HCLSoftware

HCL Workload Automation

Administration

Version 10.2.5

Note

Before using this information and the product it supports, read the information in Notices on page cdlxxiv.

This edition applies to version 10, release 2, modification level 5 of HCL Workload Automation (program number 5698-T09) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

List of Figures

List of Tables

About this publication

HCL Workload Automation: Administration Guide provides information about the administration of the main components of HCL Workload Scheduler (often called the *engine*).

What is new in this release

Learn what is new in this release.

For information about the new or changed functions in this release, see *Overview*, section *Summary of enhancements*.

New or changed content is marked with revision bars.

Accessibility

Accessibility features help users with a physical disability, such as restricted mobility or limited vision, to use software products successfully.

With this product, you can use assistive technologies to hear and navigate the interface. You can also use the keyboard instead of the mouse to operate all features of the graphical user interface.

For detailed information, see the appendix about accessibility in the *HCL Workload Automation User's Guide and Reference*.

Chapter 1. Customizing and configuring HCL Workload Automation

After installing the product you can customize it to fit your operational requirements. You can also change the customized values at any time. This chapter describes the optional customization steps for HCL Workload Automation. It is divided into the following sections:

- Setting global options on page 12
- Setting local options on page 48
- Setting user options on page 70
- Configuring the dynamic workload broker server on the master domain manager and dynamic domain manager on page 102
- Configuring the agent on page 71
- Configuring command-line client access authentication on page 123
- HCL Workload Automation console messages and prompts on page 132

For more information, see the sections about automating production plan processing and managing the production cycle in *User's Guide and Reference* .

Setting global options

Manages the HCL Workload Automation global options. You can list, show and change them.

Authorization

You must have the following security permissions for the global options file in the HCL Workload Automation security file to work with this command:

- For `optman ls` or `optman show`:

  ```
  FILE NAME=GLOBALOPTS ACCESS=DISPLAY
  ```

- For `optman chg`:

  ```
  FILE NAME=GLOBALOPTS ACCESS=MODIFY
  ```

See Configuring user authorization (Security file) on page 181 for more information on the security file.

Syntax

optman [-u | -v]

optman [*connectionParams*] chg {*option* | *shortName*} = *value*

optman [*connectionParams*] ls

optman [*connectionParams*] show {*option* | *shortName*}

Arguments

connectionParams

If you are using **optman** from the master domain manager, the connection parameters were configured at installation and do not need to be supplied, unless you do not want to use the default values.

If you are using **optman** from the command line client on another workstation, the connection parameters might be supplied by one or more of these methods:

- Stored in the `localopts` file
- Stored in the `useropts` file
- Supplied to the command in a parameter file
- Supplied to the command as part of the command string

For full details of the connection parameters see Configuring command-line client access authentication on page 123.

chg {*option* | *shortName*} = *value*

Change the value of an option to the new value supplied. The option can either be identified by its full or its short name. See Global options - summary on page 14 for a table showing all of the options with their full and short names, value ranges and default values. See Global options - detailed description on page 27 for a full description of each option.

ls

Lists the current values of all global options.

show {*option* | *shortName*}

Displays the current value of the indicated option. The option can either be identified by its full or its short name. See Global options - summary on page 14 for a table showing all of the options with their full and short names, value ranges and default values. See Global options - detailed description on page 27 for a full description of each option.

Comments

Some of the changes are effective immediately, but others require a specific action, such as running JnextPlan, restarting the WebSphere Application Server Liberty. These actions are indicated in the option descriptions. See *User's Guide and Reference* for more information on the JnextPlan command.

Users can decide to maintain an audit trail recording any changes they perform and the related justifications. To enable the justification option, set up in a system shell the HCL Workload Automation environment variables listed below before running any **optman** commands:

IWS_DESCRIPTION

Specify the description to be recorded for each change performed by commands in the shell. The maximum length for this value is 512 characters. A warning message displays if you exceed the maximum and excess characters are truncated.

IWS_CATEGORY

Specify the category to be recorded for each change performed by commands in the shell. The maximum length for this value is 128 characters. A warning message displays if you exceed the maximum and excess characters are truncated.

IWS_TICKET

Specify the ticket to be recorded for each change performed by commands in the shell. The maximum length for this value is 128 characters. A warning message displays if you exceed the maximum and excess characters are truncated.

For more information about the justification option, see the section about keeping track of changes in *Dynamic Workload Console User's Guide*.

Example

Examples

Example 1: list the global options

To list all of the global options, when your connection parameters are supplied via the `localopts` and `useropts` files, give the following command:

```
optman ls
```

Example 2: show the value of a global option

To show the current value of the `enCarryForward` global option, identifying it by its short name, give the following command:

```
optman show cf
```

Example 3: change the value of a global option

To change the current value of the `enCarryForward` global option, identifying it by its full name, give the following command:

```
optman chg enCarryForward no
```

Global options - summary

This section summarizes the global options that are managed by optman. The columns in the tables have the following meanings:

Description

The brief description of the option

Name

The option as used in the optman commands.

Short name

The shortName as used in the optman commands.

Default

The default value that is applied to the option at installation (if present).

Range

The range or choice of values you can supply (where appropriate).

Units

The units that apply to the default and range.

Effect

How to make any changes effective. The following codes have been used:

E

If you are enabling the option, start the Event Processor. If you are disabling the option, stop the Event Processor.

Imm

The change is effective immediately

Imm (DB)

The change is effective immediately in the database only.

J

Run JnextPlan.

J (Plan)

Run JnextPlan - it makes the change effective in the plan only.

NSJ

The change is effective on the next submit job stream action.

NSM

The change is effective on the next send mail action.

NOC

The change is effective on the next change performed on a security object.

W

Restart WebSphere Application Server Liberty.

The following tables summarize the global options for managing the features and functions of HCL Workload Automation:

Table 1. Workload service assurance feature

Description	Name	Short name	Default	Range	Units	Effect
Enable workload service assurance	enWorkloadServiceAssurance	wa	yes	yes, no	boolean	J
Approaching late offset	approachingLateOffset	al	120	>=0	seconds	J or W
Deadline offset	deadlineOffset	do	2	>=0	minutes	J or W
Promotion offset	promotionOffset	po	120	>=0	seconds	J
Enable forecast start time calculation	enForecastStartTime	st	no	yes, no	boolean	imm

Table 2. Condition-based workflow automation

Description	Name	Short name	Default	Range	Units	Effect
Name of the job which is automatically added to the plan to run the file monitoring task.	fileStartConditionJobName	fc	file_StartCond	40 bytes		Imm
Name of the job which is automatically added to the plan to resubmit a new instance of the job stream where the start condition is defined.	resubmitJobName	rj	restart_StartCond	40 bytes		Imm
Default offset set for the start condition deadline.	startConditionDeadlineOffset	cd	2400	0001 - 9959	hhmm	Imm
Prevent job streams from completing in error when the start condition is not met	enStartCondSuccOnDeadline	od	**Fresh installation** yes	yes - no	boolean	J

Table 2. Condition-based workflow automation (continued)

Description	Name	Short name	Default	Range	Units	Effect
			Upgrade no			

Table 3. Event-driven workload automation feature - general

Description	Name	Short name	Default	Range	Units	Effect
Enable event driven workload automation	enEventDrivenWorkloadAutomation	ed	yes	yes, no	boolean	J or E
Rules deployment frequency	deploymentFrequency	df	5	0-60	minutes	Imm
Enable event processor HTTPS protocol	enEventProcessorHttpsProtocol	eh	yes	yes, no	boolean	J
HCL event integration facility port for SSL	eventProcessorEIFSslPort	ef	31131	0 - 65535	port number	W and J
HCL event integration facility port	eventProcessorEIFPort	ee	31131	0 - 65535	port number	W and J
EIF Probe server name (used both for events in TEC and TBSM formats)	TECServerName	th	localhost		name	J
EIF Probe server port (used both for events in TEC and TBSM formats)	TECServerPort	tp	5529	0 65535	port number	J

Table 4. Event-driven workload automation feature - event mailing

Description	Name	Short name	Default	Range	Units	Effect
Mail sender name	mailSenderName	ms	TWS		name	NSM
SMTP server name	smtpServerName	sn	localhost		name	Imm

Table 4. Event-driven workload automation feature - event mailing (continued)

Description	Name	Short name	Default	Range	Units	Effect
SMTP Server port	smtpServerPort	sp	25	0 65535	port number	NSM
Mail plug-in uses SMTP authentication	smtpUseAuthentication	ua	no	yes, no	boolean	Imm
SMTP user name	smtpUserName	un	TWS_user		name	Imm
SMTP user password	smtpUserPassword	up				Imm
Mail plug-in uses SSL	smtpUseSSL	us	no	yes, no	boolean	Imm
Mail plug-in uses TLS protocol	smtpUseTLS	tl	no	yes, no	boolean	Imm

Table 5. Event-driven workload automation feature - HCL Workload Automation for Z plug-in

Description	Name	Short name	Default	Range	Units	Effect
HCL Workload Automation for Z connector remote server name	zOSRemoteServerName	zr			name	NSJ
HCL Workload Automation for Z connector server name	zOSServerName	zs	localhost		name	NSJ
HCL Workload Automation for Z connector server port	zOSServerPort	zp	31217	0 65535	port number	NSJ
HCL Workload Automation for Z connector user name	zOSUserName	zu	TWS_user		name	NSJ
HCL Workload Automation for Z connector user password	zOSUserPassword	zw				NSJ

Table 6. SSL

Description	Name	Sh ort n ame	Default	Range	Units	Effect
Enable strong password encryption	enStrEncrypt	se	no	yes, no	boolean	J
Enable the SSL full connection	enSSLFullConnection	sf	no	yes, no	boolean	J
Defines the encryption algorithm used by HCL Workload Automation	useAESEncryptionAlgorithm	ea	N/A	yes, no	boolean	N/A

Table 7. Job management

Description	Name	Sh ort n ame	Default	Range	Units	Effect
Maximum prompts after abend	baseRecPrompt	bp	1000	0 65535	prompts	J
Additional prompts after abend	extRecPrompt	xp	1000	0 65535	prompts	J
Concurrent access to resources	enExpandedResources	er	yes	yes, no	boolean	J
Automatically grant logon as batch	enLogonBatch	lb	no	yes, no	boolean	J
Long duration job threshold	longDurationThreshold	ld	150	100 - 1000	seconds	J or W
User for binding to remote jobs from shadow job	bindUser	bu	TWS_user			Imm

Table 8. Job stream management

Description	Name	Sh ort n ame	Default	Range	Units	Effect
Job streams without jobs policy	enEmptySchedsAreSucc	es	no	yes, no	boolean	J

Table 8. Job stream management (continued)

Description	Name	Sh ort n ame	Default	Range	Units	Effect
Prevent job stream without "at" dependency from starting	enPreventStart	ps	yes	yes, no	boolean	J

Table 9. Stageman

Description	Name	Sh ort n ame	Default	Range	Units	Effect
Carry job states	carryStates	cs	null		list of states	J
Enable carry forward	enCarryForward	cf	all	all, no	boolean	J
Enable carry forward for internetwork dependencies	enCFinterNetworkDeps	ci	yes	yes, no	boolean	J
Enable carry forward resource quantity	enCFResourceQuantity	rq	yes	yes, no	boolean	J
Retain rerun job name	enRetainNameOnRerunFrom	rr	no	yes, no	boolean	J
Remove obsolete job streams	untilDays	ud	0	>=0	days	J

Table 10. Planman

Description	Name	Sh ort n ame	Default	Range	Units	Effect
Maximum preproduction plan length	maxLen	xl	8	8 - 365	days	J
Minimum preproduction plan length	minLen	ml	8	7 - 365	days	J

Table 11. Logging and auditing

Description	Name	Short name	Default	Range	Units	Effect
Log cleanup frequency	logCleanupFrequency	lc	5	0 - 60	minutes	J
Log history period	logHistory	lh	10	>=0	days	J
Logman minimum and maximum run time policy	logmanMinMaxPolicy	lm	both		literal	J
Logman normal run time calculation policy	logmanSmoothPolicy	lt	-1	-1 - 100	factor	J
Enable database auditing	enDbAudit	da	0	0, 1	boolean	Imm
Type of store to be used to log database audit records	auditStore	as	both	db, file, both		Imm
Audit history period	auditHistory	ah	180	>=1	days	Imm
Enable auditing of database GET operations	enDbGetOpsAudit	dg	1	0, 1	boolean	Imm

Table 12. Cross dependencies

Description	Name	Short name	Default	Range	Units	Effect
Number of days for retrying to send notifications about job status changes to the remote engine if the notification fails	notificationTimeout	nt	5	1-90	Number	Imm

Table 13. ServiceNow

Description	Name	Short name	Default	Range	Units	Effect
Address of the ServiceNow server	servicenowUrl	nu			name	Imm
User who connects to the ServiceNow server	servicenowUserName	nn			name	Imm
Password associated with the user who connects to the ServiceNow server	servicenowUserPassword	np			name	Imm

Table 14. Automatic failover

Description	Name	Short name	Default	Range	Units	Effect
Enables or disables the automatic failover feature which invokes an automatic switch from the master domain manager, event manager, or both, to a backup.	enAutomaticFailover	af	yes	yes, no	boolean	W
Enables or disables automatic failover actions, such as, the automatic switch of the master or automatic restart of the fault-tolerant agent. This option takes effect only if the enAutomaticFailover option is set to `yes`	enAutomaticFailoverActions	aa	yes	yes, no	boolean	W
A comma-separated list of workstation names, including the current event manager workstation, that serve as backups for the event manager workstation when the automatic failover feature is enabled. If an eligible workstation is defined in a folder, use the composer li ws @;showid command to retrieve the ID	workstationEventMgrListInAutomaticFailover	we		comma-separated list of workstation names. The	name	W

Table 14. Automatic failover (continued)

Description	Name	Short name	Default	Range	Units	Effect
of the workstation you plan to define as backup.				maximum length is 256 bytes.		
A comma-separated list of workstations, including the current master domain manager, that serve as backups for the master domain manager when the automatic failover feature is enabled. If an eligible workstation is defined in a folder, use the composer li ws @;showid command to retrieve the ID of the workstation you plan to define as backup.	workstationMasterListInAutomaticFailover	wm		comma-separated list of workstation names. The maximum length is 256 bytes.	name	W

Table 15. Licensing configuration

Description	Name	Short name	Default	Range	Units	Effect
Password of the proxy server.	licenseProxyPassword	pw	N/A		string	Imm
IP of the proxy server.	licenseProxyServer	lp	N/A		number	Imm
Port of the proxy server.	licenseProxyServerPort	lo	N/A		number	Imm
User of the proxy server.	licenseProxyUser	pb	N/A		string	Imm
Token obtained from the license server	licenseRefreshToken	rt	N/A			Imm

Table 15. Licensing configuration (continued)

Description	Name	Short name	Default	Range	Units	Effect
URL of the license server which processes license usage information.	licenseServerUrl	lu	https://api.hcltechsw.com/			Imm
Type of accepted license for HCL Workload Automation	licenseType	ln	ws	ws, wa , byworkstation		J
Specify the default license type for HCL Workload Automation workstations.	defaultWksLicenseType	wn	**In a Docker environment** perJob in a fresh installation and p	• PERSERVER • PERJOB		J

Table 15. Licensing configuration (continued)

Description	Name	Short name	Default	Range	Units	Effect
			`erServer` in upgrade. **In an on-premises environment** `perServer`			

Table 16. General

Description	Name	Short name	Default	Range	Units	Effect
Company name	companyName	cn			name	J
Delete folders	folderDays	fd	10	0 - 10	days	J

Table 16. General (continued)

Description	Name	Short name	Default	Range	Units	Effect
Enable centralized security in the classic security model	enCentSec	ts	no	yes, no	boolean	J
Evaluate start-of-day	enLegacyStartOfDayEvaluation	le	no	yes, no	boolean	J
Enable list security check	enListSecChk	sc	no	yes, no	boolean	J (Plan) Imm (DB)
Enable plan auditing	enPlanAudit	pa	0	0, 1	boolean	Imm
Enable security file creation in the role-based security model	enRoleBasedSecurityFileCreation	rs	no	yes,no	boolean	Imm
Enable extended field support in the security file.	enSecFileExtendedFields	sl	no	yes, no	boolean	NOC
Enable the fault-tolerant switch manager	enSwfaultTol	sw	no	yes, no	boolean	J
Enable time zones	enTimeZone (deprecated)	tz	yes	yes, no	boolean	J (Plan) Imm (DB)
Enable What-if Analysis	enWhatIfAnalysis	wi	yes	yes, no	boolean	J
Ignore calendars	ignoreCals	ic	no	yes, no	boolean	J
Start time of processing day	startOfDay	sd	0000	0000 2359	hhmm	J
Job statistics history period	statsHistory	sh	10	>=0	days	J (Plan) Imm (DB)
Critical Jobs Risk Confidence	riskConfidence	rc	80% in fresh installation, 50% in upgrade	1-99	Number	Imm

Global options - detailed description

This section gives full descriptions of the global options managed by optman:

approachingLateOffset | al

Approaching late offset. Used in workload service assurance. The critical start time of a job in the critical network is the latest time that the job can start without causing the critical job to finish after the deadline. In most cases, a job will start well before the critical start time so that if the job runs longer than its estimated duration, the situation does not immediately become critical. Therefore, if a job has not started and the critical start time is only a few minutes away, the timely completion of the critical job is considered to be potentially at risk.

The *approachingLateOffset* option allows you to determine the length of time before the critical start time of a job in the critical network at which you are to alerted to this potential risk. If a job has still not started the specified number of seconds before the critical start time, the job is added to a hot list that can be viewed on the Dynamic Workload Console.

Note: To qualify for addition to the hot list, all time and follow dependencies must have been resolved.

This option is only active if *enWorkloadServiceAssurance* is set to *yes*.

The default is *120* seconds.

Note: Whatever value you set for this option, if HCL Workload Automation loses the connection with its database, the default value is applied to critical job processing, and the warning message AWSJCO135W is issued to tell you what has happened.

Run JnextPlan or restart WebSphere Application Server Liberty (stopappserver and startappserver) to make this change effective.

auditHistory | ah

Audit history period. Used in audit management. This setting applies only when the **auditStore** option is set to `db`. Enter the number of days for which you want to save audit record data. Audit records are discarded on a FIFO (first-in first-out) basis.

The default value is *400* days. This option takes effect immediately.

For more information about auditing, see Auditing facilities on page 366.

auditStore | as

Type of store to be used to log database and plan audit records. Enter one of the following:

file

To specify that a flat file in the `<install_dir>/TWSDATA/audit/database` directory on the master domain manager and backup master domain manager is used to store the audit records.

db

To specify that the HCL Workload Automation database itself is used to store the audit records.

both

To have audit records logged in both the file and the database.

The default value is `both`. Any change of this value is effective immediately.

Note: When you upgrade the master domain manager from a previous release, the default value for this global option is changed. The default value is now **both**. The reason for this change is to support the auditing feature which introduces reporting, versioning and rollback functions for database objects. If you customized the default value in the previous release, the value is overwritten with the new value with the exception of the **auditStore** option with the **DB** value assigned. If the **auditStore** option was set to **DB**, this value is maintained and is not overwritten.

For more information about auditing, see Auditing facilities on page 366.

baseRecPrompt | bp

Maximum prompts after abend. Specify the maximum number of prompts that can be displayed to the operator after a job abends.

The default value is *1000*. Run JnextPlan to make this change effective.

bindUser | bu

User for binding to remote jobs from shadow job. Specify the user ID that is used to bind a shadow job to a remote job during the security check for "cross dependencies". This user must be given at least the following authorizations in the security file:

- *Display* access to the *job* and *schedule* objects that need to be bound
- *List* access to *job* objects that need to be bound

However, the ID does not need to be in the user registry of the engine, nor have a password, as it is only required for authorization purposes.

The default value is the TWS_user. Any change of this value is effective immediately.

carryStates | cs

Carry job states. A preproduction option that affects the operation of the *stageman* command. Specify the jobs, by state, to be included in job streams that are carried forward. Enclose the job states in parentheses, double quotation marks, or single quotation marks. Commas can be replaced by spaces. The valid internal job states are as follows:

Table 17. Valid internal job states

abend	*abenp*	*add*	*bound*	*done*	*error*	*exec*

Table 17. Valid internal job states (continued)

fail	*hold*	*intro*	*pend*	*ready*	*rjob*	*sched*
skel	*succ*	*succp*	*suppr*	*susp*	*wait*	*waitd*

Some examples of the option are as follows:

```
carryStates="abend,exec,hold,intro"
carryStates='abend,exec,hold,intro'
carryStates="abend, exec, hold, intro"
carryStates='abend, exec, hold, intro'
```

An empty list is entered as follows:

```
carryStates=null
```

The default value is *null*, which corresponds to selecting all states. Run JnextPlan to make this change effective.

companyName | cn

Company name. Specify the name of your company. The maximum length is 40 bytes. If the name contains spaces, enclose the name in double quotation marks ("). If you use the Japanese-Katakana language set, enclose the name within single or double quotation marks.

Run JnextPlan to make this change effective.

deadlineOffset | do

Deadline offset. Used in workload service assurance. Used to calculate the critical start of a critical job in the case where a deadline has not been specified neither for the job nor its job stream. In this case the deadline is defaulted to the plan end date and time, plus this offset, expressed in minutes.

This option is only active if *enWorkloadServiceAssurance* is set to *yes*.

The default is *2* minutes.

Note:

1. **Important**: When the plan is extended, the start time of critical jobs with a deadline calculated with this mechanism is automatically changed as a consequence of the fact that it must now match the new plan finishing time.
2. Whatever value you set for this option, if HCL Workload Automation loses the connection with its database, the default value is applied to critical job processing, and the warning message AWSJCO135W is issued to tell you what has happened.

Run JnextPlan or restart WebSphere Application Server Liberty (stopappserver and startappserver) to make this change effective.

defaultWksLicenseType / wn

Specify the default licensing model for HCL Workload Automation workstations. This option is supported only if the licenseType option is set to `byWorkstation` and a specific value was not specified at creation time for each workstation. For more information, see licenseType | ln on page 39. Supported values are as follows:

perServer

to specify the HCL Workload Automation Processor Value Unit (PVU) pricing.

perJob

to specify the HCL Workload Automation Per Job (PJ) pricing.

