JOC Cockpit - Logging

Introduction

The JOC Cockpit Web Services provide extensive logging, including the compliance-conform Audit Log. Logging can be configured for a number of different operating areas independently, allowing logging performance to be optimized in line with administration requirements.The Apache Log4j logging utility is used in the web services. In addition to the logging of the Web Services, the JOC cockpit can also be used to access JobScheduler log files.

Log File Location

All JOC Cockpit log files are stored in a common `logs` folder.

If the Jetty web server provided with the JOC Cockpit installation is installed in its default location then the JOC Cockpit log files will be found in the `jetty_base/logs` directory. The default paths to this directory are:

- `/home/[user]/sos-berlin.com/joc/jetty_base/logs` on Linux systems and
- `C:\ProgramData\sos-berlin.com\joc\jetty_base\logs` on Windows.

For more information see the Jetty Installation & Configuration section of the JOC Cockpit - Installation article.

Logging Configuration

Logging for the JOC Cockpit is configured in a `log4j.properties` file, which can be found in the `jetty_base/resources/joc` directory.

Note the following:

- There is also a `log4j.properties` file in the `jetty/resources` directory. This is a Jetty configuration file and should *not* be used to configure logging for the JOC Cockpit..
- The `log4j.properties` file is overwritten during installation so system administrators wishing to change, for example, the location or name of the Audit Log file, will need to take suitable precautions whilst updating the JOC Cockpit.

Assigning changes to the log4j properties file

Running a stand-alone JOC Cockpit

Changes made to this `log4j.properties` file are effective immediately after saving the file. It is not necessary to restart the the JOC Cockpit.

Running the JOC Cockpit in a cluster

When the JOC Cockpit is running in a cluster:

- Changes have to be made to the joc4j properties file on each cluster node. Avoid having different configurations at different nodes.
- *All* cluster members have to be restarted after making changes.

Default Logging Configuration

The following code block shows the default log4j logging configuration for the JOC Cockpit:

The JOC Cockpit default log4j.properties configuration

```
log4j.reset=true

#root logger with console appender. All other loggers inherit from this logger.
log4j.rootLogger=info, stdout
log4j.appender.stdout = org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout = org.apache.log4j.EnhancedPatternLayout
log4j.appender.stdout.layout.ConversionPattern = %d{ISO8601}{Europe/Berlin} %-5p %-16.16t %-44.70c{1.} - %m%n

#General Log Files - et the following loggers to 'debug' to debug the JOC Cockpit
log4j.logger.com.sos = info
log4j.logger.org.hibernate.SQL = info

#Apache and Apache Shiro Logs
log4j.logger.org.apache.http = error
log4j.logger.org.apache.commons = error
log4j.logger.org.apache.shiro = info
log4j.logger.org.apache.shiro.session.mgt = error

#Logger for hibernate
log4j.logger.org.hibernate.hql = error
log4j.logger.org.hibernate.event = error
log4j.logger.org.hibernate.cfg = error
log4j.logger.org.hibernate.type = error
log4j.logger.org.hibernate.id = error
log4j.logger.org.hibernate.orm.deprecation = error
log4j.logger.org.hibernate.engine.transaction.internal.TransactionImpl = info

#Logger for audit log
log4j.logger.JOCAuditLog=info, audit
log4j.additivity.JOCAuditLog = false

log4j.appender.audit = org.apache.log4j.FileAppender
log4j.appender.audit.layout = org.apache.log4j.EnhancedPatternLayout
log4j.appender.audit.layout.ConversionPattern = %d{ISO8601}{Europe/Berlin} %-5p %m%n
#filename of audit log
log4j.appender.audit.File = ${jetty.base}/logs/JOCAuditLog.log

#logger for security
#log4j.logger.com.sos.auth=debug, shiro
#log4j.additivity.com.sos.auth= false
#log4j.appender.shiro = org.apache.log4j.FileAppender
#log4j.appender.shiro.layout = org.apache.log4j.EnhancedPatternLayout
#log4j.appender.shiro.layout.ConversionPattern = %d{ISO8601}{Europe/Berlin} %-5p %m%n
##filename of shiro log
#log4j.appender.shiro.File = ${jetty.base}/logs/JOCShiroLog.log

#logger for db
#log4j.logger.com.sos.hibernate=debug, db
#log4j.additivity.com.sos.hibernate= false
#log4j.appender.db = org.apache.log4j.FileAppender
#log4j.appender.db.layout = org.apache.log4j.EnhancedPatternLayout
#log4j.appender.db.layout.ConversionPattern = %d{ISO8601}{Europe/Berlin} %-5p %m%n
##filename of db log
#log4j.appender.db.File = ${jetty.base}/logs/JOCDBLog.log
```

