |
| scanBatchCreatorMaxSleep | String | 10000 | **(Required)** The maximum sleep in ms for the thread in charge of creating batches from the scan queue |
| scanBatchCreatorMaxIterations | String | 10 | **(Required)** The number of iterations without creating new scan batches before in increasing the sleep time |
| scanBatchCreatorMultiplier | String | 2 | **(Required)** The multiplier used to increase the sleep time after the specified iterations without creating new scan batches |
| processBatchCreatorBaseSleep | String | 200 | **(Required)** The base sleep time in ms for the thread in charge of creating batches from the process queue |
| processBatchCreatorMaxSleep | String | 10000 | **(Required)** The maximum sleep in ms for the thread in charge of creating batches from the process queue |
| processBatchCreatorMaxIterations | String | 10 | **(Required)** The number of iterations without creating new process batches before in increasing the sleep time |
| processBatchCreatorMultiplier | String | 2 | **(Required)** The multiplier used to increase the sleep time after the specified iterations without creating new process batches |

| crawlProgressManagerB aseSleep | String | 200 | **(Required)** The base sleep time in ms for the thread in charge of monitoring active crawls |
|---|---|---|---|
| schedulerBaseSleep | String | 10000 | **(Required)** The base sleep time in ms for the thread in charge of executing seeds based on the configured schedules |
| maxBatches | String | 1000 | **(Required)** The maximum number of batches the manager will keep in memory |
| maxBatchItems | String | 100 | **(Required)** The maximum number of documents per batch |
| debug | String | false | **(Required)** Enables the debug mode for the node |
| connectionTimeout | String | 20000 | **(Required)** The connection timeout for requests to other aspire nodes |
| socketTimeout | String | 20000 | **(Required)** The socket timeout for requests to other aspire nodes |
| maxRetries | String | 3 | **(Required)** The number of retries for requests to other aspire nodes |
| proxyHost | String | | (Optional) The proxy host to use for requests to other aspire nodes |
| proxyPort | String | | (Optional) The proxy port to use for requests to other aspire nodes. Must be provided if the proxyHost is configured. |
| proxyUser | String | | (Optional) The proxy user to use for requests to other aspire nodes |
| proxyPassword | String | | (Optional) The proxy password to use for requests to other aspire nodes. |
| pingFrequency | String | 15000 | **(Required)** The ping timeout used to determine if a node is not working. The node will be marked as failed in this case and the node eventually will shut down itself. |
| nodeFailureTimeout | String | 30000 | **(Required)** The frequency for the node to ping to Elasticsearch. The pings are used to determine if a node is alive and working properly. |
| tags | String | | (Optional) The tags of the manager node. These tags will determine which seeds this node can process. It should be a comma separated list of tags. |
| workerRoundRobin | String | false | (Optional) If round-robin should be applied when serving workers with batches |
| workerRoundRobinTime out | String | 600000 | (Optional) The time in ms after which the worker is considered timed out when round-robin is used. |

## Specifying properties for a specific node

Is possible to configure some or all properties of a specific Aspire worker or manager node with specific values the following way

```
"worker": {
  "maxMemQueueSize": "1000",
  "queueSizeThreshold": "0.75",
  "cleanUpWaitTime": "300000",
  "cleanUpThreshold": "3600000",
  "maxEnqueueRetries": "5",
  "node_hostname": {
    "cleanUpWaitTime": "150000",
    "cleanUpThreshold": "1800000"
  }
},
"manager": {
  "maxBatches" : "1000",
  "maxBatchItems" : "100",
  "node_hostname": {
    "maxBatches": "2000",
    "maxBatchItems": "150"
  }
}
```

This way, the specified node will use the specific values over the general ones.

ⓘ   All properties, except for the debug flag, can be passed as JVM parameters or environment variables.