The default value varies, depending on the environment type:

In a Docker environment

By default, the defaultWksLicenseType option is set to `perJob` in a fresh installation and to `perServer` during product upgrade. By default, the licenseType option is set to `byWorkstation` in a fresh installation. This setting is not modified during product upgrade. As a result, the value of the defaultWksLicenseType option is applied to the creation of all workstation types.

In an on-premises environment

By default, the licenseType option is set to `perServer`, and also defaultWksLicenseType option is set to `perServer`. Before changing this value, contact your sales representative.

deploymentFrequency | df

Rules deployment frequency. Used in event rule management. Specify the frequency, in minutes, with which rules are to be checked to detect if there are changes to deploy. All active rules (active rules have the `isDraft` property set to `no` in their definition) that have been changed or added since the last deployment are deployed.

Valid values are in the *0-60* minutes range. If you specify *0*, the changes are not deployed automatically and you must use the planman deploy command.

The default value is *5* minutes. The change is effective immediately.

enAddUser | au

Enable the automatic user addition into the Symphony file. This option enables the automatic addition of a user into the Symphony file after you create or modify the user in the database. If you specify "yes", the user is automatically added to the Plan. If you specify "no", the user is not automatically added to the Plan.

The default value is "yes". Changes to this parameter are effective immediately.

For more information about how to use this feature, see "HCL Workload Automation: User's Guide and Reference".

enAddWorkstation | aw

Enable the automatic dynamic agent, pool, and dynamic pool workstation addition into the Symphony file. This option enables the automatic addition of a dynamic agent, pool, or dynamic pool workstation into the

Symphony file after you created the workstation in the database. If you specify "yes", the workstation is automatically added to the Plan. If you specify "no", the workstation is not automatically added to the Plan.

The default is "no". Changes to this parameter are effective immediately.

For more information about how to use this feature, see *HCL Workload Automation: User's Guide and Reference*.

enAutomaticFailover | af

Enable or disable the automatic failover feature. This option enables or disables the automatic failover feature which invokes an automatic switch from the master domain manager, event manager, or both, to a backup workstation. Eligible backups for both the master domain manager and the event manager can be specified using the optman options, workstationMasterListInAutomaticFailover and workstationEventMgrListInAutomaticFailover, respectively. For more information about this feature, see Automatic failover on page 393.

A fresh installation of HCL Workload Automation V9.5FP2 or later enables this feature by default (`yes`). This feature is disabled (`no`) when you upgrade to HCL Workload Automation V9.5FP2 or later. Changes to this parameter require restarting WebSphere Application Server Liberty.

enAutomaticFailoverActions | aa

Enable or disable the automatic failover actions. This option enables or disables automatic failover actions, such as, the automatic switch of the master or the automatic restart of the fault-tolerant agent. This option takes effect only if the enAutomaticFailover option is set to `yes`. You can set this option to `no` in the case of a planned maintenance window.

By default, this option is set to `yes`. Changes to this parameter require restarting WebSphere Application Server Liberty.

enCarryForward | cf

Enable carry forward. A preproduction option that affects the operation of the *stageman* command. Specify if job streams that did not complete are carried forward from the old to the new production plan (Symphony). Enter `yes` to have incompleted job streams carried forward only if the *Carry Forward* option is enabled in the Job Scheduler definition. Enter `all` to have all incomplete job streams carried forward, regardless of the *Carry Forward* option. Enter `no` to completely disable the *Carry Forward* function. If you run the `JnextPlan -for 0000` command and the *Carry Forward* option is set to either `yes` or `no`, a message is displayed informing you that incompleted job streams might not be carried forward. When the stageman -carryforward command is used, it overrides *enCarryForward*. See *HCL Workload Automation: User's Guide and Reference* for more information. If this option is set to no, running jobs are moved to the USERJOBS job stream.

The default value is *all*. Run JnextPlan to make this change effective.

enCentSec | ts

Enable centralized security. In the classic security model, determine, how the security file is used within the network. Centralized security is not relevant to an end-to-end scheduling environment.

If set to *yes*, the security files of all the workstations of the network can be created and modified only on the master domain manager. In this case, the HCL Workload Automation administrator is responsible for their production, maintenance, and distribution.

If set to *no*, the security file of each workstation can be managed by the root user or administrator of the system. The local user can run the *makesec* command to create or update the file.

See Centralized security management on page 213 for more information about centralized security.

The default value is *no*. Run JnextPlan to make this change effective.

Note: This option does not apply to role-based security model.

enCFinterNetworkDeps | ci

Enable carry forward for internetwork dependencies. A preproduction option that affects the way stageman handles internetwork dependencies. It specifies if external job streams are carried forward from the old to the new production plan (Symphony file). Enter *yes* to have all external job streams carried forward. Enter *no* to have no external job streams carried forward.

The default value is *yes*. Run JnextPlan to make this change effective.

enCFResourceQuantity | rq

Enable carry forward resource quantity. A preproduction option that affects the way stageman handles resources. Enter *yes* to carry forward the resource quantity from the old production file to the new. Enter *no* to not carry forward the resource quantity. Stageman carries forward resource quantities only if the resource is needed by a job or job stream that is also being carried forward. Otherwise the resource quantities are set to the original value. See *HCL Workload Automation: User's Guide and Reference* for details on using this feature.

The default value is *yes*. Run JnextPlan to make this change effective.

enDbAudit | da

Enable auditing on information available in the database. Enable or disable auditing on information available in the database. To enable auditing on information available in the database, specify *1*. To disable auditing on information available in the database, specify *0*. Auditing information is logged to a flat file in the `TWA_home/TWS/audit/database` directory, to the HCL Workload Automation database itself, or to both. To choose which, set the optman property *auditStore*. Each HCL Workload Automation workstation maintains its own log. Only actions are logged, not the success or failure of the action. Installation of dynamic domain managers and agents is not recorded in audit logs.

The default value is *1*. Changes to this parameter are effective immediately.

Note: When you upgrade the master domain manager from a previous release, the default value for this global option is changed. The default value is now **1**. The reason for this change is to support the auditing feature which introduces reporting, versioning and rollback functions for database objects. If you customized the default value in the previous release, the value is overwritten with the new value.

For more information about auditing, see Auditing facilities on page 366.

enDbGetOpsAudit

Enable auditing of GET database operations. Enable or disable auditing on database **GET** operations. Disabling this auditing feature might improve performance. All other database operations are not affected. To disable auditing on database **GET** operations, specify *0*. To enable auditing on database **GET** operations, specify *1*. This parameter is effective only *if* general database audit is enabled. **enDbAudit=1)**.

The default value is *1*. Changes to this parameter are effective immediately.

workstationEventMgrListInAutomaticFailover | we

A list of workstations eligible to serve as a backup for the event manager. A comma-separated list of workstations that serve as backups for the event manager, including the current event manager itself, when the automatic failover feature is enabled. The maximum length is 256 bytes. If an eligible workstation is defined in a folder, use the composer li ws @;showid command to retrieve the ID of the workstation you plan to define as backup.

If no workstations are specified in this list, then all backup master domain managers in the domain are eligible backups. Changes to this parameter require restarting WebSphere Application Server Liberty.

workstationMasterListInAutomaticFailover | wm

A list of workstations eligible to serve as a backup for the master. A comma-separated list of workstations that serve as backups for the master domain manager, including the current master domain manager itself, when the automatic failover feature (**enAutomaticFailover**) is enabled. The maximum length is 256 bytes. If an eligible workstation is defined in a folder, use the composer li ws @;showid command to retrieve the ID of the workstation you plan to define as backup.

If no workstations are specified in this list, then all backup master domain managers in the domain are considered eligible backups. Changes to this parameter require restarting WebSphere Application Server Liberty.

useAESEncryptionAlgorithm | ea

AES algorithm used for passwords Specify whether or not the AES encryption algorithm is used for encrypting passwords. The AES encryption algorithm is a widely adopted encryption algorithm used to secure digital data. It is fast, reliable, and considered one of the most secure symmetric encryption methods available today. It is compliant with FIPS. If set to `yes`, it indicates that user passwords are encrypted with the AES algorithm. If set to `no`, it indicates that a different encryption algorithm is used.

enEmptySchedsAreSucc | es

Job streams without jobs policy. Specify the behavior of job streams without any jobs. If set to *yes*, the job streams that contain no jobs are set to SUCC after their dependencies are resolved. If set to *no*, the job streams are left in READY status.

The default value is *no*. Run JnextPlan to make this change effective.

enEventDrivenWorkloadAutomation | ed

Enable event-driven workload automation. Enable or disable the event-driven workload automation feature. To enable, specify *yes*. To disable, specify *no*.

The default value is *yes*.

After disabling, you must run JnextPlan and stop the event processing server (with the conman stopevtp command).

After enabling, you must run JnextPlan and start the event processing server (with the conman startevtp command).

enEventDrivenWorkloadAutomationProxy | pr

Enable event-driven workload automation proxy. Enable or disable the event-driven workload automation proxy feature. To enable, specify *yes*. To disable, specify *no*.

The default value is *no*. Run JnextPlan to make this change effective.

enEventProcessorHttpsProtocol | eh

Enable event processor HTTPS protocol. Used in event rule management. Enables or disables the use of the HTTPS protocol to connect to the event processor server. To enable, enter *yes*. To disable, enter *no*.

The default value is *yes*. Run JnextPlan to make this change effective.

enExpandedResources

Enables up to 60 concurrent holders for an HCL Workload Automation resource. Enter `yes` to enable up to 60 concurrent holders for a resource. Enter `no` to disable the feature and use only 32 holders for a resource.

The default value is `yes`. Run JnextPlan to make this change effective.

enForecastStartTime | st

Enable forecast start time. Only applicable when workload service assurance is enabled (see *enWorkloadServiceAssurance*). Enter *yes* to enable the calculation of the predicted start time of each job when running a forecast plan: this option is recommended if you want to take advantage of the enhanced forecast capability that calculates the start time of each job considering the estimated duration of its predecessor jobs. Enabling this feature could negatively impact the time taken to generate the forecast plan. Enter *no* to disable the calculation of the predicted start time of each job when running a forecast plan.

The default value is *no*. Any change of this value is effective immediately.

When this option is set to *yes*, the **enPreventStart** global option is ignored during the creation of forecast plans.

enLegacyStartOfDayEvaluation | le

Evaluate start-of-day. Specify how the *startOfDay* option is to be managed across the HCL Workload Automation network. This is a legacy setting and should always be set to *no* starting from release 9.4.0 and later. If you set this option to *yes*, the *startOfDay* value on the master domain manager is converted to the local time zone set on each workstation across the network. If you set this option to *no*, the *startOfDay* value on the master domain manager is applied as is on each workstation across the network. This option requires that the *enTimeZone* option is set to *yes* to become operational.

The default value is *no*. Run JnextPlan to make this change effective.

enListSecChk | sc

Enable list security check. Control the objects in the database and the plan that a user is permitted to list when running a query on the Dynamic Workload Console or HCL Workload Automation database, for example running a composer list, or a conman show command. If set to *yes*, objects in the plan returned from a query or show command are shown to the user only if the user has been granted the list permission in the security file. If set to *no*, all objects are shown, regardless of the settings in the security file.

Note: Setting this option to *yes* affects how the graphical user interfaces function for the users defined in the security file.

The default value is *no*. Run JnextPlan to make this change effective for the plan. For the database, this option takes immediate effect.

enLogonBatch | lb

Automatically grant logon as batch. This is for Windows® jobs only. If set to *yes*, the logon users for Windows® jobs are automatically granted the right to *Logon as batch job*. If set to *no*, or omitted, the right must be granted manually to each user or group. The right cannot be granted automatically for users running jobs on a backup domain manager, so you must grant those rights manually.

The default value is *no*. Run JnextPlan to make this change effective.

enPlanAudit | pa

Enable plan auditing. Enable or disable auditing on information available in the plan. To enable auditing on information available in the plan, specify *1*. To disable auditing on information available in the plan, specify *0*. Auditing information is logged to a flat file, to the HCL Workload Automation database itself, or to both. To define the logging location, set the **auditStore** global option. For more information, see auditStore | as on page 27. The audit file is located in the following path on the master domain manager and backup master domain manager:

```
TWA_home\TWS\audit\database
```

```
TWA_DATA_DIR/audit/database
```

For the plan, only actions are logged in the auditing file, not the success or failure of any action.

For more information about auditing, see Auditing facilities on page 366.

The default value is *1*. Changes to this parameter are effective immediately.

Note: When you upgrade the master domain manager from a previous release, the default value for this global option is changed. The default value is now **1**. The reason for this change is to support the auditing feature which introduces reporting, versioning and rollback functions for database objects. If you customized the default value in the previous release, the value is overwritten with the new value.

enPreventStart | ps

Prevent job stream without "at" dependency from starting. Specify if job streams without an *at* dependency are to be prevented from starting immediately, without waiting for the beginning of the day the run cycle specified in the Job Scheduler identifies. Valid values are *yes* and *no*.

The default value is *yes*. Run JnextPlan to make this change effective.

When the **enForecastStartTime** option is set to *yes*, this option is ignored during the creation of forecast plans.

enRetainNameOnRerunFrom | rr

Retain rerun job name. A production option that affects the operation of Batchman, the production control process of HCL Workload Automation. Its setting determines if jobs that are rerun with the Conman *rerun* command retain their original job names. To have rerun jobs retain their original job names, enter *yes*. Enter *no* to assign the *rerun from* name to rerun jobs.

The default value is *no*. Run JnextPlan to make this change effective.

Note: Starting from version 10.1, this option is deprecated and must not be modified. By default, its value is set to **no**.

enRoleBasedSecurityFileCreation | rs

Enable the role-based security model. This option enables the automatic creation of the security file using the role-based security model. You define the role-based security model in the master domain manager database by using the **Manage Workload Security** interface from Dynamic Workload Console or the **composer** command-line program.

The default value is *yes*, which means that the role-based security model is enabled for your installation.

At any time, specify *no* if you want to disable the role-based security model and replace your current security file. You can then continue to use the classic security model, that allows you to update your security file by using dumpsec and makesec commands from the command line.

For more information about user authorization and classic and role-based security models, Configuring user authorization (Security file) on page 181.

Changes to this parameter are effective immediately.

enSecFileExtendedFields

Enable extended fields support in the security file. Enable long attribute values for all scheduling objects in the security file. When this option is enabled, it permits the use of the extended version of the security file with the attribute field value length set to 64K rather then 255 bytes.

The default value is *no*. This change becomes effective the first time you edit a security object.

enSSLFullConnection | sf

Enable the SSL full connection. Specify that HCL Workload Automation uses a higher level of SSL connection than the standard level. For full details see Configuring full SSL security on page 330. Valid values are *yes* to enable the SSL full connection or *no* to disable the SSL full connection.

The default value is *no*. Run JnextPlan to make this change effective.

enStartCondSuccOnDeadline | od

Prevent job streams from completing in error when the start condition is not met Specify the behavior of the job stream when the start condition is not met and the **Start once** option is not selected. If you set the **Start once** option, this option is ignored. If you set the option to `yes`, when the deadline for the start condition is met, the monitoring job is confirmed in **Successful** status and the job stream is canceled. If you set the option to `no`, the monitoring job is killed, so both the monitoring job and the job stream change to **Error** status.

The default value is `yes` in a fresh installation and `no` in upgrade to maintain compatibility with previous versions. Run JnextPlan to make this change effective.

enSwfaultTol | sw

Enable the fault-tolerant switch manager. Enable or disable the fault-tolerant switch manager feature. Valid values are *yes* to enable the fault tolerant switch manager, and *no* to disable it. This option has not dynamic capabilities and is not designed to work with broker agents. It applies to fault-tolerant agents. See the *HCL Workload Automation: User's Guide and Reference* for more details.

The default value is *no*. Run JnextPlan to make this change effective.

Note: Starting from version 10.1, this option is deprecated and must not be modified. By default, its value is set to **no**.

enTimeZone | tz

Enable time zones. Enables the time zone option.

Note: Starting from version 10.1, this option is deprecated and must not be modified. By default, its value is set to **yes**.

enWhatIfAnalysis | wi

Enable What-if Analysis. Enables or disables What-if Analysis, which is the feature that shows plan activities displayed against time and give you a visual representation of your plan at a glance in real time. To enable What-if Analysis, specify *yes*. To disable What-if Analysis, specify *no*. See *Dynamic Workload Console User's Guide* for details on using this feature..

The default value is *yes*. Run JnextPlan to make this change effective.

enWorkloadServiceAssurance | wa

Enable workload service assurance. Enables or disables workload service assurance, which is the feature that manages the privileged processing of mission critical jobs and their predecessors. Specify *yes* to enable or *no* to disable.

Note: Before starting to use workload service assurance you must set up the *TWS_user* in the security file to have the appropriate access to the objects that this feature will modify - see The TWS_user - special security file considerations on page 254

The default value is *yes*. Run JnextPlan to make this change effective.

eventProcessorEIFSslPort | ef

Tivoli® event integration facility port. Used in event rule management. Specify the port number for SSL where the event processor server receives events from the Tivoli® Event Integration Facility (EIF). Valid values are in the *0-65535* range.

The default value is *31131*. If you change the value, restart WebSphere Application Server Liberty (stopappserver and startappserver) and run JnextPlan to make this change effective.

eventProcessorEIFPort | ee

Tivoli® event integration facility port. Used in event rule management. Specify the port number where the event processor server receives events from the Tivoli® Event Integration Facility (EIF). Valid values are in the *0-65535* range.

The default value is *31131*. If you change the value, restart WebSphere Application Server Liberty (stopappserver and startappserver) and run JnextPlan to make this change effective.

If you use a security firewall, make sure this port is open for incoming and outgoing connections.

extRecPrompt | xp

Additional prompts after abend. Specify an additional number of prompts for the value defined in *baseRecPropmt*. This applies when a job is rerun after abending and the limit specified in *baseRecPropmt* has been reached.

The default value is *1000*. Run JnextPlan to make this change effective.

fileStartConditionJobName | fc

Name of the job in charge of running the file monitoring task . Applicable only if you select file as the start condition type. Specify the name of the job which is automatically added to the plan to run the file monitoring task. This value is used by default if you do not specify any value for the job name when defining the start condition. If you specify a value for the job name, this value is ignored.

The default value is FILE_STARTCOND. The maximum supported length is 40 bytes. Changes to this parameter are effective immediately.

folderDays | fd

Remove deleted folders, prompts, resources, and workstations from the database. When deleting a folder, a prompt, or resource, if there are still objects in the plan that reference these objects, then another folder, prompt, or resource cannot be renamed with the name of the deleted folder, prompt or resource for the number of days specified by "folderDays?. However, a brand new folder, prompt, or resource can be created with the name of the deleted object.

When deleting a workstation, if the workstation is still in the plan, then another workstation cannot be renamed with the name of the deleted workstation for the number of days specified by the global option folderDays. However, a brand new workstation can be created with the name of the deleted workstation. This behavior applies only to dynamic agents, pools, and dynamic pools.

The default value is 10 days.

ignoreCals | ic

Ignore calendars. A preproduction option that affects the operation of the planman command. Its setting determines if user calendars are copied into the new production plan (Symphony) file. To prevent user calendars from being copied into the new production plan, enter *yes*.

The default value is *no*. See *HCL Workload Automation: User's Guide and Reference*. Run JnextPlan to make this change effective.

Note: Starting from version 10.1, this option is deprecated and must not be modified. By default, its value is set to **yes**.

licenseType | ln

Type of accepted license for HCL Workload Automation.

Supported values are:

ws

perServer

This value is ignored in HCL Workload Automation.

wa

perJob

to specify the HCL Workload Automation Per Job (PJ) pricing.

byWorkstation

This value is ignored in HCL Workload Automation.

The default value is **perJob**. Run JnextPlan to make this change effective. For additional information about license management and metrics, see Per Job license model on page 471.

You can define this option for the following workstation types:

- master domain manager
- fault-tolerant agent
- standard agent
- dynamic agent

licenseProxyPassword | pw

License Proxy Password The password of the proxy server which HCL Workload Automation is expected to contact. This option is required if you are using a proxy server protected by a user name and password.

The default value is null because it must be specified by the user. This option takes effect immediately.

licenseProxyServer | lp

IP of the proxy server The IP of the proxy server which HCL Workload Automation is expected to contact. This option is required if you are using a proxy server.

The default value is null because it must be specified by the user. This option takes effect immediately.

licenseProxyServerPort | lo

Port of the proxy server The port of the proxy server the master domain manager uses to connect to the Internet. This option is required if you are using a proxy server.

The default value is null because it must be specified by the user. This option takes effect immediately.

licenseProxyUser | pb

User of the proxy server The user of the proxy server the master domain manager uses to connect to the Internet. This option is required if you are using a proxy server protected by a user name and password.

The default value is null because it must be specified by the user. This option takes effect immediately.

licenseRefreshToken | rt

Token of the license server The Deployment Key you obtain from the license server after creating the deployment. For more information, see High-level procedurethe topic about enabling product license management in *HCL Workload Automation: Planning and Installation*.

The default value is null because it must be specified by the user. This option takes effect immediately.

licenseServerUrl | lu

License server URL The URL of the license server which processes license usage information. The URL value is https://api.hcltechsw.com/. When you buy HCL Workload Automation, license usage is calculated based on the number of successful jobs you run and the related information is processed on the server you specify in this option.

This option takes effect immediately.

logCleanupFrequency | lc

Log cleanup frequency. Used in event rule and audit management . Specify how often the automatic cleanup of log instances is run. Valid values are in the *0-60* minutes range. If you specify *0*, the automatic cleanup feature is disabled.

The default value is *5* minutes. This option takes effect immediately.

logHistory | lh

Log history period. Used in event rule management. Enter the number of days for which you want to save rule instance, action run, and message log data. Log instances are discarded on a FIFO (first-in first-out) basis.

The default value is *10* days. This option takes effect immediately.

logmanMinMaxPolicy | lm

Logman minimum and maximum run times policy. Specify how the minimum and maximum job run times are logged and reported by logman. Possible values are:

elapsedtime

The minimum and maximum elapsed runtimes are logged and reported.

cputime

The minimum and maximum CPU run times are logged and reported.

both

Both the minimum and maximum job runtimes are logged and reported.

See *HCL Workload Automation: User's Guide and Reference* for details on using this feature.