Note:

- A number of logger settings are set by default to the *error* level.This has been done to reduce the amount of information logged.

The Logging Functions

General Log Files

- **`YYYY_MM_DD.stderrout.log`**
    - This log rotates per restart of the JOC Cockpit and per day. It shows startup messages and error messages. This file should not grow in a considerable way as it is used mainly for error messages.
- **`sos_joc-stderr.YYYY-MM-DD.log` and `sos_joc-stdout.YYYY-MM-DD.log`**
    - This log is used on Windows systems and contains messages about the Windows Service start/stop.

Audit Log

- **`JOCAuditLog.log`**
    - This file includes the same information that is visible in the JOC Cockpit from the Audit Log View. All interventions that modify the status of jobs, job chains and orders are written to this file.
    - This file is important for compliance purposes and is not rotated.
    - Growth should not be harmful as one line of log output is created per user action.

Certificate Handling

Log information for certificate handling - including handshakes - can be activated as follows:

On Linux Systems:

- The following information has to be added to the `/etc/default/joc` file.
-

Setting up logging for SSL handshakes

```
-Djavax.net.debug=ssl
```

On Windows systems:

- In Jetty Home (e.g. `C:\Program Files\sos-berlin.com\joc`) in the `service ./jetty/bin` subfolder:
    - start the *sos_jocw.exe* application
    - select the *Java* tab
    - in the *Java Options* field:
        - add `-Djavax.net.debug=ssl`

Certificate Handling log information is written to the **`YYYY_MM_DD.stderrout.log`** file described above.

Installation Log

- **`Install*.log`**
    - The log output that is created by the installer.

Security: Authentication and Session Information

The security logger records authentication. i.e. logging any log in and log out operations, as well as session information.

The following code in the `log4j.properties` file has to be uncommented as shown in the listing below.

Set up logger for authentication and session management

```
#logger for security
log4j.logger.com.sos.auth=debug, shiro
log4j.additivity.com.sos.auth= false
log4j.appender.shiro = org.apache.log4j.FileAppender
log4j.appender.shiro.layout = org.apache.log4j.EnhancedPatternLayout
log4j.appender.shiro.layout.ConversionPattern = %d{ISO8601}{Europe/Berlin} %-5p %m%n
#filename of shiro log
log4j.appender.shiro.File = ${jetty.base}/logs/JOCShiroLog.log
```

The last line of the configuration above causes the security log file to be generated in the `jetty_base/logs/` directory with the name `JOCShiroLog.log`.

The security logger configuration is included in the log4j properties (and commented out) file from release 1.12.4 onwards.

JOC Cockpit cluster for distributed sessions.

The security logger will also log distributed session management information from a JOC Cluster, when the following code is present in the `[main]` section of the shiro ini file.

```
sessionDAO = com.sos.auth.shiro.SOSDistributedSessionDAO
securityManager.sessionManager.sessionDAO = $sessionDAO
```

See the JOC Cockpit - Clustering article for more information about the configuration of clustering and distributed sessions. When this is done the `JOCShiroLog.log` file will also contain the debug output from the session management coming from the `com.sos.auth.shiro.SOSDistributedSessionDAO` class.

JOC cluster fail-over

The hand over is logged in the `com.sos.auth.rest.SOSPermissionsCreator` class. Therefore the `log4j.logger.com.sos.auth` property will also log this debug output.

