::Go back to Oozie Documentation Index::

Oozie Installation and Configuration

Basic Setup

Follow the instructions at Oozie Quick Start .

Environment Setup

IMPORTANT: Oozie ignores any set value for `OOZIE_HOME` , Oozie computes its home automatically.

When running Oozie with its embedded Tomcat server, the `conf/oozie-env.sh` file can be used to configure the following environment variables used by Oozie:

CATALINA_OPTS : settings for the Embedded Tomcat that runs Oozie Java System properties for Oozie should be specified in this variable. No default value.

OOZIE_CONFIG_FILE : Oozie configuration file to load from Oozie configuration directory. Default value `oozie-site.xml` .

OOZIE_LOGS : Oozie logs directory. Default value `logs/` directory in the Oozie installation directory.

OOZIE_LOG4J_FILE : Oozie Log4J configuration file to load from Oozie configuration directory. Default value `oozie-log4j.properties` .

OOZIE_LOG4J_RELOAD : Reload interval of the Log4J configuration file, in seconds. Default value `10`

OOZIE_HTTP_PORT : The port Oozie server runs. Default value `11000` .

OOZIE_ADMIN_PORT : The admin port Oozie server runs. Default value `11001` .

OOZIE_HTTP_HOSTNAME : The host name Oozie server runs on. Default value is the output of the command `hostname -f` .

OOZIE_BASE_URL : The base URL for actions callback URLs to Oozie. The default value is `http://${OOZIE_HTTP_HOSTNAME}:${OOZIE_HTTP_PORT}/oozie` .

OOZIE_CHECK_OWNER : If set to `true` , Oozie setup/start/run/stop scripts will check that the owner of the Oozie installation directory matches the user invoking the script. The default value is undefined and interpreted as a `false` .

Oozie Server Setup

The oozie-setup.sh script prepares the embedded Tomcat server to run Oozie.

The oozie-setup.sh script options are:

```
Usage  : oozie-setup.sh <OPTIONS>"
         [-extjs EXTJS_PATH] (expanded or ZIP, to enable the Oozie webconsole)"
         [-hadoop HADOOP_VERSION HADOOP_PATH] (Hadoop version [0.20.1|0.20.2|0.20.104|0.20.200]"
                                               and Hadoop install dir)"
         [-jars JARS_PATH] (multiple JAR path separated by ':')"
         (without options does default setup, without the Oozie webconsole)"
```

If a directory libext/ is present in Oozie installation directory, the oozie-setup.sh script include all JARs in the libext/ directory in Oozie WAR file.

If the ExtJS ZIP file is present in the libext/ directory, it will be added to Oozie WAR as well. The ExtJS library file name be ext-2.2.zip .

Setting Up Oozie with an Alternate Tomcat

Use the addtowar.sh script to prepare the Oozie server only if Oozie will run with a different servlet container than the embedded Tomcat provided with the distribution.

The addtowar.sh script adds Hadoop JARs, JDBC JARs and the ExtJS library to the Oozie WAR file.

The addtowar.sh script options are:

```
 Usage  : addtowar <OPTIONS>
 Options: -inputwar INPUT_OOZIE_WAR
          -outputwar OUTPUT_OOZIE_WAR
          [-hadoop HADOOP_VERSION HADOOP_PATH]
          [-extjs EXTJS_PATH]
          [-jars JARS_PATH] (multiple JAR path separated by ':')
```

The original oozie.war file is in the Oozie server installation directory.

After the Hadoop JARs and the ExtJS library has been added to the oozie.war file Oozie is ready to run.

Delete any previous deployment of the oozie.war from the servlet container (if using Tomcat, delete =oozie.war= and oozie directory from Tomcat's webapps/ directory)

Deploy the prepared oozie.war file (the one that contains the Hadoop JARs adn the ExtJS library) in the servlet container (if using Tomcat, copy the prepared oozie.war file to Tomcat's webapps/ directory).

IMPORTANT: Only one Oozie instance can be deployed per Tomcat instance.

Database Configuration

Oozie works with HSQL, Derby, MySQL, Oracle or PostgreSQL databases.

By default, Oozie is configured to use Embedded Derby.

Oozie bundles the JDBC drivers for HSQL, Embedded Derby and PostgreSQL.

HSQL is normally used for testcases as it is an in-memory database and all data is lost everytime Oozie is stopped.

If using Derby, MySQL, Oracle or PostgreSQL, the Oozie database schema must be created using the ooziedb.sh command line tool.