The default value is *both*. Run JnextPlan to make this change effective.

logmanSmoothPolicy | lt

Logman normal run time calculation policy. Set the weighting factor that favors the most recent job run when calculating the normal (average) run time for a job. This is expressed as a percentage. For example, specify *40* to apply a weighting factor of 40% to the most recent job run, and 60% to the existing average. See the topic about customizing plan management using global options in *HCL Workload Automation: User's Guide and Reference* for more information about how to use this option.

The default value is *-1*. Run JnextPlan to make this change effective.

longDurationThreshold | ld

Long duration job threshold. Specify, when comparing the actual duration of a job to the estimated duration, the threshold over which the job is considered to be of "long duration." The threshold value is expressed as a percentage with respect to the estimated duration. For example, if the threshold is set to *150*, and the actual duration is more than 150% of the estimated duration (it is 50% greater), the job is considered to be a "long duration" job.

If you have the workload service assurance feature enabled, the effect of a "critical" job satisfying the long duration criteria is that the job is inserted automatically into the hot list.

Valid values are between:

100

The minimum value. All jobs that exceed the estimated duration are considered long duration jobs

1000

The maximum value. Only those jobs that last ten times as long as their estimated duration are considered as long duration jobs

The default is *150*.

Note: Whatever value you set for this option, if you have the workload service assurance feature enabled, and HCL Workload Automation loses the connection with its database, the default value is applied to critical job processing, and the warning message AWSJCO135W is issued to tell you what has happened.

Run JnextPlan or restart WebSphere Application Server Liberty (stopappserver and startappserver) to make this change effective.

mailSenderName | ms

Mail sender name. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify a string to be used as the sender of the emails.

The default value is *TWS*. Changes to this parameter are effective for the next mail send action performed.

maxLen | xl

Maximum preproduction plan length. Specify the maximum length of the preproduction plan in days after it is automatically extended or created. The value for *maxLen* must be greater than or equal to the value for *minLen* and must be in the range of *8* to *365*.

The default is *14* days. Run JnextPlan to make this change effective.

minLen | ml

Minimum preproduction plan length. Specify the minimum length in days of the preproduction plan that can pass after the production plan is created or extended, without extending the preproduction plan. If the days left in the preproduction plan after a JnextPlan are less than the value of this option, the preproduction plan is automatically extended. The value for *minLen* must be less than or equal to the value for *maxLen* and must be in the range of *7* to *365*.

The default is *8* days. Run JnextPlan to make this change effective.

notificationTimeout | nt

Notification timeout. Used in cross dependencies. Specify how many days HCL Workload Automation must retry sending notifications about job status changes to the remote engine if the notification fails. When this timeout expires, the job request subscription and the status notifications associated to this job are discarded.

Valid values are in the range of *1* to *90*. The default is *5* days. Changes to this parameter are effective immediately.

promotionOffset | po

Promotion offset. Used in workload service assurance. Specify when a job become eligible for promotion in terms of the number of seconds before its critical start time is reached. Applies only to jobs that are flagged as critical in a job stream definition and to their predecessor jobs. A critical job and its predecessors make up a critical network.

When a predecessor jeopardizes the timely completion of the critical job, it is *promoted*; that is, it is assigned additional resources and its submission is prioritized with respect to other jobs that are out of the critical network. Also critical jobs might be promoted.

The scheduler calculates the critical start time of a critical job by subtracting its estimated duration from its deadline. It calculates the critical start time of a critical predecessor by subtracting its estimated duration from the critical start time of its next successor. Within a critical network the scheduler calculates the critical start time of the critical job first and then works backwards along the chain of predecessors. These calculations are reiterated as many times as necessary until the critical job has run.

This option is only active if *enWorkloadServiceAssurance* is set to *yes*.

The default is *120* seconds.

Run JnextPlan to make this change effective.

resubmitJobName | rj

Name of the job in charge of resubmitting the job stream. Specify the name of the Job Stream Submission job which is automatically added to the plan to resubmit a new instance of the job stream where the start condition is defined.

The default value is *MASTERAGENTS#restart_StartCond*, where MASTERAGENTS is the name of the pool workstation on which the Job Stream Submission job runs. The maximum length for the workstation name is 16 bytes, and the maximum length for the job name is 40 bytes. Changes to this parameter are effective immediately.

resubmitJobUserName | rw

Name of the user in charge of resubmitting the job stream. Specify the user name which owns the Job Stream Submission job. The Job Stream Submission job is automatically added to the plan to resubmit a new instance of the job stream where the start condition is defined.

The default value is *TWS_User*. Changes to this parameter are effective immediately. If the user defined in the **resubmitJobUserName** property does not exist, the user name and password defined on WebSphere Application Server Liberty installed on the master domain manager or backup master domain manager are used. This implies that the user defined in the **resubmitJobUserName** property must be the same both on the master domain manager and on the backup master domain manager, or must be changed immediately after switching the master domain manager.

riskConfidence | rc

Critical Jobs Risk Confidence Specifies when a critical job must be set as **High Risk**, comparing the confidence factor of completing before deadline and the percentage specified in this parameter. If the probability of completing before the deadline is below **riskConfidence,** then the critical job is considered at high risk. Valid values are in the range `1-99`. The default value is `80%` when you perform a fresh installation. If you upgrade a previous version to the current version, the default value is `50%` for maintaining backward compatibility. This option is effective immediately.

servicenowUrl | nu

ServiceNow URL. Used in event rule management. If you use rules that implement an action that opens an incident in ServiceNow (or any other application that can open an incident in the ServiceNow format), specify the ServiceNow URL. You can change this value when you define the action if you want to use a different ServiceNow URL.

The default value is "`http://localhost:8080/api/now/v1/table/incident`". Changes to this parameter are effective immediately.

servicenowUserName | nn

ServiceNow user name. Used in event rule management. If you use rules that implement an action that opens an incident in ServiceNow, specify the identifier of the user connecting to the ServiceNow server.

The default value is the HCL Workload Automation user on the master domain manager. Changes to this parameter are effective immediately.

servicenowUserPassword| np

ServiceNow user password. Used in event rule management. If you use rules that implement an action that opens an incident in ServiceNow, specify the password associated with the user connecting to the ServiceNow server.

Changes to this parameter are effective immediately.

smtpServerName | sn

SMTP server name. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify the name of the SMTP server to be used by the mail plug-in.

The default value is *localhost*. Changes to this parameter are effective immediately.

smtpServerPort | sp

SMTP Server port. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify the port number used to connect to the SMTP server by the mail plug-in. Valid values are in the range *0–65535*.

The default value is *25*. Changes to this parameter are effective for the next mail send action performed.

smtpUseAuthentication | ua

Mail plug-in uses SMTP authentication. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify if the SMTP connection needs to be authenticated. Values are *yes* or *no*.

The default is *no*. Changes to this parameter are effective immediately.

smtpUserName | un

SMTP server user name. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify the SMTP server user name.

The default value is the name of the HCL Workload Automation user (the TWS_user) on the master domain manager. Changes to this parameter are effective immediately.

smtpUserPassword | up

SMTP server user password. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify the SMTP server user password. The password is stored in an encrypted form.

Changes to this parameter are effective immediately.

smtpUseSSL | us

Mail plug-in uses SSL. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify if the SMTP connection is to be authenticated via SSL. Values are *yes* or *no*.

The default is *no*. Changes to this parameter are effective immediately.

smtpUseTLS | tl

Mail plug-in uses TLS protocol. Used in event rule management. If you deploy rules implementing an action that sends emails via an SMTP server, specify if the SMTP connection is to be authenticated via the Transport Layer Security (TLS) protocol. Values are *yes* or *no*.

The default is *no*. Changes to this parameter are effective immediately.

startOfDay | sd

Start time of processing day. Specify the start time of the HCL Workload Automation processing day in 24-hour format: *hhmm* (*0000-2359*).

The default value is *0000* (0:00 a.m.), but if you upgraded your environment to version 10.2.5 starting from a version earlier than version 8.6, the default value is *0600* (6:00 a.m.). If you change this option, you must also change the launch time of the *final* Job Scheduler, which is usually set to one minute before the start time. Run JnextPlan to make the change of *startOfDay* effective.

If you need to modify in a production environment the start of day and the FINAL job stream launch time, run the following command twice:

```
JnextPlan -to newDay newSoD TZ customerTimezone
```

For example, if your timezone is America/Chicago and on the 30th of August you want to modify the start of day from `0400` to `0700`, perform the following steps:

1. Verify the current start of day using the optman ls command:

   ```
   optman ls
   startOfDay / sd=0400
   ```

2. Modify the **startOfDay** global option using optman:

   ```
   optman chg sd=0700
   ```

3. Validate all the job stream definitions to make sure they are compatible with the new **startOfDay** time running the composer command:

   ```
   composer validate <extracted schedule definition>
   ```

4. Modify the launch time of the *FINAL* and *FINALPOSTREPORTS* job streams modifying the **at** keyword from `0359` to `0659`.
5. Run the command `planman showinfo` and look for the production plan end time, for example:

   ```
   production plan end time: 06/22/2024 03:59 tz CST6CDT
   ```

6. Run the following command twice:

   ```
   JnextPlan -to 06/22/2024 0700 tz America/Chicago
   ```

   Note: The `-to` value must be later than the production plan end time

7. Cancel the previous *FINAL* and *FINALPOSTREPORTS* job streams. This final step prevents the production plan from extending an additional 24 hours.

statsHistory | sh

Job statistics history period. Specify the number of days for which you want to maintain job statistics. Statistics are discarded on a FIFO (first-in first-out) basis. For example, if you leave the default value of *400*, statistics are maintained for the last 400 days. This has no effect on job standard list files, which must be removed with the *rmstdlist* command. See the *HCL Workload Automation: User's Guide and Reference* for information about the *rmstdlist* command.

The default value is *400*. Run JnextPlan to make this change effective in the plan. For the database, this option takes effect immediately.

startConditionDeadlineOffset | cd

Start condition deadline offset. The default offset set for the start condition deadline in 24 hour format: "hhmm" (0001-9959). Specify the time range during which the start condition is active.

The default value is *2400* and the range is *0001 - 9959*. Changes to this parameter are effective immediately.

untilDays | ud

Remove obsolete job and job stream instances from the plan. If an **until** time (latest start time) has not been specified for a job or job stream, then the default **until** time is calculated adding the value of this option, expressed in number of days, to the scheduled time of the job or job stream. If the *enCarryForward* option is set to **all**, and the number of days specified for *untilDays* is reached, then any job or job stream instance in the plan that ended in error is automatically removed from the plan and not added to the new production plan.

The default value is **0**. If the default value is used, then for jobs, no default time is set for the **until** time (latest start time) . For job streams, if the default is used, then the default until time is 2 days.

Run JnextPlan to make this change effective.

workstationLimit | wl

The workstation limit.

Used in the automatic dynamic agent registration. This parameter specifies the dynamic agent workstation limit value that the dynamic agent workstation assumes after the workstation is added to the plan. You can later modify the dynamic agent workstation limit value by using the conman command line or the Dynamic Workload Console.

Valid values are in the *0-1024* range.

The default is *100*. Changes to this parameter are effective immediately.

zOSRemoteServerName | zr

HCL Workload Automation for Z connector remote server name. Used in event rule management. If you deploy rules implementing an action that submits job streams to the HCL Workload Automation for Z controller, enter the name of the controller specified as the engine to the Z connector. It must exactly match the Z connector engine name and is case sensitive.

After changing the value of this parameter, the change becomes effective when the next `submit` action is run.

zOSServerName | zs

HCL Workload Automation for Z connector server name. Used in event rule management. If you deploy rules implementing an action that submits job streams to the HCL Workload Automation for Z controller, specify the name or the hostname of the system where the HCL Workload Automation for Z connector runs. The default value is `localhost`.

After changing the value of this parameter, the change becomes effective when the next `submit` action is run.

zOSServerPort | zp

HCL Workload Automation for Z connector server port. Used in event rule management. If you deploy rules implementing an action that submits job streams to the HCL Workload Automation for Z controller, specify the bootstrap port number of the HCL Workload Automation for Z connector server. Valid values are in the range 0-65535. The default value is 31217.

After changing the value of this parameter, the change becomes effective when the next `submit` action is run.

zOSUserName | zu

HCL Workload Automation for Z connector user name. Used in event rule management. If you deploy rules implementing an action that submits job streams to the HCL Workload Automation for Z controller, specify the HCL Workload Automation for Z connector user name required to access the HCL Workload Automation for Z engine.

After changing the value of this parameter, the change becomes effective when the next `submit` action is run.

zOSUserPassword | zw

HCL Workload Automation for Z connector user password. Used in event rule management. If you deploy rules implementing an action that submits job streams to the HCL Workload Automation for Z controller, specify the HCL Workload Automation for Z connector user password required to access the HCL Workload Automation for Z engine. The password is stored in encrypted form.

After changing the value of this parameter, the change becomes effective when the next `submit` action is run.

Setting local options

Set local options, such as general attributes of the workstation for the HCL Workload Automation processes, in the `localopts` file. Changes do not take effect until netman is stopped (**conman shut;wait**) and restarted (**StartUp**).

During the installation process, a working copy of the local options file is installed as *TWA_DATA_DIR*`/localopts`.

The `localopts` file is not modified during the agent upgrade process. The file generated by the upgrade process is saved to the `/config` directory, to maintain your custom values, if any. You can then merge the two files with your customized values and save the resulting file in the *TWA_DATA_DIR* folder.

A template file containing default settings is located in *TWA_DATA_DIR*.

Note: All of the SSL settings in the localopts file relate to the network communications and do not relate to the Dynamic Workload Console.

The options in the `localopts` file are described in the following sections:

- Localopts summary on page 49
- Localopts details on page 52
- Local options file example on page 66

Localopts summary

General attributes of the workstation:

thiscpu = *workstation*

merge stdlists = *yes|no*

stdlist width = *columns*

syslog local = *facility*

restricted stdlists = *yes|no*

The attributes of the workstation for the batchman process:

bm check file = *seconds*

bm check status = *seconds*

bm look = *seconds*

bm read = *seconds*

bm stats = *on|off*

bm verbose = *on|off*

bm check until = *seconds*

bm check deadline = *seconds*

bm late every = *minutes*

The attributes of the workstation for the jobman process:

jm job table size = *entries*

jm look = *seconds*

jm nice = *value*

jm promoted nice = *UNIX® and Linux® critical job priority*

jm promoted priority = *Windows® critical job priority*

jm no root = *yes|no*

jm file no root = *yes|no*

jm read = *seconds*

jm loaduserprofile = *on|off*

The attributes of the workstation for the mailman process:

mm response = *seconds*

mm retrylink = *seconds*

mm sound off = *yes|no*

mm unlink = *seconds*

mm cache mailbox = *yes|no*

mm cache size = *bytes*

mm resolve master = *yes*|*no*

autostart monman = *yes*|*no*

The attributes of the workstation for the netman process:

nm mortal = *yes*|*no*

nm port = *port number*

nm read = *seconds*

nm retry = *seconds*

The attributes of the workstation for the writer process:

wr read = *seconds*

wr unlink = *seconds*

wr enable compression = *yes*|*no*

Optional attributes of the workstation for remote database files

mozart directory = *mozart_share*

parameters directory = *parms_share*

unison network directory = *unison_share*

The attributes of the workstation for the custom formats

date format = *integer*

composer prompt = *key*

conman prompt = *key*

switch sym prompt = *key*

The attributes of the workstation for the customization of I/O on mailbox files

sync level = *low*|*medium*|*high*

The attributes of the workstation for networking

tcp timeout = *seconds*

tcp connect timeout = *seconds*

The attributes of the workstation for SSL - General

ssl auth mode = *caonly*|*string*|*cpu*

ssl auth string = *string*

SSL FIPS compliance=*yes*|*no*

SSL config file=*SSL_configuration_file*

nm ssl full port = *value*

nm ssl port = *value*

OpenSSL attributes of the workstation

SSL key = **.pem*

SSL certificate = **.pem*

SSL key pwd = **.sth*

SSL CA certificate = **.crt*

SSL random seed = **.rnd*

SSL cert type = *certificate_type*

SSL ciphers = *SSL_ciphers*

SSL cipher suites = *encryption_algorithms*

SSL version = *ssl_version*

CLI SSL server auth = *yes|no*

CLI SSL ciphers = *ssl_ciphers*

CLI SSL cipher suites = *encryption_algorithms*

CLI SSL version=

CLI SSL server certificate = *file_name*

CLI SSL trusted dir = *directory_name*

Encryption options

encrypt keystore file = *keystore_file*

encrypt keystore pwd = *keystore_password*

encrypt label = *default*

decrypt label list =

The HCL Workload Automation instance is a command line client

is remote cli = *yes|no*

Attributes for CLI connections

host = *host_name*

protocol = *protocol*

port = *port number*

proxy = *proxy server*

proxy port = *proxy server port number*

timeout = *seconds*

defaultws = *master_workstation*

useropts = *useropts_file*

Note:

1. The SSL attributes for the command line client connection will depend on which SSL method is in use. They are included in the relevant section and all commence with "cli".
2. The command lines for the dynamic domain manager and backup dynamic domain manager will work only if you configure the **host** and **port** attributes.

Event Management parameters

can be event processor = *yes|no*

Centralized Agent Update parameters

DownloadDir = *directory_name*

Current Folder

current folder = */foldername>*

The attributes of the workstation for the Appserver Watchdog (WebSphere Application Server Liberty)

Application server check attributes on the workstation

appserver check interval = *minutes*
appserver auto restart = *on|off*
appserver min restart time = *minutes*
appserver max restarts = *number*
appserver count reset interval = *hours*

Note: The `localopts` file syntax is not case-sensitive, and the spaces between words in the option names are ignored. For example, you can validly write **is remote cli** as:

- is remote cli
- Is Remote CLI
- isremotecli
- ISREMOTECLI
- isRemoteCLI
- ...HCL

Localopts details

# comment

Treats everything from the indicated character (#) to the end of the line as a comment.

appserver auto restart = yes|no

Requests the `appservman` process to automatically start WebSphere Application Server Liberty if it is found down. The default is `Yes`.

appserver check interval = *minutes*

Specifies the frequency in minutes that the `appservman` process is to check that WebSphere Application Server Liberty is still running. The default is 3 minutes.

appserver count reset interval = *hours*

Specifies the time interval in hours after which the restart count is reset from the last WebSphere Application Server Liberty start. The default is 24 hours.

appserver max restarts = *number*

Specifies the maximum number of restarting attempts the `appservman` process can make before giving up and exiting without restarting WebSphere Application Server Liberty. The counter is reset if WebSphere Application Server Liberty runs for longer than the `appserver count reset interval` value. The default is 5.

appserver min restart time = *minutes*

Specifies in minutes the minimum elapsed time the `appservman` process must wait between each attempt to restart the WebSphere Application Server Liberty if it is down. If this value is less than the `appserver check interval`, the WebSphere Application Server Liberty is restarted as soon as it is found down. If it is found down before this time interval (min restart time) has elapsed, `appservman` exits without restarting it. The default is 2 minutes.

autostart monman = yes|no

Used in event rule management. Restarts the monitoring engine automatically when the next production plan is activated (on Windows® also when HCL Workload Automation is restarted). The default is `Yes`.

bm check deadline = *seconds*

Specify the minimum number of seconds Batchman waits before checking if a job has missed its deadline. The check is performed on all jobs and job streams included in the Symphony file, regardless of the workstation where the jobs and job streams are defined. Jobs and job streams with expired deadlines are marked as late in the local Symphony file. To obtain up-to-date information about the whole environment, define this option on the master domain manager. Deadlines for critical jobs are evaluated automatically, independently of the **bm check deadline** option. To disable the option and not check deadlines, enter a value of zero, the default value.

bm check file = *seconds*

Specify the minimum number of seconds Batchman waits before checking for the existence of a file that is used as a dependency. The default is 120 seconds.

bm check status = *seconds*

Specify the number of seconds Batchman waits between checking the status of an internetwork dependency. The default is 300 seconds.

bm check until = *seconds*

Specify the maximum number of seconds Batchman waits before reporting the expiration of an Until time for job or Job Scheduler. Specifying a value below the default setting (300) might overload the system. If it is set below the value of Local Option **bm read**, the value of **bm read** is used in its place. The default is 300 seconds.

bm late every = minutes

When an **every** job does not start at its expected start time, **bm late every** specifies the maximum number of minutes that elapse before HCL Workload Automation skips the job. This option applies only to jobs defined with **every** option together with the **at** time dependency, it has no impact on jobs that have only the **every** option.

bm look = ***seconds***

Specify the minimum number of seconds Batchman waits before scanning and updating its production control file. If you install the 9.4, FP1 version as a fresh installation, the default value is automatically set to 5 for improving product performance. The previous default value was 15 seconds and is maintained if you perform a product upgrade.

bm read = ***seconds***

Specify the maximum number of seconds Batchman waits for a message in the `intercom.msg` message file. If no messages are in queue, Batchman waits until the timeout expires or until a message is written to the file. If you install the 9.4, FP1 version as a fresh installation, the default value is automatically set to 3 for improving product performance. The previous default value was 10 seconds and is maintained if you perform a product upgrade.

bm stats = on|off

To have Batchman send its startup and shut down statistics to its standard list file, specify **on**. To prevent Batchman statistics from being sent to its standard list file, specify **off**. The default is **off**.

bm verbose = on|off

To have Batchman send all job status messages to its standard list file, specify **on**. To prevent the extended set of job status messages from being sent to the standard list file, specify **off**. The default is **off**.

can be event processor = yes|no

Specify if this workstation can act as event processing server or not. It is set by default to **yes** for master domain managers and backup masters, otherwise it is set to **no**.

CLI SSL ciphers = ***cipher_class***

Specify the cipher class to be used when the command-line client and the server are using SSL authentication. The default is MD5.

If you want to use an OpenSSL cipher class, use the following command to determine the list of available classes:

```
openssl ciphers
```

For a full list of supported ciphers, see SSL Ciphers and OpenSSL.

CLI SSL cipher suites ***suites***

Specify one or more supported algorithms for TLS version 1.3, This option does not apply to TLS version 1.2 or earlier.

CLI SSL server auth = yes|no

Specify **yes** if server authentication is to be used in SSL communications with the command line client. The default is **no**.