Database processing: Logging Database debug information

This logger records any database access debug information and requires the following code in the `log4j.properties` file:

Setting up the logger for database operations

```
#logger for db
log4j.logger.com.sos.hibernate=debug, db
log4j.additivity.com.sos.hibernate= false
log4j.appender.db = org.apache.log4j.FileAppender
log4j.appender.db.layout = org.apache.log4j.EnhancedPatternLayout
log4j.appender.db.layout.ConversionPattern = %d{ISO8601}{Europe/Berlin} %-5p %m%n
#filename of db log
log4j.appender.db.File = ${jetty.base}/logs/JOCDBLog.log
```

The last line of the configuration above causes the security log file to be generated in the `jetty_base/logs/` directory with the name `JOCDBLog.log`.

The database processing logger configuration is included in the log4j properties (and commented out) file from release 1.12.4 onwards.

Enabling the JETTY request log

It is possible to enable the request log for the JOC Cockpit Web Service. This means that the contents of any requests for the JOC Cockpit will be logged.

This will be done by calling the script:

- `./install/install_jetty_base requestlog`

which executes:

- `java -jar "%JOC_JETTY_HOME%\start.jar" -Djetty.home="%JOC_JETTY_HOME%" -Djetty.base="%JOC_JETTY_BASE%" --add-to-start=requestlog`

This call will modify the file:

- `$joc_home/jetty_base/start.ini` .

with:

- `--module=requestlog`

The following code block shows the

Content of $joc_home/jetty_base/start.ini

```
 --module=requestlog

## Logging directory (relative to $jetty.base)
# jetty.requestlog.dir=logs

## File path
# jetty.requestlog.filePath=${jetty.requestlog.dir}/yyyy_mm_dd.request.log

## Date format for rollovered files (uses SimpleDateFormat syntax)
# jetty.requestlog.filenameDateFormat=yyyy_MM_dd

## How many days to retain old log files
# jetty.requestlog.retainDays=90

## Whether to append to existing file
# jetty.requestlog.append=true

## Whether to use the extended log output
#jetty.requestlog.extended=true

## Whether to log http cookie information
# jetty.requestlog.cookies=true

## Timezone of the log entries
# jetty.requestlog.timezone=GMT

## Whether to log LogLatency
# jetty.requestlog.loglatency=false
```