If using MySQL or Oracle, the corresponding JDBC driver JAR file mut be copied to Oozie's libext/ directory and

it must be added to Oozie WAR file using the `bin/addtowar.sh` or the `oozie-setup.sh` scripts using the `-jars` option.

The SQL database used by Oozie is configured using the following configuration properties (default values shown):

```
oozie.db.schema.name=oozie
oozie.service.JPAService.create.db.schema=false
oozie.service.JPAService.validate.db.connection=false
oozie.service.JPAService.jdbc.driver=org.apache.derby.jdbc.EmbeddedDriver
oozie.service.JPAService.jdbc.url=jdbc:derby:${oozie.data.dir}/${oozie.db.schema.name}-db;create=true
oozie.service.JPAService.jdbc.username=sa
oozie.service.JPAService.jdbc.password=
oozie.service.JPAService.pool.max.active.conn=10
```

NOTE: If the `oozie.db.schema.create` property is set to `true` (default value is `false`) the Oozie tables will be created automatically without having to use the `ooziedb` command line tool. Setting this property to `true` it is recommended only for development.

NOTE: If the `oozie.db.schema.create` property is set to true, the `oozie.service.JPAService.validate.db.connection` property value is ignored and Oozie handles it as set to `false` .

Once `oozie-site.xml` has been configured with the database configuration execute the `ooziedb.sh` command line tool to create the database:

```
$ bin/ooziedb.sh create -sqlfile oozie.sql -runValidate DB Connection.
DONE
Check DB schema does not exist
DONE
Check OOZIE_SYS table does not exist
DONE
Create SQL schema
DONE
DONE
Create OOZIE_SYS table
DONE
Oozie DB has been created for Oozie version '3.2.0'
The SQL commands have been written to: oozie.sql
$
```

NOTE: If using MySQL or Oracle, copy the corresponding JDBC driver JAR file to the `libext/` directory before running the `ooziedb.sh` command line tool.

NOTE: If instead using the '-run' option, the '-sqlfile ' option is used, then all the database changes will be written to the specified file and the database won't be modified.

If using HSQL there is no need to use the `ooziedb` command line tool as HSQL is an im-memory database. Use the following configuration properties in the oozie-site.xml:

```
oozie.db.schema.name=oozie
oozie.service.JPAService.create.db.schema=true
oozie.service.JPAService.validate.db.connection=false
oozie.service.JPAService.jdbc.driver=org.hsqldb.jdbcDriver
oozie.service.JPAService.jdbc.url=jdbc:hsqldb:mem:${oozie.db.schema.name}
oozie.service.JPAService.jdbc.username=sa
oozie.service.JPAService.jdbc.password=
oozie.service.JPAService.pool.max.active.conn=10
```

Oozie Configuration

By default, Oozie configuration is read from Oozie's `conf/` directory

The Oozie configuration is distributed in 3 different files:

- `oozie-site.xml` : Oozie server configuration

- oozie-log4j.properties : Oozie logging configuration
- adminusers.txt : Oozie admin users list

Oozie Configuration Properties

All Oozie configuration properties and their default values are defined in the oozie-default.xml file.

Oozie resolves configuration property values in the following order:

- If a Java System property is defined, it uses its value
- Else, if the Oozie configuration file (=oozie-site.xml=) contains the property, it uses its value
- Else, it uses the default value documented in the oozie-default.xml file

NOTE: The oozie-default.xml file found in Oozie's conf/ directory is not used by Oozie, it is there for reference purposes only.

Logging Configuration

By default, Oozie log configuration is defined in the oozie-log4j.properties configuration file.

If the Oozie log configuration file changes, Oozie reloads the new settings automatically.

By default, Oozie logs to Oozie's logs/ directory.

Oozie logs in 4 different files:

- oozie.log: web services log streaming works from this log
- oozie-ops.log: messages for Admin/Operations to monitor
- oozie-instrumentation.log: intrumentation data, every 60 seconds (configurable)
- oozie-audit.log: audit messages, workflow jobs changes

The embedded Tomcat and embedded Derby log files are also written to Oozie's logs/ directory.

Oozie User Authentication Configuration

Oozie supports Kerberos HTTP SPNEGO authentication, pseudo/simple authentication and anonymous access for client connections.

Anonymous access (*default*) does not require the user to authenticate and the user ID is obtained from the job properties on job submission operations, other operations are anonymous.