CLI SSL server certificate = *file_name*

Specify the file, including its full directory path, that contains the SSL certificate when the command-line client and the server use SSL authentication in their communication. There is no default. See Configuring SSL attributes on page 328.

CLI SSL trusted dir = *directory_name*

Specify the directory that contains an SSL trusted certificate contained in files with hash naming (#) when the command-line client and the server are using SSL authentication in their communication. When the directory path contains blanks, enclose it in double quotation marks ("). There is no default.

CLI SSL version *version*

Specify the SSL version to be used. Supported values are:

- **atleast.TLSv1.0**
- **atleast.TLSv1.1**
- **atleast.TLSv1.2**
- **atleast.TLSv1.3**

where you specify the minimum version of the TLS protocol to be used. In this case, HCL Workload Automation uses the specified version of the protocol or a higher version, if supported.

- **max.TLSv1.0**
- **max.TLSv1.1**
- **max.TLSv1.2**
- **max.TLSv1.3**

where you specify the maximum version of the TLS protocol to be used. In this case, HCL Workload Automation uses the specified version of the protocol or a lower version.

- **TLSv1.0**
- **TLSv1.1**
- **TLSv1.2**
- **TLSv1.3**

where you specify the exact version of the TLS protocol to be used. In this case, HCL Workload Automation uses the specified version of the protocol.

composer prompt = *prompt*

Specify the prompt for the composer command line. The prompt can be of up to 10 characters in length. The default is dash (**-**).

conman prompt = *prompt*

Specify the prompt for the conman command line. The prompt can be of up to 8 characters in length. The default is percent (**%**).

current folder = */foldername*>

When submitting commands that involve folders from either the composer or conman command line, you can change the default folder or working directory from the root (/) to another folder path so that you can submit commands from the composer or conman command line using relative folder paths.

date format = 0|1|2|3

Specify the value that corresponds to the date format you require. The values can be:

- 0 corresponds to *yy/mm/dd*
- 1 corresponds to *mm/dd/yy*
- 2 corresponds to *dd/mm/yy*
- 3 indicates usage of Native Language Support variables

The default is **1**.

decrypt label list

The list of previously used aliases for key encryption. When you modify a key alias for key rotation, store the previous alias in this property. This storage method is useful if the obsolete key is still used in the product. Separate each value with a comma ",". Note that this property is commented. This property is case insensitive. For more information about encryption and key rotation, see Automatic encryption for key product files on page 133.

defaultws = *manager_workstation*

The default workstation when you are accessing using a command line client. Specify the domain manager workstation.

jm file no root = yes|no

For UNIX® and Linux® operating systems only, specify **yes** to prevent Jobman from executing commands in file dependencies as **root**. Specify **no** to allow Jobman to execute commands in file dependencies as **root**. The default is **no**.

DownloadDir = *directory_name*

Defines the name of the directory where the fix pack installation package or upgrade eImage is downloaded during the centralized agent update process. If not specified, the following default directory is used:

On Windows operating systems:

```
TWA_home\TWS\stdlist\JM\download
```

On UNIX operating systems:

```
TWA_home/TWS/stdlist/JM/download
```

encrypt keystore file *file_name*

The path to the keystore PKCS12 file, containing the AES-256 or AES-128 key. The keystore is created automatically at installation time and the related path is inserted in this parameter. If you want to use a different keystore, you can create it and add the path in this option.

encrypt keystore pwd *password*

The path to the keystore stash file.

encrypt label *label*

The label of the key in the keystore. When you modify a key label for key rotation, store the previous label in the **decrypt label list** property, so it can be retrieved if it is still used in the product. This property is case insensitive. For more information about encryption and key rotation, see Automatic encryption for key product files on page 133.

host = *hostname_or_IP_address*

The name or IP address of the host when accessing using a command line client. For **Agents**, the host or ip address of the master is used. For **Backup Master Domain Manager** the value is the default: 127.0.0.1

is remote cli = yes|no

Specify if this instance of HCL Workload Automation is installed as a command line client (yes).

jm job table size = *entries*

Specify the size, in number of entries, of the job table used by Jobman. The default is 1024 entries.

jm loaduserprofile = on|off

Only on Windows operating systems. Specify if the jobman process loads the user profile and its environment variables for the user specified in the logon field of each job, before starting the job on the workstation. Specify **on** to load the user profile on the workstation before running jobs for the logon user; otherwise specify **off**. Roaming profiles are not supported. The default is **on**.

jm look = *seconds*

Specify the minimum number of seconds Jobman waits before looking for completed jobs and performing general job management tasks. The default is 300 seconds.

jm nice = *nice_value*

For UNIX® and Linux® operating systems only, specify the **nice** value to be applied to jobs launched by Jobman to change their priority in the kernel's scheduler. The default is zero.

The **nice** boundary values vary depending upon each specific platform, but generally lower values correspond to higher priority levels and vice versa. The default depends upon the operating system.

Applies to jobs scheduled by the root user only. Jobs submitted by any other user inherit the same **nice** value of the Jobman process.

See also jm promoted nice on page 58.

jm no root = yes|no

For UNIX® and Linux® operating systems only, specify **yes** to prevent Jobman from launching **root** jobs. Specify **no** to allow Jobman to launch **root** jobs. The default is **yes**.

jm promoted nice = *nice_value*

Used in workload service assurance. For UNIX® and Linux® operating systems only, assigns the priority value to a critical job that needs to be promoted so that the operating system processes it before others. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time.

Boundary values vary depending upon each specific platform, but generally lower values correspond to higher priority levels and vice versa. The default is -1.

Be aware that:

- The promotion process is effective with negative values only. If you set a positive value, the system runs it with the -1 default value and logs a warning message every time Jobman starts.
- An out of range value (for example -200), prompts the operating system to automatically promote the jobs with the lowest allowed **nice** value. Note that in this case no warning is logged.
- Overusing the promotion mechanism (that is, defining an exceedingly high number of jobs as mission critical and setting the highest priority value here) might overload the operating system, negatively impacting the overall performance of the workstation.

You can use this and the jm nice on page 57 options together. If you do, remember that, while **jm nice** applies only to jobs submitted by the root user, **jm promoted nice** applies only to jobs that have a critical start time. When a job matches both conditions, the values set for the two options add up. For example, if on a particular agent the local options file has:

```
jm nice= -2
jm promoted nice= -4
```

when a critical job submitted by the root user needs to be promoted, it is assigned a cumulative priority value of -6.

jm promoted priority = *value*

Used in workload service assurance. For Windows® operating systems only, sets to this value the priority by which the operating system processes a critical job when it is promoted.

Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time.

The possible values are:

- `High`
- `AboveNormal` (the default)
- `Normal`

- `BelowNormal`
- `Low` or `Idle`

Note that if you a set a lower priority value than the one non-critical jobs might be assigned, no warning is given and no mechanism like the one available for **jm promoted nice** sets it back to the default.

jm read = *seconds*

Specify the maximum number of seconds Jobman waits for a message in the `courier.msg` message file. The default is 10 seconds.

merge stdlists = yes|no

Specify **yes** to have all of the HCL Workload Automation control processes, except Netman, send their console messages to a single standard list file. The file is given the name **TWSmerge**. Specify **no** to have the processes send messages to separate standard list files. The default is **yes**.

mm cache mailbox = yes|no

Use this option to enable Mailman to use a reading cache for incoming messages. In this case, only messages considered essential for network consistency are cached. The default is **yes**.

mm cache size = *messages*

Specify this option if you also use **mm cache mailbox**. The maximum value (default) is **512**.

mm resolve master = yes|no

When set to **yes** the $MASTER variable is resolved at the beginning of the production day. The host of any extended agent is switched after the next JnextPlan (long term switch). When it is set to **no**, the $MASTER variable is not resolved at JnextPlan and the host of any extended agent can be switched after a conman **switchmgr** command (short- and long-term switch). Starting from Version 9.5 Fix Pack 2, the default is **no** (for previous releases, it was set to **yes**. When you switch a master domain manager and the original has mm resolve master set to **no** and the backup has mm resolve master set to **yes**, after the switch any extended agent that is hosted by $MASTER switches to the backup master domain manager. When the backup master domain manager restarts, the keyword $MASTER is locally expanded by Mailman. You should keep the mm resolve master value the same for master domain managers and backup domain managers.

mm response = *seconds*

Specify the maximum number of seconds Mailman waits for a response before reporting that a workstation is not responding. The minimum wait time for a response is **90** seconds. The default is 600 seconds.

mm retrylink = *seconds*

Specify the maximum number of seconds Mailman waits after unlinking from a non-responding workstation before it attempts to link to the workstation again. The default is 600 seconds. The **tomserver** optional mailman servers do not unlink non-responding agents. The link is repetitively checked every 60 seconds, which is the default **retrylink** for these servers.

mm sound off = yes|no

Specify how Mailman responds to a conman **tellop ?** command. Specify **yes** to have Mailman display information about every task it is performing. Specify **no** to have Mailman send only its own status. The default is **no**.

mm symphony download timeout = *seconds*

Specify the maximum number of minutes Mailman waits after attempting to initialize a workstation on a slow network. If the timeout expires without the workstation being initialized successfully, Mailman initializes the next workstation in the sequence. The default is no timeout (0).

mm unlink = *seconds*

Specify the maximum number of seconds Mailman waits before unlinking from a workstation that is not responding. The wait time should not be less than the response time specified for the Local Option **nm response**. The default is 960 seconds.

nm mortal = yes|no

Specify **yes** to have Netman quit when all of its child processes have stopped. Specify **no** to have Netman keep running even after its child processes have stopped. The default is **no**.

nm port = *port*

Specify the TCP port number that Netman responds to on the local computer. This must match the TCP/IP port in the computers workstation definition. It must be an unsigned 16-bit value in the range 1- 65535 (values between 0 and 1023 are reserved for services such as, FTP, TELNET, HTTP, and so on). The default is the value supplied during the product installation. If you disable this port at installation time using the port parameter, this option is automatically set to 0.

If you run event-driven workload automation and you have a security firewall, make sure this port is open for incoming and outgoing connections.

nm read = *seconds*

Specify the maximum number of seconds Netman waits for a connection request before checking its message queue for **stop** and **start** commands. The default is 10 seconds.

nm retry = *seconds*

Specify the maximum number of seconds Netman waits before retrying a connection that failed. The default is 800 seconds.

nm SSL full port = *port*

The port used to listen for incoming SSL connections when full SSL is configured by setting global option `enSSLFullConnection` to `yes` (see Configuring full SSL security on page 330 for more details). This value must match the one defined in the **secureaddr** attribute in the workstation definition in the database. It must be different from the **nm port** local option that defines the port used for normal communication.

Note:

1. If you install multiple instances of HCL Workload Automation on the same computer, set all SSL ports to different values.
2. If you plan not to use SSL, set the value to 0.

The default value is 31113.

nm SSL port = ***port***

The port used to listen for incoming SSL connections, when full SSL is not configured (see Configuring full SSL security on page 330 for more details). This value must match the one defined in the **secureaddr** attribute in the workstation definition in the database. It must be different from the **nm port** local option that defines the port used for normal communication.

Note:

1. If you install multiple instances of HCL Workload Automation on the same computer, set all SSL ports to different values.
2. If you plan not to use SSL, set the value to 0.

The default value is 0.

port = ***port***

The TCP/IP port number of the protocol used when accessing using a command line client. The default is 31116.

protocol = http|https

The protocol used to connect to the host when accessing using a command line client.

proxy = ***proxy_server_hostname_or_IP_address***

The name of the proxy server used when accessing using a command line client.

proxy port = ***proxy_server_port***

The TCP/IP port number of the proxy server used when accessing using a command line client.

restricted stdlists = yes|no

Use this option to set a higher degree of security to the `stdlist` directory (and to its subdirectories) allowing only selected users to create, modify, or read files.

This option is available for UNIX workstations only. After you define it, make sure you erase your current `stdlist` directory (and subdirectories) and that you restart HCL Workload Automation. The default is `no`.

If the option is not present or if it is set to `no`, the newly created `stdlist` directory and its subdirectories are unaffected and their rights are as follows:

```
drwxrwxr-x  22 twsmdm staff          4096 Nov 09 12:12
drwxrwxr-x   2 twsmdm staff           256 Nov 09 11:40 2009.11.09
```

```
drwxrwxr-x   2 twsmdm staff          4096 Nov 09 11:40 logs
drwxr-xr-x   2 twsmdm staff          4096 Nov 09 11:40 traces
```

If the option is set to `yes`, these directories have the following rights:

```
drwxr-x--x  5 twsmdm staff             256 Nov 13 18:15
rwxr-x--x   2 twsmdm staff             256 Nov 13 18:15 2009.11.13
rwxr-x--x   2 twsmdm staff             256 Nov 13 18:15 logs
rwxr-x--x   2 twsmdm staff             256 Nov 13 18:15 traces
```

Do the following to define and activate this option:

1. Change the line `restricted stdlists = no` to `restricted stdlists = yes` in your local options file.
2. Stop HCL Workload Automation.
3. Stop WebSphere Application Server Liberty if present.
4. Remove the `stdlist` directory (or at least its files and subdirectories).
5. Start HCL Workload Automation.
6. Start WebSphere Application Server Liberty if present.

SSL auth mode = caonly|string|cpu

The behavior of HCL Workload Automation during an SSL handshake is based on the value of the SSL authentication mode option as follows:

caonly

HCL Workload Automation checks the validity of the certificate and verifies that the peer certificate has been issued by a recognized CA. Information contained in the certificate is not examined. The default value.

string

HCL Workload Automation checks the validity of the certificate and verifies that the peer certificate has been issued by a recognized CA. It also verifies that the Common Name (CN) of the Certificate Subject matches the string specified into the SSL auth string option. See SSL auth string = string on page 62.

cpu

HCL Workload Automation checks the validity of the certificate and verifies that the peer certificate has been issued by a recognized CA. It also verifies that the Common Name (CN) of the Certificate Subject matches the name of the workstation that requested the service.

SSL auth string = *string*

Used in conjunction with the **SSL auth mode** option when the "string" value is specified. The **SSL auth string** (ranges from 1 - 64 characters) is used to verify the certificate validity. The default string is "tws".

SSL CA certificate = *file_name*

Specify the name of the file containing the trusted certification authority (CA) certificates required for SSL authentication. The CAs in this file are also used to build the list of acceptable client CAs passed to the client when the server side of the connection requests a client certificate. This file is the concatenation, in order of preference, of the various PEM-encoded CA certificate files.

The default is `/TWS_DATA_DIR/ssl/certs/TWSClientTrustStore.pem`. This file is part of the SSL configuration distributed with the product to all customers. It is thus not secure and should be replaced with your own secure SSL configuration. See Configuring SSL attributes on page 328.

SSL certificate = *file_name*

Specify the name of the local certificate file used in SSL communication.

The default is `/TWS_DATA_DIR/ssl/certs/TWSClientTrustStore.pem`. This file is part of the SSL configuration distributed with the product to all customers. It is thus not secure and should be replaced with your own secure SSL configuration. See Configuring SSL attributes on page 328.

SSL cert type = *certificate_type*

Specify the type of the local certificate file used in SSL communication. The default value is `P12`.

SSL ciphers = *cipher_class*

Define the ciphers that the workstation supports during an SSL connection.

If you want to use an OpenSSL cipher class, use the following command to find out the list of available classes:

```
openssl ciphers
```

For a full list of supported ciphers, see SSL Ciphers and OpenSSL.

SSL cipher suites *suites*

Specify one or more supported algorithms for TLS version 1.3, This option does not apply to TLS version 1.2 or earlier.

SSL config file *file_name*

Specify the name and path of the OpenSSL configuration file. See OpenSSL documentation for details about the file format and options. If you modify this file, ensure the changes are consistent with the security configuration in your environment.

SSL FIPS compliance= yes|no

Specifies whether to enable FIPS compliance. In the current release, FIPS is not supported, so this option must not be changed.

SSL key = *file_name*

The full path to the private key file in **pem** format.

This file is part of the SSL configuration distributed with the product to all customers. It is thus not secure and should be replaced with your own secure SSL configuration. See Configuring SSL attributes on page 328.

SSL key pwd = *file_name*

The full path to the file containing the password encoded in Base64 for the private key.

The default is `<TWS_DATA_DIR>/ssl/certs/tls.sth`. This file is part of the SSL configuration distributed with the product to all customers. It is thus not secure and should be replaced with your own secure SSL configuration. See Configuring SSL attributes on page 328.

SSL random seed = *file_name*

Specify the pseudo random number file used by OpenSSL on some operating systems. Without this file, SSL authentication might not work correctly.

The default is `<TWS_DATA_DIR>/ssl/certs/tls.rnd`. This file is part of the SSL configuration distributed with the product to all customers. It is thus not secure and should be replaced with your own secure SSL configuration. See Configuring SSL attributes on page 328.

SSL version *version*

Specify the SSL version to be used. Supported values are:

- **atleast.TLSv1.0**
- **atleast.TLSv1.1**
- **atleast.TLSv1.2**
- **atleast.TLSv1.3**

where you specify the minimum version of the TLS protocol to be used. In this case, HCL Workload Automation uses the specified version of the protocol or a higher version, if supported.

- **max.TLSv1.0**
- **max.TLSv1.1**
- **max.TLSv1.2**
- **max.TLSv1.3**

where you specify the maximum version of the TLS protocol to be used. In this case, HCL Workload Automation uses the specified version of the protocol or a lower version.

- **TLSv1.0**
- **TLSv1.1**
- **TLSv1.2**
- **TLSv1.3**

where you specify the exact version of the TLS protocol to be used. In this case, HCL Workload Automation uses the specified version of the protocol.

stdlist width = *columns*

Specify the maximum width of the HCL Workload Automation console messages. You can specify a column number in the range **1** to **255**. Lines are wrapped at or before the specified column, depending on the presence of imbedded carriage control characters. Specify a negative number or zero to ignore line width. On UNIX®

and Linux® operating systems, you should ignore line width if you enable system logging with the **syslog local** option. The default is 0 columns.

switch sym prompt = *prompt*

Specify a prompt for the conman command line after you have selected a different Symphony file with the **setsym** command. The maximum length is 8 characters. The default is **n%**.

sync level = low|medium|high

Specify the rate at which HCL Workload Automation synchronizes information written to disk. This option affects all mailbox agents and is applicable to UNIX® and Linux® operating systems only. Values can be:

low

Allows the operating system to handle it.

medium

Flushes the updates to disk after a transaction has completed.

high

Flushes the updates to disk every time data is entered.

The default is **low**.

syslog local = *value*

Enables or disables HCL Workload Automation system logging for UNIX® and Linux® operating systems only. Specify **-1** to turn off system logging for HCL Workload Automation. Specify a number from **0** to **7** to turn on system logging and have HCL Workload Automation use the corresponding local facility (LOCAL0 through LOCAL7) for its messages. Specify any other number to turn on system logging and have HCL Workload Automation use the USER facility for its messages. The default is -1. See HCL Workload Automation console messages and prompts on page 132.

tcp connect timeout = *seconds*

Specify the maximum number of seconds that can be waited to establish a connection through non-blocking socket. The default is 15 seconds.

tcp timeout = *seconds*

Specify the maximum number of seconds that can be waited for the completion of a request on a connected workstation that is not responding. The default is 300 seconds.

this cpu = *workstation_name*

The unique identifier of the workstation. Even when the workstation is subsequently moved to a different folder, the unique identifier remains the same. The name can be a maximum of 16 alphanumeric characters in length and must start with a letter. When a switch is made between the master domain manager and a backup domain manager, using the switchmgr command, the Symphony header value for this cpu is overwritten by the this cpu value in the `localopts` file. The default is the host name of the computer.

timeout = ***seconds***

The timeout in seconds to await for the server operation completion was reached. The command continues to run on the server until its completion. The default value is 3600 seconds.

unison network directory = ***directory_name***

This parameter applies only to versions of HCL Workload Automation prior to version 8.3. Defines the name of the Unison network directory. The default is `TWA_home>/../unison/network`.

useropts = ***file_name***

If you have multiple instances of HCL Workload Automation on a system, use this to identify the *useropts* file that is to be used to store the connection parameters for the instance in which this *localops* file is found. See Multiple product instances on page 71 for more details.

wr enable compression = yes|no

Use this option on fault-tolerant agents. Specify if the fault-tolerant agent can receive the Symphony file in compressed form from the master domain manager. The default is **no**.

wr read = ***seconds***

Specify the number of seconds the Writer process waits for an incoming message before checking for a termination request from Netman. The default is 600 seconds.

wr unlink = ***seconds***

Specify the number of seconds the Writer process waits before exiting if no incoming messages are received. The minimum is 120 seconds. The default is 180 seconds.

Local options file example

The following is an example of a default `localopts` file:

Note: Some parameters might not be present depending upon your version and configuration.