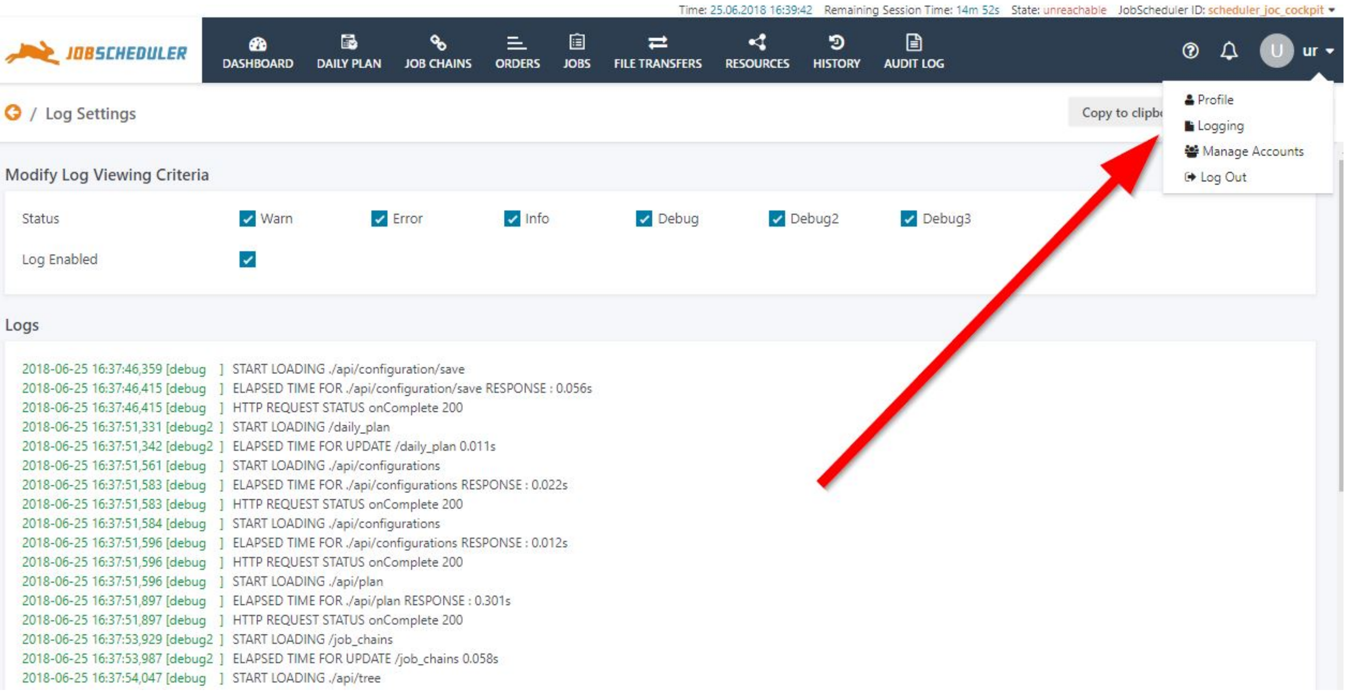
Deactivating the request log

The file `$jetty_home/start.ini` must be changed manually to deactivate the request log.

Enabling and accessing the JOC Cockpit log

The JOC Cockpit log (i.e. the log for the JOC Cockpit client) is enabled in the JOC Cockpit user interface.

- Login to the JOC Cockpit
- Click "Logging" in the drop down menu in the upper right corner
- Enable the log levels the log should include