Pseudo/simple authentication requires the user to specify the user name on the request, this is done by the PseudoAuthenticator class by injecting the user.name parameter in the query string of all requests. The user.name parameter value is taken from the client process Java System property user.name .

Kerberos HTTP SPNEGO authentication requires the user to perform a Kerberos HTTP SPNEGO authentication sequence.

If Pseudo/simple or Kerberos HTTP SPNEGO authentication mechanisms are used, Oozie will return the user an authentication token HTTP Cookie that can be used in later requests as identy proof.

Oozie uses Apache Hadoop-Auth (Java HTTP SPENGO) library for authentication. This library can be extended to support other authentication mechanisms.

Oozie user authentication is configured using the following configuration properties (default values shown):

```
oozie.authentication.type=simple
oozie.authentication.token.validity=36000
oozie.authentication.signature.secret=
oozie.authentication.cookie.domain=
oozie.authentication.simple.anonymous.allowed=true
oozie.authentication.kerberos.principal=HTTP/localhost@${local.realm}
oozie.authentication.kerberos.keytab=${oozie.service.HadoopAccessorService.keytab.file}
```

The `type` defines authentication used for Oozie HTTP endpoint, the supported values are: simple | kerberos | #AUTHENTICATION_HANDLER_CLASSNAME#.

The `token.validity` indicates how long (in seconds) an authentication token is valid before it has to be renewed.

The `signature.secret` is the signature secret for signing the authentication tokens. If not set a random secret is generated at startup time.

The `oozie.authentication.cookie.domain` The domain to use for the HTTP cookie that stores the authentication token. In order to authentiation to work correctly across all Hadoop nodes web-consoles the domain must be correctly set.

The `simple.anonymous.allowed` indicates if anonymous requests are allowed. This setting is meaningful only when using 'simple' authentication.

The `kerberos.principal` indicates the Kerberos principal to be used for HTTP endpoint. The principal MUST start with 'HTTP/' as per Kerberos HTTP SPNEGO specification.

The `kerberos.keytab` indicates the location of the keytab file with the credentials for the principal. It should be the same keytab file Oozie uses for its Kerberos credentials for Hadoop.

Oozie Hadoop Authentication Configuration

Oozie works with Hadoop versions which support Kerberos authentication.

Oozie Hadoop authentication is configured using the following configuration properties (default values shown):

```
oozie.service.HadoopAccessorService.kerberos.enabled=false
local.realm=LOCALHOST
oozie.service.HadoopAccessorService.keytab.file=${user.home}/oozie.keytab
oozie.service.HadoopAccessorService.kerberos.principal=${user.name}/localhost@{local.realm}
```

The above default values are for a Hadoop 0.20 secure distribution (with support for Kerberos authentication).

To enable Kerberos authentication, the following property must be set:

```
oozie.service.HadoopAccessorService.kerberos.enabled=true
```

When using Kerberos authentication, the following properties must be set to the correct values (default values shown):

```
local.realm=LOCALHOST
oozie.service.HadoopAccessorService.keytab.file=${user.home}/oozie.keytab
oozie.service.HadoopAccessorService.kerberos.principal=${user.name}/localhost@{local.realm}
```

IMPORTANT: When using Oozie with a Hadoop 20 with Security distribution, the Oozie user in Hadoop must be configured as a proxy user.

User ProxyUser Configuration

Oozie supports impersonation or proxyuser functionality (identical to Hadoop proxyuser capabilities and conceptually similar to Unix 'sudo').

Proxyuser enables other systems that are Oozie clients to submit jobs on behalf of other users.

Because proxyuser is a powerful capability, Oozie provides the following restriction capabilities (similar to Hadoop):

- Proxyuser is an explicit configuration on per proxyuser user basis.
- A proxyuser user can be restricted to impersonate other users from a set of hosts.
- A proxyser user can be restricted to impersonate users belonging to a set of groups.

There are 2 configuration properties needed to set up a proxyuser:

- oozie.service.ProxyUserService.proxyuser.#USER#.hosts: hosts from where the user #USER# can impersonate other users.
- oozie.service.ProxyUserService.proxyuser.#USER#.groups: groups the users being impersonated by user #USER# must belong to.

Both properties support the '*' wildcard as value. Although this is recommended only for testing/development.