Example

```
#############################################################################
# Licensed Materials - Property of IBM* and HCL**
# 5698-WSH
# (c) Copyright IBM Corp. 1998, 2016 All rights reserved.
# (c) Copyright HCL Technologies Ltd. 2016, 2024 All rights reserved.
# * Trademark of International Business Machines
# ** Trademark of HCL Technologies Limited
#############################################################################
#
# The HCL Workload Scheduler localopts file defines the attributes of this
# workstation, for various processes.
#
#----------------------------------------------------------------------------
# General attributes of this workstation:
#
thiscpu=KU-DWK-TCL122
```

```
merge stdlists      =yes
stdlist width       =0
syslog local        =-1
restricted stdlists =no
#
#----------------------------------------------------------------------------
# The attributes of this workstation for the batchman process:
#
bm check file      =120
bm check status    =300
bm look            =5
bm read            =3
bm stats           =off
bm verbose         =off
bm check until     =300
bm check deadline  =0
bm late every      =0
#
#----------------------------------------------------------------------------
# The attributes of this workstation for the jobman process:
#
jm job table size  =1024
jm look            =300
jm nice            =0
jm promoted nice   =-1    #UNIX
jm promoted priority =AboveNormal #WINDOWS
jm no root         =yes
jm file no root    =no
jm read            =10
jm load user profile= on
jm loaduserprofile =on
#
#----------------------------------------------------------------------------
# The attributes of this workstation for the TWS mailman process:
#
mm response        =600
mm retrylink       =600
mm sound off       =no
mm unlink          =960
mm cache mailbox   =yes
mm cache size      =512
mm resolve master  =no
autostart monman   =yes
mm symphony download timeout =0
#
#----------------------------------------------------------------------------
# The attributes of this workstation for the netman process:
#
nm mortal          =no
nm port            =31111
nm read            =10
nm retry           =800
#
#----------------------------------------------------------------------------
# The attributes of this workstation for the writer process:
#
wr read                =600
wr unlink              =180
wr enable compression  =no
#
#----------------------------------------------------------------------------
# Optional attributes of this Workstation for remote database files
#
# mozart directory =         /opt/HCL/TWA/TWS/mozart
# parameters directory =     /opt/HCL/TWA
# unison network directory = /opt/HCL/TWA/TWS/../unison/network
#
#----------------------------------------------------------------------------
# The attributes of this workstation for custom formats
#
date format             =1 # The possible values are 0-yyyy/mm/dd, 1-mm/dd/yyyy, 2-dd/mm/yyyy, 3-NLS.
composer prompt         =-
conman prompt           =%
switch sym prompt       =<n>%
#
#----------------------------------------------------------------------------
# The attributes of this workstation for the customization of I/O on mailbox files
#
sync level              =low
#
#----------------------------------------------------------------------------
# The attributes of this workstation for networking
#
tcp timeout             =300
tcp connect timeout     =15
#
#----------------------------------------------------------------------------
# General Secure options
#
SSL auth mode           =caonly
#
# Use "SSL auth string" only if "SSL auth mode" is set to "string"
#
SSL auth string     =tws
#
## This flag set to "yes" require the FIPS 140-2 policies. The default value is "no".
#
SSL FIPS compliance     =no
#
# This property specify the path to the file containing OpenSSL configuration properties.
#
SSL config file  ="/opt/HCL/TWA/TWSDATA/ssl/openssl.cnf"
#
# Netman full SSL port, use "nm SSL full port"it on if "enSSLFullConnection" is set to "yes"
# the value "0" means port close
#
nm SSL full port = 31113
#
# Netman SSL port
# the value "0" means port close
#
nm SSL port             =0
#
# End General Secure options
#----------------------------------------------------------------------------
```

```
#----------------------------------------------------------------------------
# OpenSSL options
#
# This section is ignored for agent dynamic only
#
# when custom certificate are provided in SSL folder by the user,
# these lines are substituted by the installation with correct datadir path
# pointing to the following files: "tls.crt", "tls.sth", "tls.key", "tls.rnd", "ca.crt"

SSL key                 =
SSL certificate = "/opt/HCL/TWA/TWSDATA/ssl/certs/TWSClientKeyStore.p12"
SSL key pwd = "/opt/HCL/TWA/TWSDATA/ssl/certs/tls.sth"
SSL CA certificate = "/opt/HCL/TWA/TWSDATA/ssl/certs/TWSClientTrustStore.pem"
SSL random seed = "/opt/HCL/TWA/TWSDATA/ssl/certs/tls.rnd"
SSL cert type = "P12"

SSL Ciphers =
SSL Version = atleast.TLSv1.2
#
#CLI SSL server auth =
CLI SSL Ciphers =
CLI SSL cipher suites =
CLI SSL Version = atleast.TLSv1.2
#CLI SSL server certificate =
#CLI SSL trusted dir =
#----------------------- End OpenSSL options --------------------------------

#----------------------------------------------------------------------------
# Encryption options
#
encrypt keystore file = "/opt/HCL/TWA/TWSDATA/ssl/aes/key.p12"
encrypt keystore pwd = "/opt/HCL/TWA/TWSDATA/ssl/aes/key.sth"
encrypt label = "default"
#decrypt label list =
#----------------------- End Encryption options ----------------------------------

#----------------------------------------------------------------------------
# The TWS instance has been installed as REMOTE CLI
IS REMOTE CLI = no  # yes for a REMOTE CLI installation, no otherwise

#----------------------------------------------------------------------------
# Attributes for CLI connections
#
# General attributes for CLI connections
#
HOST              = 127.0.0.1        # Master hostname used when attempting a connection.
PROTOCOL          = https            # Protocol used to establish a connection with the Master.
PORT              = 31116            # Protocol port
#PROXY            =
#PROXYPORT        =
TIMEOUT           = 3600       # Timeout in seconds to wait a server response.
#CLI SSL SERVER AUTH = yes

DEFAULTWS         = KU-DWK-TCL122
USEROPTS          = useropts_ITAuser

#----------------------------------------------------------------------------
# Event Management parameters
```

```
#
CAN BE EVENT PROCESSOR = yes # yes for MDM and BKM, no otherwise

#----------------------------------------------------------------------------
# Centralized Agent Update
#
#DownloadDir =

#----------------------------------------------------------------------------
# Current Folder
#
#current folder = /


#----------------------------------------------------------------------------
# Appserver Watchdog default settings
#
Appserver check interval =                      3               #minutes
Appserver auto restart =                        yes             #yes/no
Appserver min restart time =                    2               #minutes
Appserver max restarts =                        5               #restarts number
Appserver count reset interval =                24              #hours
```

Note: The "REMOTE CLI" term indicates the command line client.

Setting user options

Set the user options you require for each user on a workstation in the `useropts` file. Changes do not take effect until HCL Workload Automation is stopped and restarted.

The `useropts` file contains values for `localopts` parameters that must be customized for an individual user. Both files are located in the `*user_home*/.TWS` directory of the user. When HCL Workload Automation needs to access data from the `localopts` file, it looks first to see if the property it requires is stored also in the `useropts` file for the current user. The value specified in the `useropts` file always takes precedence over the value in the `localopts` file. If a property is not specified when invoking the command that requires it, or in the `useropts` and `localopts` files, an error is displayed.

The main use of the `useropts` file is to store the user-specific connection parameters for accessing the command line client, for example conman or composer.

You can define authentication using either the **username** and **password** parameters or the **JWT** parameter:

username

User name used to access the master domain manager. The user must be defined in the security file on the master domain manager. For more information, see the section about configuring user authorization in *Administration Guide*.

password

Password used to access the master domain manager. The presence of the `ENCRYPT` or `AES` label in the password field indicates that the specified setting has been encrypted; if this label is not present, you must exit and access the interface program again to allow the encryption of that field.

JWT

JSON Web Token (JWT) used to access the master domain manager. The **JWT** parameter always takes precedence over the **username** and **password** parameters. If it is incorrect, the connection does not take place, even if the **username** and **password** parameters are present and correct. You retrieve the JWT from the Dynamic Workload Console.

Note that these parameters are not stored in the `localopts` file.

For more information, see Configuring command-line client access authentication on page 123.

A `useropts` file is created for the *TWS_user* during the installation, but you must create a separate file for each user that needs to use user-specific parameters on a workstation.

Multiple product instances

Because HCL Workload Automation supports multiple product instances installed on the same computer, there can be more than one instance of the `useropts` file per user. This is achieved by giving the `useropts` files for a user different names for each instance.

In the `localopts` file of each instance the option named ***useropts*** identifies the file name of the `useropts` file that has to be accessed in the *`user_home/`*`.TWS` directory to connect to that installation instance.

This means that, for example, if two HCL Workload Automation instances are installed on a computer and the user `operator` is a user of both instances, you could define the `useropts` credentials as follows:

- In the `localopts` file of the *first* instance the local option *`useropts`*` = useropts1` identifies the *`operator_home/`*`.TWS/useropts1` file containing the connection parameters settings that user `operator` needs to use to connect to the *first* HCL Workload Automation instance.
- In the `localopts` file of the *second* HCL Workload Automation instance the local option *`useropts`*` = useropts2` identifies the *`operator_home/`*`.TWS/useropts2` file containing the connection parameters settings that user `operator` needs to use to connect to the *second* HCL Workload Automation instance.

Configuring the agent

The configuration settings of the agent are stored in the `JobManager.ini` file. Starting with version 10.2.4, if you are integrating with a password vault tool, the related configuration is no longer saved in the `JobManager.ini` file. Instead, it is now stored in a profile and, optionally, in a configuration file. For more information, see Configuring the agent to work with a password vault on page 90.

In a distributed environment, if a gateway is configured to allow the master domain manager or dynamic domain manager to communicate with a dynamic agent located behind a network boundary, then the gateway configuration settings of the agent are contained in the `JobManagerGW.ini` file. This file is almost identical to the `JobManager.ini` file, however, only parameters in the [ITA], [Env], and [ResourceAdvisorAgent] sections require configuration. For these parameters, definitions are given for both the `JobManager.ini` and `JobManagerGW.ini` files.

To find out where these files are located, see the section about installation paths in *HCL Workload Automation: Planning and Installation*.

Only a subset of the available parameters is documented, because some parameters are reserved for internal use.

These files are made up of many different sections. Each section name is enclosed between square brackets and each section includes a sequence of `variable = value` statements.

You can customize properties for the following:

- Event-driven workload automation properties
- Log properties
- Trace properties when the agent is stopped. You can also customize traces when the agent is running using the procedure described in Configuring trace properties when the agent is running on page 78.
- Native job executor
- Java™ job executor
- Resource advisor agent
- System scanner

The log messages are written in the following file:

On Windows operating systems:

<*TWA_home*>`\TWS\stdlist\JM\JobManager_message.log`

On UNIX and Linux operating systems:

<*TWA_DATA_DIR*>`/stdlist/JM/JobManager_message.log`

The trace messages are written in the following file:

On Windows operating systems:

- <*TWA_home*>`\TWS\stdlist\JM\ITA_trace.log`
- <*TWA_home*>`\TWS\stdlist\JM\JobManager_trace.log`
- <*TWA_home*>`\TWS\JavaExt\logs\javaExecutor0.log`

On UNIX and Linux operating systems:

- <*TWA_DATA_DIR*>`/stdlist/JM/ITA_trace.log`
- <*TWA_DATA_DIR*>`/stdlist/JM/JobManager_trace.log`
- <*TWA_DATA_DIR*>`/JavaExt/logs/javaExecutor0.log`

Logging information about job types with advanced options

You can use the `logging.properties` file to configure the logging process for job types with advanced options, with the exception of the Executable and Access Method job types.

The `logging.properties` file is located on the HCL Workload Automation for Z Agent, located in the following path:

On Windows operating systems:

<*TWA_home*>`/TWS/JavaExt/cfg/logging.properties`

On UNIX and Linux operating systems:

<*TWA_DATA_DIR*>`/JavaExt/cfg/logging.properties`

.

After installation, this file is as follows:

```
# Specify the handlers to create in the root logger
# (all loggers are children of the root logger)
# The following creates two handlers
handlers = java.util.logging.ConsoleHandler,
           java.util.logging.FileHandler

# Set the default logging level for the root logger
.level = INFO

# Set the default logging level for new ConsoleHandler instances
java.util.logging.ConsoleHandler.level = INFO

# Set the default logging level for new FileHandler instances
java.util.logging.FileHandler.level
    = ALL
java.util.logging.FileHandler.pattern
    = C:\TWA_home\TWS\JavaExt\logs\javaExecutor%g.log
java.util.logging.FileHandler.limit
    = 1000000
java.util.logging.FileHandler.count
    = 10

# Set the default formatter for new ConsoleHandler instances
java.util.logging.ConsoleHandler.formatter =
            java.util.logging.SimpleFormatter
java.util.logging.FileHandler.formatter =
            java.util.logging.SimpleFormatter

# Set the default logging level for the logger named com.mycompany
com.ibm.scheduling = INFO
```

You can customize:

- The logging level (from INFO to WARNING, ERROR, or ALL) in the following keywords:

  `.level`

  Defines the logging level for the internal logger.

  `com.ibm.scheduling`

  Defines the logging level for the job types with advanced options. To log information about job types with advanced options, set this keyword to ALL.

- The path where the logs are written, specified by the following keyword:

  ```
  java.util.logging.FileHandler.pattern
  ```

Not all the properties in the `JobManager.ini` and `JobManagerGW.ini` files can be customized. For a list of the configurable properties, see the following sections:

- Configuring log message properties [JobManager.Logging.cclog] on page 75
- Configuring trace properties when the agent is stopped [JobManager.Logging.cclog] on page 77.
- Configuring common launchers properties [Launchers] on page 81
- Configuring properties of the native job launcher [NativeJobLauncher] on page 83.
- Configuring properties of the Java job launcher [JavaJobLauncher] on page 86.
- Configuring properties of the Resource advisor agent [ResourceAdvisorAgent] on page 87.
- Configuring properties of the System scanner [SystemScanner] on page 89
- Configuring environment variables [Env] on page 90
- The section about configuring properties of event-driven workload automation [EventDrivenWorkload] in *Scheduling End-to-end with z-centric Capabilities*

Configuring general properties [ITA]

About this task

In the `JobManager.ini` or `JobManagerGW.ini` file, you can add some general properties to the following section:

```
[ITA]
```

You can add or modify the following properties:

ActionPollers

The number of the thread processes started on the gateway workstation to communicate with the broker server installed on the master domain manager or dynamic domain manager. The default value is 1. Specify this value if you have more than 100 dynamic agents that communicate with the broker server installed on the master domain manager or dynamic domain manager by using the same gateway. Restart the agent after the property change.

http_proxy

The URL of the proxy configured in a distributed environment through which agents or gateways communicate to the broker server installed on the master domain manager or dynamic domain manager. The value is `https_proxy =http://<proxy_workstation>:<proxy_workstation_port>`, where:

- *<proxy_workstation>* is the fully qualified host name of the workstation where the proxy is configured.
- *<proxy_workstation_port>* is the port number of the workstation where the proxy is configured.

Restart the agent after the property change.

DebugDir

You can use this parameter to enable tracing for the dynamic agent to help determine what information is being sent to and from a dynamic agent workstation. Perform the following steps:

1. Create a directory to dump the files sent and received by the dynamic agent, for example `/tmp/DA_DD`. This directory needs to be writable by the user owning the dynamic agent.
2. Add the **DebugDir** parameter to the path of the directory you created, for example:

```
DebugDir = /tmp/DA_DD
```

3. Restart the dynamic agent using the following commands:

```
ShutDownLwa
StartUpLwa
```

Remember to monitor the debug directory on a regular basis to ensure it does not become too large. No automatic check is performed on the debug director. For more information, see Enable packet tracing for dynamic agent using ITA parameter DebugDir.

Configuring log message properties [JobManager.Logging.cclog]

About this task

To configure the logs, edit the [JobManager.Logging.cclog] section in the `JobManager.ini` file. This procedure requires that you stop and restart the HCL Workload Automation agent

The section containing the log properties is named:

```
[JobManager.Logging.cclog]
```

You can change the following properties:

JobManager.loggerhd.fileName

The name of the file where messages are to be logged. the default value is

On Windows operating systems

`TWA_home\stdlist\JM\JOBMANAGER-FFDC`

On UNIX operating systems

`$(`*TWA_DATA_DIR*`)/stdlist/JM/JobManager_message.log`

JobManager.loggerhd.maxFileBytes

The maximum size that the log file can reach. The default is **1024000** bytes.

JobManager.loggerhd.maxFiles

The maximum number of log files that can be stored. The default is **3**.

JobManager.loggerhd.fileEncoding

By default, log files for the agent are coded in UTF-8 format. If you want to produce the log in a different format, add this property and specify the required codepage.

JobManager.loggerfl.level

The amount of information to be provided in the logs. The value ranges from 3000 to 7000. Smaller numbers correspond to more detailed logs. The default is **3000**.

JobManager.ffdc.maxDiskSpace

Exceeding this maximum disk space, log files collected by the first failure data capture mechanism are removed, beginning with the oldest files first.

JobManager.ffdc.baseDir

The directory to which log and trace files collected by the ffdc tool are copied. The default directory is

On Windows operating systems

`TWA_home\stdlist\JM\JobManager_message.log`

On UNIX operating systems

`$(`*TWA_DATA_DIR*`)/stdlist/JM/JobManager_message.log`

JobManager.ffdc.filesToCopy

Log and trace files (`JobManager_message.log` and `JobManager_trace.log`) collected by the ffdc tool located in <*TWA_home*>`\TWS\stdlist\JM`. The files are available in the following paths:

On Windows operating systems

- TWA_home`/TWS/stdlist/JM/JobManager_message.log`
- TWA_home`/TWS/stdlist/JM/JobManager_trace.log`

On UNIX operating systems

- `$(`*TWA_DATA_DIR*`)/stdlist/JM/JobManager_message.log`
- `$(`*TWA_DATA_DIR*`)/stdlist/JM/JobManager_trace.log`

When a message is logged (JobManager.ffdc.triggerFilter = JobManager.msgIdFilter) that has an ID that matches the pattern "AWSITA*E" (JobManager.msgIdFilter.msgIds = AWSITA*E), which corresponds to all error messages, then the log and trace files (JobManager.ffdc.filesToCopy = "/opt/HCL/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_message.log" "/opt/HCL/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_trace.log") are copied (JobManager.ffdc.className = ccg_ffdc_filecopy_handler) to the directory `JOBMANAGER-FFDC`

(JobManager.ffdc.baseDir = /opt/HCL/TWA_<TWS_user>/TWS/stdlist/JM/JOBMANAGER-FFDC). If the files copied exceed 10 MB (JobManager.ffdc.maxDiskSpace = 10000000), then the oldest files are removed first (JobManager.ffdc.quotaPolicy = QUOTA_AUTODELETE).

After installing the z-centric agent or dynamic agent on Windows 2012, the `JobManager_message.log` might not be created. In this case, perform the following procedure:

1. Stop the agent.
2. Create a backup copy of `JobManager.ini`, and edit the original file by changing the row:

   ```
   JobManager.loggerhd.className = ccg_multiproc_filehandler
   ```

   to

   ```
   JobManager.loggerhd.className = ccg_filehandler
   ```

3. Restart the agent.

Configuring trace properties when the agent is stopped [JobManager.Logging.cclog]

How to configure the trace properties when the agent is stopped.

To configure the trace properties when the agent is stopped, edit the [JobManager.Logging] section in the `JobManager.ini` file and then restart the HCL Workload Automation agent.

The section containing the trace properties is named:

```
[JobManager.Logging.cclog]
```

You can change the following properties:

JobManager.trhd.fileName

The name of the trace file. the default path is as follows:

On Windows operating systems

`TWA_home/TWS/stdlist/JM/JobManager_trace.log`

On UNIX operating systems

`$(`*TWA_DATA_DIR*`)/stdlist/JM/JobManager_trace.log`

JobManager.trhd.maxFileBytes

The maximum size that the trace file can reach. The default is 10240000 bytes.

JobManager.trhd.maxFiles

The maximum number of trace files that can be stored. The default is 5.

JobManager.trfl.level

Determines the type of trace messages that are logged. Change this value to trace more or fewer events, as appropriate, or on request from HCL Software Support. Valid values are:

DEBUG_MAX

Maximum tracing. Every trace message in the code is written to the trace logs.

INFO

All *informational*, *warning*, *error* and *critical* trace messages are written to the trace. The default value.

WARNING

All *warning*, *error* and *critical* trace messages are written to the trace.

ERROR

All *error* and *critical* trace messages are written to the trace.

CRITICAL

Only messages which cause the agent to stop are written to the trace.

The output trace (`JobManager_trace.log`) is provided in XML format.

After installing the z-centric agent or dynamic agent on Windows 2012, the `JobManager_trace.log` might not be created. In this case, perform the following procedure:

1. Stop the agent.
2. Create a backup copy of `JobManager.ini`, and edit the original file by changing the row:

   ```
   JobManager.trhd.className = ccg_multiproc_filehandler
   ```

   to

   ```
   JobManager.trhd.className = ccg_filehandler
   ```

3. Restart the agent.

Configuring trace properties when the agent is running

Use the **twstrace** command to set the trace on the agent when it is running.

Using the **twstrace** command, you can perform the following actions on the agent when it is running:

- See command usage and verify version on page 79.
- Enable or disable trace on page 79.
- Set the traces to a specific level, specify the number of trace files you want to create, and the maximum size of each trace file. See Set trace information on page 79.
- Show trace information on page 80.
- Collect trace files, message files, and configuration files in a compressed file using the command line. See Collect trace information on page 80.

You can also configure the traces when the agent is not running by editing the [JobManager.Logging] section in the `JobManager.ini` file as described in Configuring the agent on page 71. This procedure requires that you stop and restart the agent.

twstrace command

Use the **twstrace** command to configure traces, and collect logs, traces, and configuration files (ita.ini and jobManager.ini) for agents. You collect all the information in a compressed file when it is running without stopping and restarting it.

See command usage and verify version

To see the command usage and options, use the following syntax.

Syntax

twstrace -u | **-v**

Parameters

-u

Shows the command usage.

-v

Shows the command version.

Enable or disable trace

To set the trace to the maximum or minimum level, use the following syntax.

Syntax

twstrace -enable | **-disable**

Parameters

-enable

Sets the trace to the maximum level. The maximum level is **1000**.

-disable

Sets the trace to the minimum level. The minimum level is **3000**.

Set trace information

To set the trace to a specific level, specify the number of trace files you want to create, and the maximum size the trace files can reach, use the following syntax.

Syntax

twstrace [**-level** <level_number>] [**-maxFiles** <files_number>] [**-maxFileBytes** <bytes_number>]

Parameters

-level <level_number>

Sets the trace level. Specify a value in the range from 1000 to 3000, which is also the default value. Note that if you set this parameter to 3000, you have the lowest verbosity level and the fewest trace messages. To have a better trace level, with the most verbose trace messages and the maximum trace level, set it to **1000**.

-maxFiles <files_number>

Specify the number of trace files you want to create.

-maxFileBytes <bytes_number>

Set the maximum size in bytes that the trace files can reach. The default is **1024000** bytes.

Show trace information

To display the current trace level, the number of trace files, and the maximum size the trace files can reach, use the following syntax.

Syntax

twstrace -level | **-maxFiles** | **-maxFileBytes**

Parameters

-level

See the trace level you set.

-maxFiles

See the number of trace files you create.

-maxFileBytes

See the maximum size you set for each trace file

Example

Sample

The example shows the information you receive when you run the following command:

```
twstrace -level -maxFiles -maxFileBytes
```

```
AWSITA176I The trace properties are: level="1000",
max files="3", file size="1024000".
```

Collect trace information

To collect the trace files, the message files, and the configuration files in a compressed file, use the following syntax.