Modify Log Viewing Criteria

Status ☑ Warn ☑ Error ☑ Info ☑ Debug ☑ Debug2 ☑ Debug3

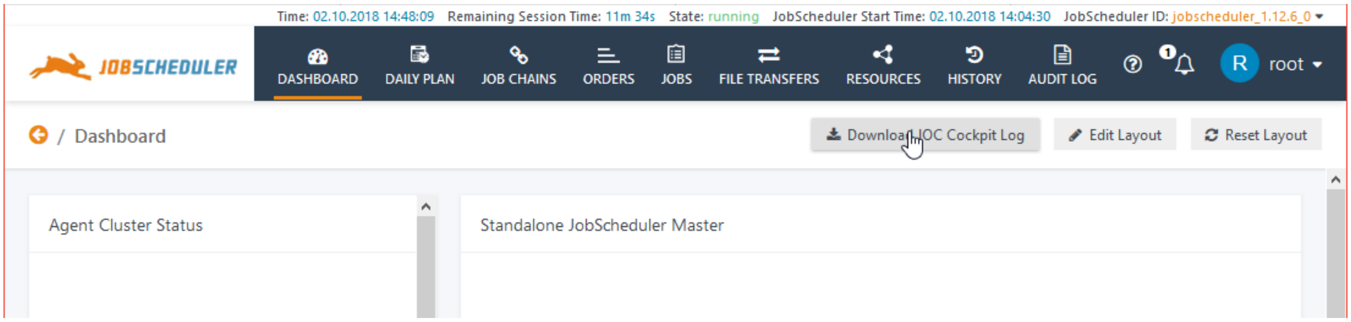
Log Enabled ☑

Logs

```
2018-06-25 16:37:46,359 [debug   ] START LOADING ./api/configuration/save
2018-06-25 16:37:46,415 [debug   ] ELAPSED TIME FOR ./api/configuration/save RESPONSE : 0.056s
2018-06-25 16:37:46,415 [debug   ] HTTP REQUEST STATUS onComplete 200
2018-06-25 16:37:51,331 [debug2 ] START LOADING /daily_plan
2018-06-25 16:37:51,342 [debug2 ] ELAPSED TIME FOR UPDATE /daily_plan 0.011s
2018-06-25 16:37:51,561 [debug   ] START LOADING ./api/configurations
2018-06-25 16:37:51,583 [debug   ] ELAPSED TIME FOR ./api/configurations RESPONSE : 0.022s
2018-06-25 16:37:51,583 [debug   ] HTTP REQUEST STATUS onComplete 200
2018-06-25 16:37:51,584 [debug   ] START LOADING ./api/configurations
2018-06-25 16:37:51,596 [debug   ] ELAPSED TIME FOR ./api/configurations RESPONSE : 0.012s
2018-06-25 16:37:51,596 [debug   ] HTTP REQUEST STATUS onComplete 200
2018-06-25 16:37:51,596 [debug   ] START LOADING ./api/plan
2018-06-25 16:37:51,897 [debug   ] ELAPSED TIME FOR ./api/plan RESPONSE : 0.301s
2018-06-25 16:37:51,897 [debug   ] HTTP REQUEST STATUS onComplete 200
2018-06-25 16:37:53,929 [debug2 ] START LOADING /job_chains
2018-06-25 16:37:53,987 [debug2 ] ELAPSED TIME FOR UPDATE /job_chains 0.058s
2018-06-25 16:37:54,047 [debug   ] START LOADING ./api/tree
```

Downloading the JOC Cockpit Log

Users with the necessary permissions can download the JOC Cockpit log file from the Dashboard view by clicking on the "Download JOC Cockpit Log" button as shown in the next screenshot.


Time: 02.10.2018 14:48:09 Remaining Session Time: 11m 34s State: running JobScheduler Start Time: 02.10.2018 14:04:30 JobScheduler ID: jobscheduler_1.12.6_0
JOBSCHEDULER
DASHBOARD
DAILY PLAN
JOB CHAINS
ORDERS
JOBS
FILE TRANSFERS
RESOURCES
HISTORY
AUDIT LOG
1
R root
/ Dashboard
Download JOC Cockpit Log
Edit Layout
Reset Layout
Agent Cluster Status
Standalone JobScheduler Master

From version 1.12.6 onwards, users require the permission `sos:products:joc_cockpit:joc:view:log` before the download button is presented to them. FEATURE AVAILABILITY STARTING FROM RELEASE 1.12.6