User Authorization Configuration

Oozie has a basic authorization model:

- Users have read access to all jobs
- Users have write access to their own jobs
- Users have write access to jobs based on an Access Control List (list of users and groups)
- Users have read access to admin operations
- Admin users have write access to all jobs
- Admin users have write access to admin operations

If security is disabled all users are admin users.

Oozie security is set via the following configuration property (default value shown):

```
oozie.service.AuthorizationService.security.enabled=false
```

NOTE: the old ACL model where a group was provided is still supported if the following property is set in `oozie-site.xml`:

```
oozie.service.AuthorizationService.default.group.as.acl=true
```

Admin users are determined from the list of admin groups, specified in `oozie.service.AuthorizationService.admin.groups` property. Use commas to separate multiple groups, spaces, tabs and ENTER characters are trimmed.

If the above property for admin groups is not set, then the admin users are the users specified in the `conf/adminusers.txt` file. The syntax of this file is:

- One user name per line
- Empty lines and lines starting with '#' are ignored

Oozie System ID Configuration

Oozie has a system ID that is is used to generate the Oozie temporary runtime directory, the workflow job IDs, and the workflow action IDs.

Two Oozie systems running with the same ID will not have any conflict but in case of troubleshooting it will be easier to identify resources created/used by the different Oozie systems if they have different system IDs (default value shown):

```
oozie.system.id=oozie-${user.name}
```

Fine Tuning an Oozie Server

Refer to the oozie-default.xml for details.

Starting and Stopping Oozie

Use the standard Tomcat commands to start and stop Oozie.

Oozie Command Line Installation

Copy and expand the `oozie-client` TAR.GZ file bundled with the distribution. Add the `bin/` directory to the `PATH` .

Refer to the Command Line Interface Utilities document for a a full reference of the `oozie` command line tool.

Oozie Share Lib

The Oozie share lib TAR.GZ file bundled with the distribution contains the necessary files to run Oozie map-reduce streaming and pig actions.

The bundled Streaming and Pig JARs are the ones used by Oozie testcases.

Oozie Coordinators/Bundles Processing Timezone

By default Oozie runs coordinator and bundle jobs using `UTC` timezone for datetime values specified in the application XML and in the job parameter properties. This includes coordinator applications start and end times of jobs, coordinator datasets initial-instance, bundle applications kick-offtimes. In addition, coordinator dataset instance URI templates will be resolved using datetime values of the Oozie processing timezone.

It is possible to set the Oozie processing timezone to a timezone that is an offset of UTC, alternate timezones must expressed in using a GMT offset (`GMT+/-####`). For example: `GMT+0530` (India timezone).

To change the default `UTC` timezone, use the `oozie.processing.timezone` property in the `oozie-site.xml` . For example:

```
<configuration>
    <property>
        <name>oozie.processing.timezone</name>
        <value>GMT+0530</value>
    </property>
</configuration>
```

IMPORTANT: If using a processing timezone other than `UTC` , all datetime values in coordinator and bundle jobs must be expressed in the corresponding timezone, for example `2012-08-08T12:42+0530` .

NOTE: It is strongly encouraged to use `UTC` , the default Oozie processing timezone.

For more details on using an alternate Oozie processing timezone, please reffer to the Coordinator Fuctional Specification, section '4. Datetime'

Advanced/Custom Environment Settings

Oozie can be configured to use Unix standard filesystem hierarchy for its different files (configuration, logs, data and temporary files).

These settings must be done in the `bin/oozie-env.sh` script.

This script is sourced before the configuration `oozie-env.sh` and supports additional environment variables (shown with their default values):

```
export OOZIE_CONF=${OOZIE_HOME}/conf
export OOZIE_DATA={OOZIE_HOME}/data
export OOZIE_LOG={OOZIE_HOME}/logs
export CATALINA_BASE=${OOZIE_HOME}/oozie-server
export CATALINA_TMPDIR=${OOZIE_HOME}/oozie-server/temp
export CATALINA_OUT=${OOZIE_LOGS}/catalina.out
export CATALINA_PID=/tmp/oozie.pid
```

Sample values to make Oozie follow Unix standard filesystem hierarchy:

```
export OOZIE_CONFIG=/etc/oozie
export OOZIE_DATA=/var/lib/oozie
export OOZIE_LOG=/var/log/oozie
export CATALINA_BASE=${OOZIE_DATA}/oozie-server
export CATALINA_TMPDIR=/tmp
export CATALINA_PID=/tmp/oozie.pid
```

::Go back to Oozie Documentation Index::