Syntax

twstrace -getLogs [**-zipFile** <compressed_file_name>] [**-host** <host_name>] [**-protocol** {http | https } [**-port** <port_number>][**-iniFile** <ini_file_name>]

Parameters

-zipFile <compressed_file_name>

Specify the name of the compressed file that contains all the information, that is logs, traces, and configuration files (ita.ini and jobManager.ini) for the agent. The default is **logs.zip**.

-host <host_name>

Specify the host name or the IP address of the agent for which you want to collect the trace. The default is **localhost**.

-protocol http|https

Specify the protocol of the agent for which you are collecting the trace. The default is the protocol specified in the **.ini** file of the agent.

-port <port_number>

Specify the port of the agent. The default is the port number of the agent where you are running the command line.

-iniFile <ini_file_name>

Specify the name of the **.ini** file that contains the SSL configuration of the agent for which you want to collect the traces. If you are collecting the traces for a remote agent for which you customized the security certificates, you must import the certificate on the local agent and specify the name of the **.ini** file that contains this configuration. To do this, perform the following actions:

1. Extract the certificate from the keystore of the remote agent.
2. Import the certificate in a local agent keystore. You can create an ad hoc keystore whose name must be **TWSClientKeyStore.kdb**.
3. Create an **.ini** file in which you specify:
   - **0** in the **tcp_port** property as follows:

     ```
     tcp_port=0
     ```

   - The port of the remote agent in the **ssl_port** property as follows:

     ```
     ssl_port=<ssl_port>
     ```

   - The path to the keystore you created in Step 2 on page 81 in the **key_repository_path** property as follows:

     ```
     key_repository_path=<local_agent_keystore_path>
     ```

Configuring common launchers properties [Launchers]

About this task

In the `JobManager.ini` file, the section containing the properties common to the different launchers (or executors) is named:

```
[Launchers]
```

The following properties are available:

BaseDir

The installation path of the HCL Workload Automation agent. Do not modify this value.

CommandHandlerMinThreads

Indicates the maximum number of commands that can be run on the agent concurrently. Limits to the number of jobs vary depending on the resources of your workstation, however consider that operations on comdhandler are usually short. The default is **20**. Usually, there is no need to modify this setting, even if you plan a very high workload on the agent. You might want to change it if many commands are run concurrently on the agent, for example, many concurrent requests to retrieve job logs.

CommandHandlerMaxThreads

Indicates the maximum number of commands that can be run on the agent concurrently. Limits to the number of jobs vary depending on the resources of your workstation, however consider that operations on comdhandler are usually short. The default is **100**. Usually, there is no need to modify this setting, even if you plan a very high workload on the agent. You might want to change it if many commands are run concurrently on the agent, for example, many concurrent requests to retrieve job logs.

CpaHeartBeatTimeSeconds

The polling interval in seconds used to verify if the **agent** process is still up and running. If the agent process is inactive the product stops also the **JobManager** process. The default is **30**. Modify only if you use dynamic pools with CPU-based requirements or optimization policies. With a lower value, the agent reacts quickly to CPU modifications, but this might cause unstable values in case of CPU spikes. Lower values causes a higher use of resources on the agent.

DirectoryPermissions

The access rights assigned to the agent for creating directories when running jobs. The default is **0755**. Supported values are UNIX-format entries in hexadecimal notation.

DownloadDir

The name of the directory where the fix pack installation package or upgrade eImage for dynamic agents is downloaded during the centralized agent update process. If not specified, the following default directory is used:

On Windows operating systems:

`TWA_home\TWS\stdlist\JM\download`

On UNIX operating systems:

TWA_DATA_DIR`/TWS/stdlist/JM/download`

The centralized agent update process does not apply to z-centric agents.

ExecutorsMaxThreads

Specifies the maximum number of jobs the dynamic agent can run concurrently. For example, to allow the dynamic agent to run a maximum of 500 jobs concurrently, set this parameter to **500**. The default is **400**.

ExecutorsMinThreads

Specifies the minimum number of jobs the dynamic agent can run concurrently. For example, to allow the dynamic agent to run a minimum of 500 jobs concurrently, set this parameter to **500**. The default is **38**. Modify if the number of expected concurrent jobs is much higher than 38. The agent dynamically allocates more threads if necessary, until it reaches the value specified in **ExecutorsMaxThreads**.

FilePermissions

The access rights assigned to the agent for creating files when running jobs. The default is **0755**. Supported values are UNIX-format entries in hexadecimal notation.

MaxAge

The number of days that job logs are kept (in path `TWA_home/TWS/stdlidst/JM`) before being deleted. The default is **30**. Possible values range from a minimum of 1 day.

NotifierMaxThreads

Notifier threads are in charge of notifying the dynamic workload broker of each status change in a job. This parameter specifies the maximum number of job status changes that can be notified to the dynamic workload broker.

NotifierMinThreads

Notifier threads are in charge of notifying the dynamic workload broker of each status change in a job. This parameter specifies the minimum number of job status changes that can be notified to the dynamic workload broker. The default value is **3**. Modify this parameters only in case of unexpected errors and after consulting with software support team.

SpoolDir

The path to the folder containing the jobstore and outputs. The default is:

On Windows operating systems

`TWA_home/TWS/stdlist/JM`

On UNIX operating systems

`$(TWA_DATA_DIR/stdlist/JM`

StackSizeBytes

The size of the operating system stack in bytes. The default is **DEFAULT**, meaning that the **agent** uses the default value for the operating system. Do not modify this parameter unless instructed to do so by the software support team. Incorrect values can cause the agent to crash.

Configuring properties of the native job launcher [NativeJobLauncher]

About this task

In the `JobManager.ini` file, the section containing the properties of the native job launcher is named:

```
[NativeJobLauncher]
```

You can change the following properties:

AllowRoot

Applies to UNIX™ systems only. Specifies if the root user can run jobs on the agent. It can be `true` or `false`. The default is false. This property does not apply to IBM i, use the AllowQSECOFR option instead

AllowQECOFR

Applies to IBM i systems only. Specifies if QSECOFR user can run jobs on the agent. It can be `true` or `false`. The default is `true`. Add a line like AllowQSECOFR = `false` to the JobManager.ini file to deny job execution to QSECOFR.

CheckExec

If `true`, before launching the job, the agent checks both the availability and the execution rights of the binary file. The default is `true`.

DefaultWorkingDir

Specifies the working directory of native jobs. You can also specify the value for the working directory when creating or editing the job definition in the Graphical Designer. When specified in the Graphical Designer, this value overrides the value specified for the **DefaultWorkingDir** property. If you do not specify any working directories, the `<`*`TWS_home`*`>\bin` directory is used.

JobUnspecifiedInteractive

Applies to Windows™ operating systems only. Specifies if native jobs are to be launched in interactive mode. It can be `true` or `false`. The default is `false`.

KeepCommandTraces

Set to `true` to store the traces of the method invocation for actions performed on a job definition, for example, when selecting from a picklist. These files are stored in the path `/opt/HCL/TWA_<TWS_user>/TWS/stdlist/JM/r3batch_cmd_exec`. The default setting is `false`.

KeepJobCommandTraces

Set to `true` to store the traces of the method invocation for actions performed on a job instance, for example, viewing a spool list. These files are stored in the .zip file of the job instance. The default setting is `true`.

LoadProfile

Applies to agents on Windows servers only. Specifies if the user profile is to be loaded. It can be `true` or `false`. The default is `true`.

MonitorQueueName

Specifies the name of the queue where the IBM i jobs are monitored. If you do not specify this property, the default queue (QBATCH) is used.

PortMax

The maximum range of the port numbers used by the task launcher to communicate with the Job Manager. The default is 0, meaning that the operating system assigns the port automatically.

PortMin

The minimum range of the port numbers used by the task launcher to communicate with the Job Manager. The default is 0, meaning that the operating system assigns the port automatically.

PostJobExecScriptPathName

The fully qualified path of the script file that you want to run when the job completes. By default, this property is not present in the `JobManager.ini` file. If you do not specify any file path or the script file doesn't exist, no action is taken.

This property applies to dynamic agent and z/OS agent. For details about running a script when a job completes, see *User's Guide and Reference*.

PromotedNice

Used in workload service assurance. This property is not supported on the Agent for z/OS.

For UNIX and Linux operating systems only, assigns the priority value to a critical job that needs to be promoted so that the operating system processes it before others. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time.

Boundary values vary depending upon each specific platform, but generally lower values correspond to higher priority levels and vice versa. The default is -1.

Be aware that:

- The promotion process is effective with negative values only. If you set a positive value, the system runs it with the -1 default value.
- An out of range value (for example -200), prompts the operating system to automatically promote the jobs with the lowest allowed nice value.
- Overusing the promotion mechanism (that is, defining an exceedingly high number of jobs as mission critical and setting the highest priority value here) might overload the operating system, negatively impacting the overall performance of the workstation.

PromotedPriority

Used in workload service assurance. This property is not supported on the Agent for z/OS.

For Windows operating systems only, sets to this value the priority by which the operating system processes a critical job when it is promoted. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time. Valid values are:

- `High`
- `AboveNormal` (the default)
- `Normal`
- `BelowNormal`
- `Low` or `Idle`

Note that if you a set a lower priority value than the one non-critical jobs might be assigned, no warning is given.

RequireUserName

When `true`, requires that you add the user name in the JSDL job definition.

When `false`, runs with the user name used by job manager, that is:

- *TWS_user* on UNIX™ and Linux™ systems
- The local system account on Windows™ systems

The default is `false`.

RunExecutablesAsIBMiJobs

If you set this property to `true`, you can define IBM i jobs as generic jobs without using the XML definition. Generic jobs are automatically converted to IBM i jobs. As a side effect, generic jobs cannot be run when this parameter is enabled (`RunExecutablesAsIBMiJobs=true`). There is no default value because this property is not listed in the `JobManager.ini` file after the agent installation.

If you set this property to `true`, ensure that the user you used to install the agent has been granted the `*ALLOBJ` special authority.

RunInteractiveJobOnInvalidSession

Applies only to native and executable jobs starting interactive programs running on dynamic agents installed on Windows operating systems. Interactive programs run only if the job user has an active session open when the job runs. If there is no active session for the job user, the job behavior is defined by this property, as follows. Set the property to `true` to enable jobs to start interactive programs even if there is no active session for the job user. Set the property to `false` to prevent jobs from starting interactive programs if there is no active session for the job user.

ScriptSuffix

The suffix to be used when creating the script files. It is:

`.cmd`

For Windows™

`.sh`

For UNIX™

VerboseTracing

Enables verbose tracing. It is set to `true` by default.

Configuring properties of the Java™ job launcher [JavaJobLauncher]

About this task

In the `JobManager.ini` file, the section containing the properties of the Java™ job launcher is named:

```
[JavaJobLauncher]
```

You can change the following property:

JVMOptions

The options to provide to the Java™ Virtual Machine used to start job types with advanced options. Supported keywords for establishing a secure connection are:

- htttps.proxyHost
- https.proxyPort

Supported keywords for establishing a non-secure connection are:

- Dhttp.proxyHost
- Dhttp.proxyPort

For example, to set job types with advanced options, based on the default JVM http protocol handler, to the unauthenticated proxy server called with name myproxyserver.mycompany.com, define the following option:

```
JVMOptions = -Dhttp.proxyHost=myproxyserver.mycompany.com -Dhttp.proxyPort=80
```

Configuring properties of the Resource advisor agent [ResourceAdvisorAgent]

About this task

In the `JobManager.ini` and `JobManagerGW.ini` files, the section containing the properties of the Resource advisor agent is named:

```
[ResourceAdvisorAgent]
```

You can change the following properties:

BackupResourceAdvisorUrls

The list of URLs returned by the HCL Workload Automation master in a distributed environment or by the dynamic domain manager either in a z/OS or in a distributed environment. The agent uses this list to connect to the master or dynamic domain manager.

CPUScannerPeriodSeconds

The time interval that the Resource advisor agent collects resource information about the local CPU. The default value is every 10 seconds.

FullyQualifiedHostname

The fully qualified host name of the agent. It is configured automatically at installation time and is used to connect with the master in a distributed environment or with the dynamic domain manager in a z/OS or in a distributed environment. Edit only if the host name is changed after installation.

NotifyToResourceAdvisorPeriodSeconds

The time interval that the Resource advisor agent forwards the collected resource information to the Resource advisor. The default value is every 119 seconds.

ResourceAdvisorUrl

JobManager.ini

The URL of the master in a distributed environment, or of the dynamic domain manager in a z/OS or in a distributed environment, that is hosting the agent. This URL is used until the server replies with the list of its URLs. The value is `https://$(tdwb_server):$(tdwb_port)/JobManagerRESTWeb/JobScheduler/resource`, where:

$(*tdwb_server*)

is the fully qualified host name of the master in a distributed environment or of the dynamic domain manager either in a z/OS or in a distributed environment.

$(*tdwb_port*)

is the port number of the master in a distributed environment or of the dynamic domain manager either in a z/OS or in a distributed environment.

It is configured automatically at installation time. Edit only if the host name or the port number are changed after installation, or if you do not use secure connection (set to `http`). If you set the port number to zero, the resource advisor agent does not start. The port is set to zero if at installation time you specify that you will not be using the master in a distributed environment or the dynamic domain manager either in a z/OS or in a distributed environment.

In a distributed environment, if **-gateway** is set to either `local` or `remote`, then this is the URL of the dynamic agent workstation where the gateway resides and through which the dynamic agents communicate. The value is `https://$(tdwb_server):$(tdwb_port)/ita/JobManagerGW/JobManagerRESTWeb/JobScheduler/resource`, where:

$(*tdwb_server*)

The fully qualified host name of the dynamic agent workstation where the gateway resides and through which the dynamic agent communicates with the dynamic workload broker.

$(*tdwb_port*)

The port number of the dynamic agent workstation where the gateway resides and through which the dynamic agent communicates with the dynamic workload broker.

JobManagerGW.ini

In a distributed environment, if **-gateway** is set to `local`, then **ResourceAdvisorUrl** is the URL of the master or dynamic domain manager. The value is `https://$(tdwb_server):$(tdwb_port)/JobManagerRESTWeb/JobScheduler/resource`, where:

$(*tdwb_server*)

The fully qualified host name of the master or dynamic domain manager.

$(*tdwb_port*)

The port number of the master or dynamic domain manager.

ScannerPeriodSeconds

The time interval that the Resource advisor agent collects information about all the resources in the local system other than CPU resources. The default value is every 120 seconds.

The resource advisor agent, intermittently scans the resources of the machine (computer system, operating system, file systems and networks) and periodically sends an update of their status to the master or dynamic domain manager either in a z/OS or in a distributed environment.

The CPU is scanned every `CPUScannerPeriodSeconds` seconds, while all the other resources are scanned every `ScannerPeriodSeconds` seconds. As soon as one of the scans shows a significant change in the status of a resource, the resources are synchronized with the master in a distributed environment or the dynamic domain manager either in a z/OS or in a distributed environment. The following is the policy followed by the agent to tell if a resource attribute has significantly changed:

- A resource is added or deleted
- A string attribute changes its value
- A CPU value changes by more than `DeltaForCPU`
- A file system value changes by more than `DeltaForDiskMB` megabytes
- A Memory value changes by more than `DeltaForMemoryMB` megabytes

If there are no significant changes, the resources are synchronized with the HCL Workload Automation master in a distributed environment or with thedynamic domain manager either in a z/OS or in a distributed environment every `NotifyToResourceAdvisorPeriodSeconds` seconds.

Configuring properties of the System scanner [SystemScanner]

About this task

In the `JobManager.ini` file, the section containing the properties of the System scanner is named:

```
[SystemScanner]
```

You can change the following properties:

CPUSamples

The number of samples used to calculate the average CPU usage. The default value is 3.

DeltaForCPU

The change in CPU usage considered to be significant when it becomes higher than this percentage (for example, DeltaForCPU is 20 if the CPU usage changes from 10 percent to 30 percent). The default value is 20 percent.

DeltaForDiskMB

The change in use of all file system resources that is considered significant when it becomes higher than this value. The default value is 100 MB.

DeltaForMemoryMB

The change in use of all system memory that is considered significant when it becomes higher than this value. The default value is 100 MB.

Configuring environment variables [Env]

About this task

Add the section `[Env]` to the `JobManagerGW.ini` configuration file and insert the environment variables that you need in your dynamic scheduling environment.

Configuring the agent to work with a password vault

Configure dynamic agents to work with a password vault by creating a dedicated profile.

About this task

Enhance your password management by configuring dynamic agents to function as proxies to password vaults. You can use a password vault of your choice. If you prefer to use CyberArk, see Configuring the agent to work with CyberArk on page 91.

Perform the following steps:

1. **Prerequisistes check**:
    - Ensure all HCL Workload Automation components are at version 10.2.4 or later.
    - Ensure at least one profile is present on each agent acting as a proxy.
    - The profile is created automatically if you are upgrading from version 10.2.1 and later, otherwise it must be created manually.
    - In a z/OS environment, a dynamic domain manager is required.
    - If you are using pools, make sure all agents within the pool are correctly configured to integrate with CyberArk. Additionally, verify that all profiles on all agents are identical.
2. **Create a Profile**: On the agent, create a profile located in `/home/TWA_DATA_DIR/integrations/vault-profiles` using a flat-text editor. If you name the profile `default`, it is selected automatically if no profile is specified in the job definition. You can create multiple profiles for the same password vault or for different password vaults to meet different requirements.
3. **Specify Parameters**: The profile must contain the following parameters:

```
[VaultProfile.Common]
Type =
Description =
PasswordSolver =
ConfigFile =
```

where

Type

Specify the type of password vault to be used. This parameter is a string, and no validations are performed on its contents.

Description

Optionally write a description for the profile.

PasswordSolver

Contains the path to the password vault libraries. Alternatively, you can specify the absolute path to a script that retrieves the desired password from the password vault of your choice. This absolute path is consistent and independent of the current working directory, no matter where the file is located within the agent's file system.

If you plan to use a script to integrate with a password vault rather than a library, you have to write a dedicated script and ensure it returns a string containing the value of the password to be used in the job.

ConfigFile

Specify the name and path of the configuration file for the password vault. Alternatively, you can insert the whole configuration file directly in this parameter.

4. **Job definition creation**: Specify the desired profile when creating the job definition. If you do not specify a profile in the job definition, the `default` profile is used. For more information about creating the job definition so that the password is retrieved from a password vault, see the topic about obtaining passwords from password vaults in *User's Guide and Reference*.

Configuring the agent to work with CyberArk

Configure dynamic agents to work with CyberArk by creating a dedicated profile.

About this task

If you plan to use CyberArk as your password vault, configure the profile as follows:

```
[VaultProfile.Common]
Type = CyberArk
Description =
PasswordSolver = installation_dir/TWS/integrations/bin/libCyberArkVault.so
ConfigFile = TWA_DATA_DIR/integrations/config
```

Type

Specify the type of password vault to be used. This parameter is a string, and no validations are performed on its contents.

Description

Optionally write a description for the profile.

PasswordSolver

Contains the path to the password vault libraries. For example, you can point to the CyberArk libraries installed by default with the agent in the following paths:

On Windows operating systems

installation_dir\TWS\integrations\bin\CyberArkVault.dll

On UNIX operating systems

installation_dir/TWS/integrations/bin/libCyberArkVault.so

ConfigFile

Specify the name and full path of the configuration file for the password vault. If you have a `CyberArk.ini` configuration file from a previous installation, merge the contents of the file with the new configuration file. Alternatively, you can insert the whole configuration file directly in this parameter, for example by copying the whole `CyberArk.ini` file in this parameter.

For information about configuring secure communication with CyberArk, see Configuring secure communication with CyberArk on page 97.

What to do next

After the agent is configured, you can proceed to define a job that is designed to securely retrieve passwords from the password vault, as described in the topic about obtaining passwords from password vaults in *User's Guide and Reference*.

Defining parameters in the `CyberArk.ini` file

About this task

Set up the `CyberArk.ini` file to configure password retrieval.

To configure CyberArk, you can use the `CyberArk.ini` template file available in `TWS/integrations/config_templ`, as follows:

1. Create a copy of the template file to one of the following paths, depending on your operating system:

   On Windows operating systems

   installation_directory\integrations\config

   On UNIX operating systems

   TWA_DATA_DIR/integrations/config

2. On UNIX operating systems, ensure you apply to the new file the same permissions and ownership settings assigned to the `JobManager.ini` file.
3. Fill in the parameters listed below as required.

You can also copy the updated file to a different path and specify the full path in the **ConfigFile** parameter of the profile you plan to use.

In the `CyberArk.ini` file, the following sections and parameters are available:

[CyberArk.Config]

CPAccessLibrary

The full path to the CyberArk proprietary library file.

HandlePasswordChangeInProcess

The operation to be performed if another user changes the password while you are requesting it from CyberArk. Supported values are `true` and `false`.

If you set this property to `true`, the job remains in waiting status and password retrieval is attempted again, based on the values set for the **RetryIntervalForPasswordChangeInProcess** and **RetryAttemptsForPasswordChangeInProcess** parameters.

RetryIntervalForPasswordChangeInProcess

The time interval in seconds HCL Workload Automation waits before sending a new password request to CyberArk.

RetryAttemptsForPasswordChangeInProcess

The number of times HCL Workload Automation retries to obtain the password from CyberArk. If the specified number of retries is exceeded, the operation fails.

[CyberArk.CP.Connection]

This section applies only when you use the **Credential Provider** (**CP**) and specify the full path to the CyberArk library file in the **CPAccessLibrary** property.

Port

The port that is used to connect to the CP.

ConnectionTimeout

The time interval in seconds HCL Workload Automation waits for the host to answer the connection request.

[CyberArk.CCP.Connection]

This section applies only when you use the **Central Credential Provider** (**CCP**) and is used when you leave the **CPAccessLibrary** property empty. These properties are mandatory.

Host

The host name of the workstation where CyberArk Central Credential Provider is installed.

Protocol

The protocol used to connect to the host.

Port

The port used to connect to the host.

Path

The path where the REST API is located.

ConnectionTimeout

The time interval in seconds HCL Workload Automation waits for the host to answer the connection request.

Timeout

The time interval in seconds HCL Workload Automation waits for CyberArk to return the password.

FollowLocation

Set this property to `true` to enable the HTTP protocol. You cannot enable the HTTP protocol from the command line. This property instructs the composer command to follow any **Location: header** that the server sends as part of the HTTP header in a 3xx response. The **Location: header** can specify a relative or an absolute URL to follow.