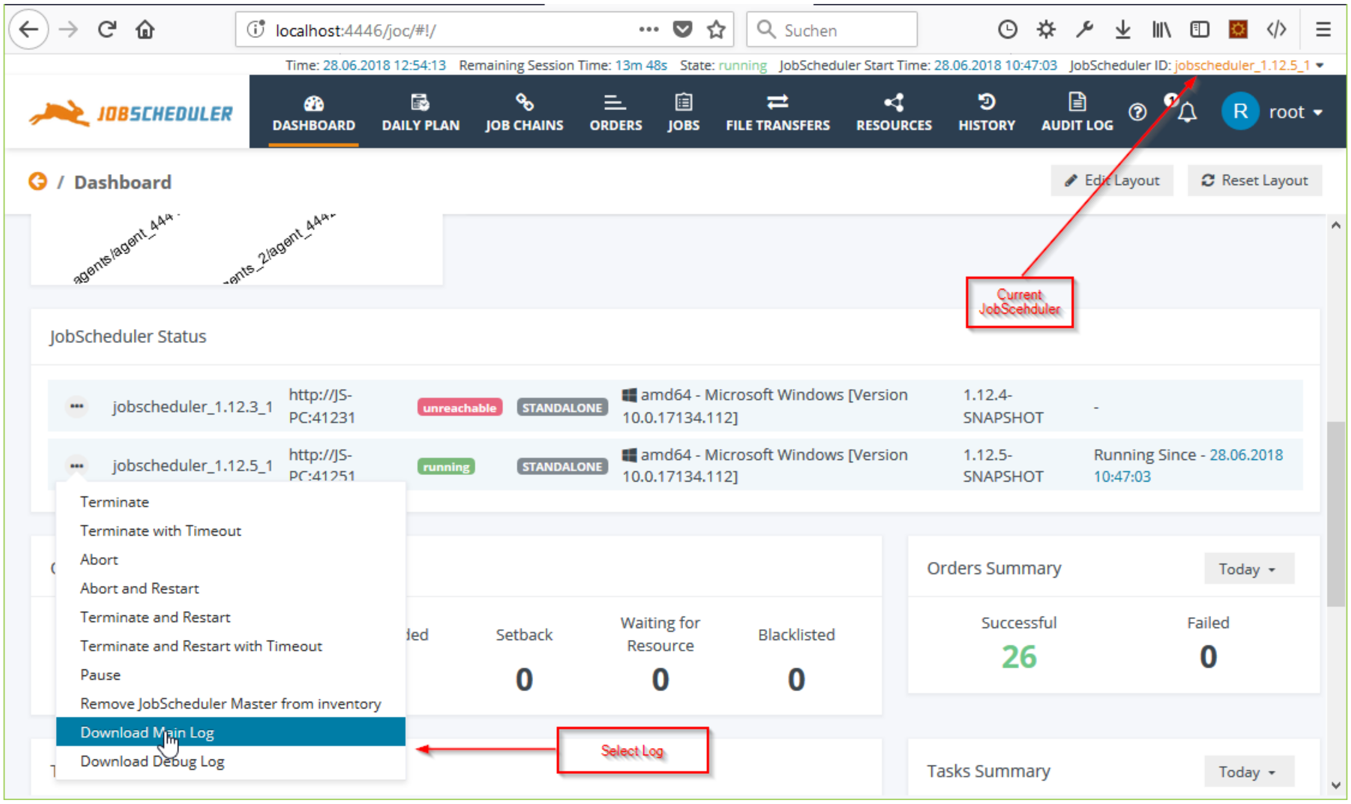
This permission is allocated to the *Administrator* role by default.

Viewing JobScheduler Log Files in the JOC Cockpit

Log files for the current JobScheduler Master can be downloaded from the JOC Cockpit Dashboard view by all account users with permissions to view the *JobScheduler Status* widget in the Dashboard.

The JobScheduler Main and Debug logs can be downloaded from the *Additional Options* menu as shown in the screenshot below. Note that log files can only be accessed in this way (as opposed to being read from the file system) when the JobScheduler Master in question does not have the status *unreachable*.

More information about the JobScheduler Log files can be found in the What logging possibilities does JobScheduler provide? article.



Time: 28.06.2018 12:54:13 Remaining Session Time: 13m 48s State: running JobScheduler Start Time: 28.06.2018 10:47:03 JobScheduler ID: jobscheduler_1.12.5_1

JobScheduler Status

jobscheduler_1.12.3_1	http://JS-PC:41231	unreachable	STANDALONE	amd64 - Microsoft Windows [Version 10.0.17134.112]	1.12.4-SNAPSHOT	-
jobscheduler_1.12.5_1	http://JS-PC:41251	running	STANDALONE	amd64 - Microsoft Windows [Version 10.0.17134.112]	1.12.5-SNAPSHOT	Running Since - 28.06.2018 10:47:03

Orders Summary: Successful 26, Failed 0

Current JobScehduler

Select Log

Custom JobScheduler Log File Locations

The default location for JobScheduler log files is the `${SCHEDULER_DATA}/logs` folder, which is set by the `log_dir` parameter in the `${SCHEDULER_DATA}/config/factory.ini` file.

Administrators are, however, free to specify an alternative location for their JobScheduler log files. In this situation, to ensure that the JOC Cockpit is able to find the alternative log file location, we recommend that administrators do not change the `log_dir` parameter in the `factory.ini` file but create a SymLink pointing to the new log directory in place of the default folder specified by the `log_dir` parameter. The JOC cockpit will then look at the default log file location and follow the SymLink to the new location.