SSLVerifyServer

Specify `yes` if server authentication is to be used in SSL communications.

Proxy

The name of the proxy server used when connecting to the specified host.

ProxyPort

The TCP/IP port number of the proxy server used when connecting to the specified host.

SSLVersion

Specify the SSL version to be used. Supported values are:

- **atleast.TLSv1.0**
- **atleast.TLSv1.1**
- **atleast.TLSv1.2**
- **atleast.TLSv1.3**

where you specify the minimum version of the TLS protocol to be used. In this case, HCL Workload Automation uses the specified version of the protocol or a higher version, if supported.

- **max.TLSv1.0**
- **max.TLSv1.1**
- **max.TLSv1.2**
- **max.TLSv1.3**

where you specify the maximum version of the TLS protocol to be used. In this case, HCL Workload Automation uses the specified version of the protocol or a lower version.

- **TLSv1.0**
- **TLSv1.1**
- **TLSv1.2**
- **TLSv1.3**

where you specify the exact version of the TLS protocol to be used. In this case, HCL Workload Automation uses the specified version of the protocol.

SSLCiphers

Define the ciphers that the workstation supports during an SSL connection.

If you want to use an OpenSSL cipher class, use the following command to find out the list of available classes:

```
openssl ciphers
```

For a full list of supported ciphers, see SSL Ciphers and OpenSSL.

SSLCipherSuites

Specify one or more supported algorithms for TLS version 1.3, This option does not apply to TLS version 1.2 or earlier.

SSLConfigFile

Specify the name and path of the OpenSSL configuration file. See OpenSSL documentation for details about the file format and options. If you modify this file, ensure the changes are consistent with the security configuration in your environment.

[CyberArk.CCP.Connection.OpenSSL]

For more information about configuring secure communication with CyberArk, see Configuring secure communication with CyberArk on page 97.

SSLKey

The full path to the private key file in **pem** format.

If you use certificates in pem format

Specify the full path to the private key file in **pem** format. For example, if you use certman to generate the certificates, specify in this parameter the full path to the `tls.key` file.

If you use certificates in p12 format

Leave this parameter blank.

SSLKeyPwd

The full path to the file containing the password encoded in Base64 for the private key.

SSLCertificate

Specify the full path to the local certificate file used in SSL communication. You can either use the certificates available on the agent or generate brand new certificates using the certman command, as follows:

you use the certificates available on the agent

Specify in the **SSL certificate** parameter the full path to the certificates, for example `/<TWS_DATA_DIR>/ssl/certs/TWSClientKeyStore.p12`. This ensures secure communication without further steps.

you generate new certificates using certman

1. Generate the certificates using certman, as described in Managing certificates using Certman on page 314
2. Set the **SSL cert type** parameter to **pem**, which is the format used by certman.
3. Specify in the **SSL certificate** parameter the full path to the tls.crt file generated by the certman command.
4. Specify in the **SSLKey** parameter the full path to the private key file.

SSLCertificateType

Specify the type of your private key and certificate file used in SSL communication. Supported formats are **p12** and **pem** .

If the certificate type is in pem format

- Specify the full path to the private key file in the **SSLKey** parameter.
- Specify the full path to the local certificate file in the **SSLCertificate** parameter.

If the certificate type is in p12 format

- Store both private key and certificate in the **p12**.
- Leave the **SSLKey** parameter blank.
- Specify the full path to the local certificate file in the **SSLCertificate** parameter.

SSLCACertificate

Specify the name of the file containing the trusted certification authority (CA) certificates required for SSL authentication. The CAs in this file are also used to build the list of acceptable client CAs passed to the client when the server side of the connection requests a client certificate. This file is the concatenation, in order of preference, of the various PEM-encoded CA certificate files.

SSLRandomSeed

Specify the pseudo random number file used by OpenSSL on some operating systems. Without this file, SSL authentication might not work correctly.

[CyberArk.AppDescs]

AppID

The unique ID of the application issuing the password request. This parameter is required.

[CyberArk.Query]

Safe

The name of the Safe where the password is stored.

Folder

The name of the folder where the password is stored.

Object

The name of the password object to retrieve.

Username

Defines search criteria according to the **UserName** account property.

Address

Defines search criteria according to the **Address** account property.

PolicyID

Defines the format that will be used in the **setPolicyID** method.

Database

Defines search criteria according to the **Database** account property.

[CyberArk.Query.Result]

NormalizedUsername

Standardized format of a user name.

What to do next

After the agent is configured, you can proceed to define a job that is designed to securely retrieve passwords from the password vault, as described in the topic about obtaining passwords from password vaults in *User's Guide and Reference*.

Configuring secure communication with CyberArk

About this task

To establish secure communication, you can use several certificate formats. The required configuration varies depending on the format you use, as follows:

If the certificate type is in pem format

- Specify the full path to the private key file in the **SSLKey** parameter.
- Specify the full path to the local certificate file in the **SSLCertificate** parameter.

If the certificate type is in p12 format

- Store both private key and certificate in the **p12**.
- Leave the **SSLKey** parameter blank.
- Specify the full path to the local certificate file in the **SSLCertificate** parameter.

When establishing secure communication with CyberArk, you can encounter one of the following scenarios:

You want to use your own certificates and CA

The following steps apply:

1. Provide CyberArk with your CA, which validates your certificate.
2. CyberArk returns its CA, which validates the certificate from CyberArk.
3. Depending on whether you use certificates in **pem** or **p12** format, specify the following parameters:

   certificates in pem format

   Specify the full path to the private key file in the **SSLKey** and the full path to the local certificate in the **SSLCertificate** parameters.

   certificates in p12 format

   a. Add private key and certificate into a **p12** keystore.
   b. Specify the full path to the **p12** keystore you just created in the **SSLCertificate** parameter.
   c. Leave the **SSLKey** parameter blank.
4. Import the CyberArk CA into the **pem** truststore which must be specified in the **SSLCACertificate** parameter in the `CyberArk.ini` file.

You request the certificates from CyberArk

The following steps apply:

1. CyberArk provides you with private key and certificate.
2. Depending on whether you use certificates in **pem** or **p12** format, specify the following parameters:

   certificates in pem format

   Specify the full path to the key in the **SSLKey** and the full path to the certificate in the **SSLCertificate** parameters.

   certificates in p12 format

   a. Add private key and certificate into a **p12** keystore.
   b. Specify the full path to the **p12** keystore you just created in the **SSLCertificate** parameter.
   c. Leave the **SSLKey** parameter blank.
3. Import the CyberArk CA into the **pem** truststore which must be specified in the **SSLCACertificate** parameter in the `CyberArk.ini` file.

Configuring the agent to work with Kerberos

With the Kerberos integration, you can communicate securely over an insecure network by leveraging the Kerberos Authentication Protocol for submitting jobs on dynamic agents.

To configure Kerberos on 64-bit Linux Intel (linux-x86_64) operating systems, you can use the `Kerberos.ini` template file available on dynamic agents in `TWS/integrations/config_templ`, as follows:

1. Create a copy of the template file to one of the following paths, depending on your operating system:

   On Windows operating systems

   installation_directory`\ITA\cpa\config`

   On UNIX operating systems

   TWA_DATA_DIR`/ITA/cpa/config`

2. Ensure you apply to the new file the same permissions and ownership settings assigned to the `JobManager.ini` file.
3. Fill in the parameters listed in Configuring the `Kerberos.ini` file on page 99 as required.
4. Browse to the `JobManager.ini` file, located in one of the following paths, depending on your operating system:

   On Windows operating systems

   installation_directory`\ITA\cpa\config`

   On UNIX operating systems

   TWA_DATA_DIR`/ITA/cpa/config`

5. Add the following keys to the **NativeJobLauncher** section in the `JobManager.ini` file:

   AuthMethod

   The full path to the `libKerberos.so` library file, as follows:*`installation_directory`*`/TWS/bin/libKerberos.so`

   IsAuthMethodMandatory

   The behavior in case the authentication fails. The default value is **false**: if Kerberos authentication fails, the job continues with the authentication methods provided by the service in use, for example, by requesting the user and password required by SSH. If you set this key to **true** and Kerberos authentication fails, the job fails.

6. Start all processes on the dynamic agent by running the StartUpLwa command.

Configuring the `Kerberos.ini` file

You can configure the following properties in the `Kerberos.ini` file:

Kerberos.Config section

UseDefaultCCache

The credentials cache to be used for storing intermediate objects. The default value is `false`: a cache file is automatically assigned by the `libKerberos.so` library for each job. If you set it to `true`, Kerberos defines the cache location.

KDCConnectionRetryAttempts

The number of times HCL Workload Automation retries to authenticate with Kerberos, in case the first attempt fails. The default value is `0`, which means the integrations tries to authenticate a single time and performs no further attempts.

KDCConnectionRetryInterval

The time interval in seconds HCL Workload Automation waits before sending a new authentication request to Kerberos. The default value is `5` seconds.

Kerberos.InitCredsOpts section

The following properties are internal Kerberos properties. If you specify a value, it overrides the corresponding setting on Kerberos. If you leave the property blank, the value defined on Kerberos applies. For more information about these properties, see Kerberos documentation.

Proxiable

Whether credentials should be proxiable.

Forwardable

Whether the credentials should be forwardable.

TicketLifetime

The default lifetime for initial ticket requests.

Kerberos.Logging.cclog section

Most of the properties in this section are reserved for internal use and should not be changed. You can configure the following properties:

Kerberos.trfl.level

Determines the type of trace messages that are logged. Change this value to trace more or fewer events, as appropriate, or on request from Software Support. The default value is `3000`, which means minimum trace information is captured. To enable maximum level of tracing, set this property to `1000`.

Kerberos.trhd.maxFileBytes

The maximum size that the trace file can reach. The default value is `10240000` bytes.

Kerberos.trhd.maxFiles

The maximum number of trace files that can be stored. The default value is `5`.

User management

The integration supports two authentication modes:

- You can specify the same user for authenticating to Kerberos and running the job.
- You can specify a user for authenticating to Kerberos and a different user for running the job. In this case, when you create the job definition from the **Workload Designer** or **Graphical Designer**, specify both users in the **Credentials** tab of the job definition with the following syntax:

  ```
  job_user/kerberos_user
  ```

  where

  job_user

  Is the user running the job

  kerberos_user

  Is the user authenticating to Kerberos

What to do next

After configuring Kerberos, you can proceed to create job definitions as usual. When you specify a user in the job definition, the Kerberos Authentication Protocol is applied.

The job can run only on the dynamic agent on which you have configured Kerberos.

Regular maintenance

Regular maintenance refers to the mechanisms that are used on your dynamic workstation agents to free up storage space and improve performance.

You can have regular maintenance performed on your dynamic agent workstations to keep disk space under control by configuring the following parameters as appropriate.

Table 18. Agent configuration parameters

File	Parameter	Description
JobManager.ini located in the path **On UNIX™ operating systems** *TWA_DATA_DIR*/`ITA/cpa/config/JobManager.ini`	MaxAge	The number of days that job logs are kept before being deleted. The default is 2. Possible values range from a minimum of 1 day.
	JobManager.loggerhd.maxFileBytes	The maximum size that the log file can reach. The default is 1024000 bytes.

Table 18. Agent configuration parameters

(continued)

File	Parameter	Description
On Windows™ operating systems *TWA_home*`\TWS\ITA\cpa\config\JobManager.ini`	JobManager.loggerhd.maxFiles	The maximum number of log files that can be stored in the `stdlist/JM` directory. The default is 3.
	JobManager.ffdc.maxDiskSpace	The maximum disk space reached, by the log files collected by the First Failure Data Capture tool, after which the oldest files are removed.
	JobManager.trhd.maxFileBytes	The maximum size that the log file can reach. The default is 10240000 bytes.
	JobManager.trhd.maxFiles	The maximum number of log files that can be stored. The default is 5.
`logging.properties` located in the path **On UNIX™ operating systems** *TWA_DATA_DIR*`/TWS/JavaExt/cfg/` **On Windows™ operating systems** *TWA_home*`\TWS\JavaExt\cfg` Logs related to jobs with advanced options.	java.util.logging.FileHandler.limit	The maximum amount to write log messages to a file. Default value is 1000000 (bytes)
	java.util.logging.FileHandler.count	The number of output files to cycle through. Default value is 10.

Configuring the dynamic workload broker server on the master domain manager and dynamic domain manager

About this task

You can perform these configuration tasks after completing the installation of your master domain manager, dynamic domain manager, and dynamic agents, and any time that you want to change or tune specific parameters in your environment.

The configuration parameters for the dynamic workload broker server are defined by default at installation time. You modify a subset of these parameters using the files that are created when you install dynamic workload broker. The following files are created in the path:

On Windows systems

<*TWA_home*>`\broker\config`

On UNIX systems

<*TWA_DATA_DIR*>`/broker/config`

ResourceAdvisorConfig.properties

Contains configuration information about the **Resource Advisor**. For more information, see ResourceAdvisorConfig.properties file on page 105.

JobDispatcherConfig.properties

Contains configuration information about the **Job Dispatcher**. For more information, see JobDispatcherConfig.properties file on page 107.

BrokerWorkstation.properties

Contains configuration information about the broker server. BrokerWorkstation.properties file on page 109

CLIConfig.properties

Contains configuration information for the dynamic workload broker command line. This file is described in the section about Command-line configuration file in *User's Guide and Reference*.

You can modify a subset of the parameters in these files to change the following settings:

- Heartbeat signal from the agents.
- Time interval for job allocation to resources
- Time interval for notifications on resources
- Polling time when checking the status of remote engine workstations
- Maximum number of results when allocating jobs to global resources
- Encryption of any passwords sent in the JSDL definitions
- Time interval for retrying the operation after a **Job Dispatcher** failure
- Time interval for retrying the operation after a client notification failure
- Archive settings for job data
- Job history settings
- Command line properties

The editable parameters are listed in the following sections. If you edit any parameters that are not listed, the product might not work. After modifying the files, you must stop and restart WebSphere Application Server Liberty.

Maintaining the dynamic workload broker server on the master domain manager and dynamic domain manager

About this task

Because one dynamic workload broker server is installed with your master domain manager and dynamic domain manager, and one server with every backup manager, you have at least two servers present in your HCL Workload Automation network. The server running with the master domain manager is the only one active at any time. The servers installed in the backup managers are idle until you switch managers, and the server in the new manager becomes the active server. To have a smooth transition from one server to another, when you switch managers, it is important that you keep the same configuration settings in the `ResourceAdvisorConfig.properties` and `JobDispatcherConfig.properties` files in all your servers. When you make a change in any of these files of your running dynamic workload broker server, remember to apply the same change also in the dynamic workload broker server idling on your backup manager.

Some of the settings for the dynamic workload broker server are stored in the local **BrokerWorkstation.properties** file and also in the HCL Workload Automation database. When you switch to the backup master domain manager or dynamic domain manager, the dynamic workload broker server settings are automatically updated on the backup workstation. For more information about the **BrokerWorkstation.properties** file, see BrokerWorkstation.properties file on page 109.

Note: The database is automatically populated with the information from the active workstation, regardless of whether it is the manager or the backup workstation. For example, if you modify the dynamic workload broker server settings on the backup master domain manager or dynamic domain manager, this change is recorded in the database. When you switch back to the manager workstation, the change is applied to the master domain manager or dynamic domain manager and the related local settings are overwritten.

It is important that you also keep the data pertinent to every dynamic workload broker server up-to-date. If you change the host name or port number of any of your dynamic workload broker servers, use the `exportserverdata` and `importserverdata` commands from the dynamic workload broker command line to record these changes in the HCL Workload Automation database. For information about these commands, see *Scheduling Workload Dynamically*.

The database records for your workload broker workstations all have LOCALHOST as the host name of the workstation. Leave the record as-is. Do not replace LOCALHOST with the actual host name or IP address of the workstation. LOCALHOST is used intentionally to ensure that the jobs submitted from HCL Workload Automation are successfully sent to the new local dynamic workload broker when you switch the master domain manager or dynamic domain manager.

Enabling unsecure communication with the dynamic workload broker server

About this task

By default, the dynamic workload broker server uses secure communication. You might need to enable unsecure communication, even though this type of communication is not recommended.

To enable unsecure communication with the dynamic workload broker server, perform the following steps on the master domain manager:

1. Run the exportserverdata command located in *TWA_home*`/TDWB/bin`:

   ```
   exportserverdata -dbUsr db_instance_admin -dbPwd db_instance_admin_pwd
   ```

2. Open the resulting `server.properties` file in a flat-text editor.
3. Copy the following line:

   ```
   https://hostname:port/JobManagerRESTWeb/JobScheduler
   ```

4. Change the copied line by replacing **https** with **http**:

   ```
   http://hostname:port/JobManagerRESTWeb/JobScheduler
   ```

   The file now contains two lines specifying the connection mode, one line specifying the https mode and one line specifying the http mode.
5. Save the file.
6. Import the new data with the importserverdata command located in *TWA_home*/`TDWB/bin`:

   ```
   importserverdata -dbUsr db_instance_admin -dbPwd db_instance_admin_pwd
   ```

For more information about the exportserverdata and importserverdata commands, see *HCL Workload Automation: Scheduling Workload Dynamically*.

ResourceAdvisorConfig.properties file

The parameters in this file affect the following dynamic workload broker server settings:

- Heartbeat signal from the agents
- Time interval for job allocation to resources
- Time interval for notifications on resources
- Polling time when checking the status of remote engine workstations
- Maximum number of results when allocating jobs to global resources

You can modify the following parameters in the `ResourceAdvisorConfig.properties` file:

DatabaseCheckInterval

Specifies the time interval within which the dynamic workload broker server checks the availability of the database. The default value is **180** seconds.

ResourceAdvisorURL

Specifies the URL of the **Resource Advisor**.

RaaHeartBeatInterval

Specifies the time interval within which the **Resource Advisor** expects a heartbeat signal from the dynamic agent. The default value is **200** seconds. After the maximum number of retries (specified in the **MissedHeartBeatCount** parameter) is exceeded, the **Resource Advisor** reports the related computer as unavailable. In a slow network, you might want to set this parameter to a higher value. However, defining a higher value might delay the updates on the availability status of computer systems. If, instead, you decrease this value together with the value defined for the **NotifyToResourceAdvisorPeriodSeconds** parameter, this might generate network traffic and increase CPU usage when updating cached data. The value defined in this parameter must be consistent with the **NotifyToResourceAdvisorPeriodSeconds** parameter defined in the `JobManager.ini` file, which defines the time interval for each dynamic agent to send the heartbeat signal to the **Resource Advisor**.

MissedHeartBeatCount

Specifies the number of missed heartbeat signals after which the computer is listed as not available. The default value is 2. In a slow network, you might want to set this parameter to a higher value.

MaxWaitingTime

Specifies the maximum time interval that a job must wait for a resource to become available. If the interval expires before a resource becomes available, the job status changes to Resource Allocation Failure. The default value is 600 seconds. You can override this value for each specific job by using the **Maximum Resource Waiting Time** parameter defined in the Dynamic Workload Console. For more information about the **Maximum Resource Waiting Time** parameter, see the Dynamic Workload Console online help. If you set this parameter to -1, no waiting interval is applied for the jobs. If you set this parameter to 0, the **Resource Advisor** tries once to find the matching resources and, if it does not find any resource, the job changes to the ALLOCATION FAILED status. If you increase this value, all submitted jobs remain in WAITING status for a longer time and the **Resource Advisor** tries to find matching resources according to the value defined for the **CheckInterval** parameter.

CheckInterval

Specifies how long the **Resource Advisor** waits before retrying to find matching resources for a job that did not find any resource in the previous time slot. The default value is 60 seconds.

TimeSlotLength

Specifies the time slot interval during which the **Resource Advisor** allocates resources to each job. Jobs submitted after this interval has expired are considered in a new time slot. The default value is 15 seconds. The default value is adequate for most environments and should not be modified. Setting this parameter to a higher value, causes the **Resource Advisor** to assign resources to higher priority jobs rather than to lower priority jobs when all jobs are trying to obtain the same resource. It might also, however, cause the job resource matching processing to take longer and the resource state updates from agents to be slowed down. Setting this parameter to a lower value, causes the **Resource Advisor** to process the resource matching faster and, if you have a high number of agents with frequent updates, to update the resource repository immediately. If job requirements match many resources, lower values ensure a better load balancing. If most jobs use resource allocation, do not lower this value because the allocation evaluation requires many processing resources.

NotifyTimeInterval

Specifies the interval within which the **Resource Advisor** retries to send notifications on the job status to the **Job Dispatcher** after a notification failed. The default value is 15 seconds. The default value is adequate for most environments and should not be modified.

MaxNotificationCount

Specifies the maximum number of attempts for the **Resource Advisor** to send notifications to the **Job Dispatcher**. The default value is 100. The default value is adequate for most environments and should not be modified.

ServersCacheRefreshInterval

Specifies with what frequency (in seconds) the Resource Advisor checks the list of active and backup dynamic workload broker servers for updates. This list is initially created when the master domain manager is installed, and after that it is updated every time a new backup master is installed and connected to the master domain manager database (the master domain manager and every backup master include also a dynamic workload broker server). When the Resource Advisor agents send their data about the resources discovered in each computer, they are able to automatically switch between the servers of this list, so that the dynamic workload broker server that is currently active can store this data in its Resource Repository. For this reason, the Resource Advisor agents must know at all times the list of all dynamic workload broker servers. The possible values range between 300 (5 minutes) and 43200 (12 hours). The default value is 600 seconds.

StatusCheckInterval

Specifies the time interval in seconds the Resource Advisor waits before polling for the status of a resource. For example this timeout applies when checking the status of a remote engine. The default value is 120 seconds.

After modifying the file, you must stop and restart WebSphere Application Server Liberty.

JobDispatcherConfig.properties file

The parameters in this file affect the following settings for the dynamic workload broker server installed on a master domain manager or dynamic domain manager:

- Encryption of any passwords sent in the JSDL definitions
- Time interval for retrying the operation after a **Job Dispatcher** failure
- Time interval for retrying the operation after a client notification failure
- Archive settings for job data
- Job history settings
- Gateways and dynamic workload broker server connection settings.

After modifying the file, you must stop and restart the IBM® WebSphere® server.

During the upgrade from version 8.5.1 the values you set for the properties for version 8.5.1 are preserved. The default values for the properties for version 8.6 are different from those in version 8.5.1. If you want to use the version 8.6 defaults, change them manually.

In the `JobDispatcherConfig.properties` file, the following parameters are available:

DatabaseCheckInterval

Specifies the time interval within which the dynamic workload broker server checks the availability of the database. The default value is **180** seconds.

EnablePasswordEncryption

Specifies that any user passwords contained in the JSDL definitions are to be encrypted when the definitions are sent to the agents. The default is `true`. Setting this property to `false` forces the dynamic workload broker server to send the passwords in plain text. This applies to any password field.

RAEndpointAddress

Specifies the URL of the **Resource Advisor**.

JDURL

Specifies the URL of the **Job Dispatcher**.

FailQInterval

Specifies the numbers of seconds for retrying the operation after the following failures:

- Client notification.
- Allocation, Reallocate, Cancel Allocation requests to **Resource Advisor**.
- Any database operation failed for connectivity reasons.

The default value is 30 seconds. Increasing this value improves recovery speed after a failure but can use many system resources if the recovery operation is complex. For example, if the workload broker workstation is processing a new Symphony file, this operation might require some time, so you should set this parameter to a high value. If you are not using workload broker workstation, this parameter can be set to a lower value.

MaxCancelJobAttemptsCount

The maximum number of times the Job Dispatcher attempts to cancel a shadow job or a job running on a dynamic agent when a request to kill the job is made and the kill request cannot be immediately processed. The default is 1440 attempts. The Job Dispatcher attempts to cancel the job every 30 seconds for a maximum number of times specified by this parameter.

MaxNotificationCount

Specifies the maximum number of retries after a client notification failure. The default value is 1440. For example, if the workload broker workstation is processing a new Symphony file, this operation might require some time, so you should set this parameter to a high value. If you are not using the workload broker workstation, this parameter can be set to a lower value.

MoveHistoryDataFrequencyInMins

Specifies how often job data must be deleted. The unit of measurement is minutes. The default value is 60 minutes. Increasing this value causes the **Job Dispatcher** to check less frequently for jobs to be deleted. Therefore, the volume of jobs in the **Job Repository** might increase and all queries might take longer to complete. Dynamic workload broker servers with high job throughput might require lower values, while low job throughputs might require higher values.

SuccessfulJobsMaxAge

Specifies how long successfully completed or canceled jobs must be kept in the **Job Repository** database before being archived. The unit of measurement is hours. The default value is 240 hours, that is ten days.

UnsuccessfulJobsMaxAge

Specifies how long unsuccessfully completed jobs or jobs in unknown status must be kept in the **Job Repository** database before being archived. The unit of measurement is hours. The default value is 720 hours, that is 30 days.

AgentConnectTimeout

Specifies the number of minutes that the dynamic workload broker server waits for a scheduling agent response after it first attempts to establish a connection with that agent. If the agent does not reply within this time, the server does not open the connection. Values range from 0 to 60 (use 0 to wait indefinitely). The default is 3.

AgentReadTimeout

Specifies the number of minutes that the dynamic workload broker server waits to receive data from established connections with a scheduling agent or a gateway. If no data arrives within the specified time, the server closes the connection with the agent. Values range from 0 to 60 (use 0 to wait indefinitely). The default is 3.

GatewayPollingTimeout

Add this parameter to specify the number of minutes that the gateway waits to receive data from established connections with a dynamic workload broker. If no data arrives within the specified time, the gateway closes the connection with the dynamic workload broker. Values range from 1 to 60. The default is 1 minute.

GatewayConnectionTimeout

Add this parameter to specify the number of seconds that the dynamic workload broker server waits for a gateway receiving data after the dynamic workload broker first attempts to send data to the gateway. If the gateway does not reply within this time, the dynamic workload broker does not open the connection. Values range from 1 to 60. The default is 10 seconds.

MaxNumberOfParallelGateways

Add this parameter to specify the number of gateways that dynamic workload broker server can manage without lack of performances. Values range from 3 to 100. The default is 3.

Note:

If an unexpected job workload peak occurs and a cleanup of the database is required earlier than the value you specified in the `MoveHistoryDataFrequencyInMins` parameter, you can use the movehistorydata command to perform a cleanup before the scheduled cleanup is performed.

BrokerWorkstation.properties file

If you need to make configuration changes to the broker server after the installation has completed, you can edit the `BrokerWorkstation.properties` file. The `BrokerWorkstation.properties` file contains the following configuration properties:

Broker.AuthorizedCNs

The list of prefixes of common names authorized to communicate with the broker server.

Broker.CertificateExpirationInterval

The number of days before the certificate on the agent expires. During this interval, the certificate is set in expiring status and the agent tries to download a new version of the certificate from the master domain manager, if available. Supported values are any integer greater than zero. The default value is 15 days.

Broker.fileproxy.urls

If you defined a stand-alone file proxy using the fileproxyinst command, you can optionally use this property to specify the URL of an alternate stand-alone file proxy for high availability configuration. You can implement one of the following configurations:

- Specify one or more proxy servers, separated by commas. If one of the file proxies stops unexpectedly, the other file proxy takes over.
- Specify a load balancer. If one of the file proxies stops unexpectedly, the load balancer switches the workload to one of the available file proxies.

Use the following syntax:

```
[http/https]://[fileproxy host]:[file proxy port]
```

This procedure does not apply if you use the default file proxy installed with each master domain manager.

For more information about the File Transfer integration, see the File Transfer integration on Automation Hub.

For more information about installing the file proxy as a stand-alone component, see the section about the fileproxyinst script in *HCL Workload Automation: Planning and Installation*.

Broker.jobLog.corePoolSize

The size of the thread pool that serves core services and job log requests. The maximum supported number of concurrent job log requests corresponds to the value set for **Broker.jobLog.corePoolSize** minus 5. For example, if **Broker.jobLog.corePoolSize** is set to 20, the maximum supported number of concurrent job log requests is 15. Issuing a number of concurrent job log requests higher than this value (15 in the example) causes the dynamic domain manager to hang and requires a restart of the application server. The default value, when the property is not specified in the `BrokerWorkstation.properties` file or empty, is 50.

Broker.Workstation.CpuType

The workstation type assigned to the broker server. Supported values are:

- master domain manager (master)
- backup master domain manager (fta)
- dynamic domain manager (fta, broker, agent)
- backup dynamic domain manager (fta, broker, agent)

Broker.Workstation.Enable

A switch that enables or disables the broker server. The value can be `true` or `false`. The default value is `true`.

Set this value to `false` if you decide not to use a broker server. Not using the broker server means that you can submit jobs dynamically on the dynamic workload broker directly (using either the Dynamic Workload Console or the dynamic workload broker command line) without using the scheduling facilities of HCL Workload Automation.

Broker.Workstation.Name

The name of the broker server in the HCL Workload Automation production plan. This name is first assigned at installation time.

Broker.Workstation.PortSSL

The port used by the broker server to listen to the incoming traffic (equivalent to the Netman port) in SSL mode. It is first assigned at installation time. This port number must always be the same for all the broker servers that you define in your HCL Workload Automation network (one with the master domain manager and one with every backup master you install) to ensure consistency when you switch masters.

Broker.Workstation.Port

The port used by the broker server to listen to the incoming traffic (equivalent to the Netman port). It is first assigned at installation time. This port number must always be the same for all the broker servers that you define in your HCL Workload Automation network (one with the master domain manager and one with every backup master you install) to ensure consistency when you switch masters.

Broker.Workstation.RetryLink

The number of seconds between consecutive attempts to link to the broker server. The default is 600.

DomainManager.Workstation.Domain

The name of the domain where the broker server is registered.

DomainManager.Workstation.Name

The name of the domain manager workstation.

DomainManager.Workstation.Port

The port of the domain manager workstation.

MasterDomainManager.HostName

The host name of the master domain manager workstation.

MasterDomainManager.HttpsPort

The HTTPS port of the master domain manager workstation.

MasterDomainManager.Name

The name of the master domain manager workstation.

If you need to modify the event processor server, for example to use a load balancer, add the following two keywords in the file:

Broker.Workstation.evtproc.*previous_hostname*=*new_hostname*

Specify the previous hostname and the new hostname of the event processor.

Broker.Workstation.evtproc.*previous_port*=*new_port*

Specify the previous and new port of the event processor.

After stopping and restarting WebSphere Application Server Liberty, the dynamic domain manager sends the updated information to the dynamic agents.

Archiving job data

Job definitions created using the Dynamic Workload Console are stored in the **Job Repository** database. The **Job Repository** database stores also the jobs created when the job definitions are submitted to the dynamic workload broker.

Job information is archived on a regular basis. By default, successful jobs are archived every 24 hours. Jobs in failed or unknown status are archived by default every 72 hours.

You can configure the time interval after which job data is archived using the following parameters:

- **MoveHistoryDataFrequencyInMins**
- **SuccessfulJobsMaxAge**
- **UnsuccessfulJobsMaxAge**

These parameters are available in the `JobDispatcherConfig.properties` file, as described in JobDispatcherConfig.properties file on page 107. You can also use the movehistorydata command to perform a cleanup before the scheduled cleanup is performed.

Configuring to schedule J2EE jobs

About this task

Using the dynamic workload broker component you can schedule J2EE jobs. To do this you must complete the following configuration tasks:

- Configure the J2EE executor on page 112 on every agent on which you submit J2EE jobs.
- Configure the J2EE Job Executor Agent on page 117 on an external WebSphere Application Server

Configuring the J2EE executor

About this task

To dynamically schedule J2EE jobs, you must configure the following property files on every agent on which you submit J2EE jobs:

- J2EEJobExecutorConfig.properties
- logging.properties
- soap.client.props

These files are configured with default values at installation time. The values that you can customize are indicated within the description of each file.

J2EEJobExecutorConfig.properties file

Use the `J2EEJobExecutorConfig.properties` file to configure the J2EE executor

The file is located in:

On Windows operating systems

TWA_home>`\JavaExt`*version_number*`>\cfg`

On UNIX operating systems

TWA_DATA_DIR>`/JavaExt/cfg`

The keywords of this file are described in the following table:

Table 19. J2EEJobExecutorConfig.properties file keywords

Table 19. J2EEJobExecutorConfig.properties file keywords (continued)

Keyword	Specifies...	Default value	Must be customized
	means that dynamic workload broker uses an existing WebSphere Application Server scheduling infrastructure that is already configured on a target external WebSphere Application Server.	- `indirect` keyword - Name of the scheduler: `sch/MyScheduler` - `soap` keyword - Host name of the external WebSphere Application Server instance: `washost.mydomain.com` - SOAP port of the WebSphere Application Server instance: `8880` - Path to the `soap.client.props` file: *TWA_home*`/TWS/JavaExt/cfg/soap.client.props` - Credentials keyword: `mycred`	- The scheduler name. Replace the `sch/MyScheduler` string with the JNDI name of the IBM® WebSphere® scheduler that you plan to use. - The host name of the external WebSphere Application Server instance. - The SOAP port of the external WebSphere Application Server instance.
connector.direct	The name of the direct communication channel without using the WebSphere Application Server scheduler. Select a direct invoker to have dynamic workload brokerimmediately forward the job to the external WebSphere Application Server instance components (EJB or JMS). When creating the job definition, you can specify if you want to use a direct or indirect connector in the **J2EE Application** pane in the **Application** page in the Job Brokering Definition Console, or in the **invoker** element in the JSDL file. For more	A single line with the following values separated by commas: - `direct` keyword - The following string: `com.ibm.websphere.naming.WsnInitialContextFactory` - The following string: `corbaloc:iiop:washost.mydomain.com:2809`	You must customize the following: - The host name of the external WebSphere Application Server instance: `washost.mydomain.com` - The RMI port of the external WebSphere Application Server instance: `2809`

Table 19. J2EEJobExecutorConfig.properties file keywords (continued)

Keyword	Specifies...	Default value	Must be customized
	information about the Job Brokering Definition Console, see the online help.		
trustStore.path	The path to the WebSphere Application Server trustStore file (this file must be copied to this local path from the WebSphere Application Server instance).	*TWA_home*/TWS/JavaExt/cfg/DummyClientTrustFile	You can change the path (*TWA_home*/TWS/JavaExt/cfg), if you copy the trustStore path from the external WebSphere Application Server to this path.
trustStore.password	The password for the WebSphere Application Server trustStore file.	WebAs	Yes

The logging.properties file

About this task

The path to this file is `TWA_home/TWS/JavaExt/cfg/logging.properties` (`TWA_home\TWS\JavaExt\cfg\logging.properties`) on the agent.

After installation, this file is as follows:

```
# Set the default logging level for the logger named com.mycompany
com.ibm.scheduling = INFO
```

You can customize:

- The logging level (from INFO to WARNING, ERROR, or ALL) in the following keywords:

  `.level`

  Defines the logging level for the internal logger.

  `com.ibm.scheduling`

  Defines the logging level for the job types with advanced options. To log information about job types with advanced options, set this keyword to ALL.

- The path where the logs are written, specified by the following keyword:

  ```
  java.util.logging.FileHandler.pattern
  ```

The soap.client.props file

About this task

The path to this file is as follows:

On Windows operating systems

<*TWA_home*>`\TWS\JavaExt\cfg\soap.client.props`

On UNIX operating systems

<*TWA_DATA_DIR*>`/JavaExt/cfg/soap.client.props`

After installation, this file is as follows:

```
#   prompt: GUI dialog box; falls back to stdin if GUI not allowed
#
#   (So to disable auto prompting, set loginSource to nothing)
#------------------------------------------------------------------------------
com.ibm.SOAP.loginSource=prompt

#------------------------------------------------------------------------------
# SOAP Request Timeout
#
# - timeout (specified in seconds [default 180], 0 implies no timeout)
#
#------------------------------------------------------------------------------
com.ibm.SOAP.requestTimeout=180

#------------------------------------------------------------------------------
# SSL configuration alias referenced in ssl.client.props
#------------------------------------------------------------------------------
com.ibm.ssl.alias=DefaultSSLSettings
```

If you want to enable SOAP client security, you must:

1. Change `com.ibm.SOAP.securityEnabled` to `true`
2. Customize:
   - `com.ibm.SOAP.loginUserid` with the true WebSphere Application Server Liberty administrator user ID.
   - `com.ibm.SOAP.loginPassword` with the true WebSphere Application Server Liberty administrator password in {xor} encrypted format.

Configuring the J2EE Job Executor Agent

About this task

To set up the environment on the external WebSphere Application Server, Version 7.0 for the J2EE Job Executor Agent, do the following:

Create a Service Integration Bus

1. Open the WebSphere® Administrative Console (for example, `http://localhost:9060/admin`, depending on the admin port you configured).
2. Expand **Service Integration** and select **Buses**. The Buses window is displayed.
3. Click **New** to display the Buses configuration window.
4. Type a name for the new bus, for example **MyBus** and click **Next** and then **Finish** to confirm.
5. Click the MyBus name and the MyBus properties are displayed.
6. Under Topology, click **Bus Members**. The `Buses→MyBus→Bus` members window is displayed.
7. Click **Add**, select the **Server** radio button, choose **your_application_server_name**, click **Next**, and then click **Finish**.
8. When the `Confirm the addition of a new bus member` panel is displayed, click **Finish**.
9. Select **Service Integration → Buses → MyBus → Destinations → New**.
10. Select **Queue** as the type and click **Next**
11. Type **BusQueue** as the identifier and assign the queue to a bus member. Click **Next**. In the confirmation panel click **Finish**.

Configure the Default Messaging Service

1. From the left panel of the WebSphere® Administrative Console, expand **`Resources`→ `JMS`→ JMS Providers**, then click **Default messaging** at the server level as scope.
2. In the **Connection Factories** section, click **New**.
3. On the New JMS connection factory window, type in the following fields:

   Name

   MyCF

   JNDI name

   jms/MyCF

   Bus name

   MyBus

   Provider endpoints

   <hostname>:<Basic SIB port number>:BootstrapBasicMessaging;<hostname>:<Secure SIB port number>:BootstrapSecureMessaging
4. Select again **Resources –→ JMS-→ JMS Providers → Default Messaging** at the server level as scope, locate the section **Destinations**, and click **Queues**. Click **New** and type in the following fields as shown:

   Name=MyQueue
   JNDI name=jms/MyQueue
   Bus name=MyBus
   Queue name=BusQueue

   Click **Ok**.
5. Select again **Resources → JMS → JMS Providers → Default Messaging** at the server level as scope, and locate the section **Activation Specifications**.
6. Click **JMS activation specification**. Click **New** and type in the following fields as shown:

   Name=MyActSpec
   JNDI name=eis/MyActSpec
   Bus name=MyBus
   Destination type=Queue
   Destination JNDI name=jms/MyQueue

   Click **Ok**.

Configure the Java security

1. Select **Security → Secure Administration, applications and infrastructure**.
2. Locate the **Authentication** section, expand the **Java Authentication and Authorization Service**, and click **J2C authentication data**.
3. Click **New** and type in the following fields as shown:

Alias=*usr*

User ID=*usr*

Password=*pwd*

where *usr* is the user ID authenticated when using connector security and *pwd* is the related password.

4. Click **Ok**.

Create an XA DataSource

1. In the left pane, go to **Resources → `JDBC..`→ JDBCProviders**. In the resulting right pane, check that the scope is pointing to **your_application_server_name**.
2. Locate the **DERBY JDBC Provider (XA)** entry and click it.
3. Locate the **Additional Properties** section and click **Data Sources**.
4. Click **New** and type in the following fields as shown:

   Name = MyScheduler XA DataSource

   JNDI name = jdbc/SchedulerXADS

   Database name = ${USER_INSTALL_ROOT}/databases/Schedulers/${SERVER}/SchedulerDB;create=true
5. At the top of the page, click **Test connection button**.
6. Even if you get a negative result, modify the **Database name** field, deleting the part `;create=true`. Click **Ok**.

Create a WorkManager

1. In the left pane, go to **Resources → Asynchronous beans → Work managers** and click **New**.
2. type in the following fields as shown:

   Name=SchedulerWM

   JNDI name=wm/SchedulerWM
3. Click **Ok**.

Create and configure a scheduler

1. In the left pane, go to **Resources → Schedulers** and click **New**.
2. type in the following fields as shown:

   Name=MyScheduler

   JNDI name=sch/MyScheduler

   Data source JNDI name=jdbc/SchedulerXADS

   Table prefix=MYSCHED

   Work managers JNDI name=wm/SchedulerWM
3. Click **Ok**.
4. Select **MyScheduler** and click **Create tables**.
5. Deploy the test application.

Security order of precedence used for running J2EE tasks

There are three ways of verifying that a task runs with the correct user credentials. Tasks run with specified security credentials using the following methods:

1. Java™ Authentication and Authorization Service (JAAS) security context on the thread when the task was created.
2. `setAuthenticationAlias` method on the `TaskInfo` object.
3. A specified security identity on a `BeanTaskInfo` task `TaskHandler` EJB method.

The authentication methods are performed in the order listed above, so that if an authentication method succeeds, the following checks are ignored. This means that the *usr* and *pwd* credentials defined in **Configure the Java™ security** take precedence over any credentials specified in the tasks themselves.

Configuring to schedule job types with advanced options

About this task

You can define job types with advanced options by using the related configuration files. The options you define in the configuration files apply to all job types with advanced options of the same type. You can override these options when defining the job by using the Dynamic Workload Console or, if you are in a distributed environment, the **composer** command.

Configuration files are available on each dynamic agent in TWA_home/TWS/JavaExt/cfg for the following job types with advanced options:

Table 20. Configuration files for job types with advanced options

Job type	File name	Keyword
• Database job type • MSSQL Job	DatabaseJobExecutor.properties	Use the `jdbcDriversPath` keyword to specify the path to the JDBC drivers. Define the keyword so that it points to the JDBC jar files directory, for example: `jdbcDriversPath=c:\\mydir\\jars\\jdbc` The JDBC jar files must be located in the specified directory or its subdirectories. Ensure you have list permissions on the directory and its sub subdirectories. **Note:** For the MSSQL database, use version 4 of the JDBC drivers.
Java™ job type	JavaJobExecutor.properties	Use the `jarPath` keyword to specify the path to the directory where the jar files are stored.

Table 20. Configuration files for job types with advanced options (continued)

Job type	File name	Keyword
		This includes all jar files stored in the specified directory and all sub directories.
J2EE job type	J2EEJobExecutorConfig.properties	For more information about the J2EE job type, see the topic about configuring to schedule J2EE jobs in the *HCL Workload Automation: Administration Guide*.

Configuring security roles for users and groups

The dynamic workload broker provides two commands for managing resources and job definitions:

resource

to create, modify, associate, query, or set resources online or offline. For more information, see the section about the resource command in *User's Guide and Reference*.

jobstore

to manage job definitions. For more information, see the section about the jobstore command in *Scheduling Workload Dynamically*.

At master domain manager installation time, the `broker_role_mapping.xml` template is created to configure in WebSphere Application Server Liberty the users and groups authorized to use the dynamic workload broker commands. For the configuration procedure, see Mapping security roles to users and groups in WebSphere Application Server Liberty on page 121.

Mapping security roles to users and groups in WebSphere Application Server Liberty

About this task

When the dynamic workload broker instance is installed on your master domain manager, corresponding roles are set up in WebSphere Application Server Liberty. By default, these roles are not used. However, the authorization required to perform any tasks is always validated by WebSphere Application Server Liberty. Users are required to provide credentials for managing resources and job definitions using the resource and jobstore commands. These credentials correspond to existing users defined in the domain user registry or the LDAP server.

To allow users and groups to access the dynamic workload broker functions, they must be mapped to the security roles in WebSphere Application Server Liberty. This mapping allows those users and groups to access applications defined by the role. At installation time, the HCL Workload Automation administrative user (**wauser**) is assigned the **Administrator** role in WebSphere Application Server Liberty. The following roles are also created but they are not assigned to any users nor groups:

Operator

Monitors and controls the jobs submitted.

Administrator

Manages the scheduling infrastructure.

Submitter

Manages the submission of their own jobs and monitors and controls the job lifecycle. This is the typical role for an HCL Workload Automation user.

HCL Workload Automation acts as submitter of jobs to the HCL Workload Automation dynamic agent.

Configurator

Is the entity responsible for running the jobs on a local environment.

To map security roles to users and groups on the WebSphere Application Server Liberty, edit the `broker_role_mapping.xml` file located in `<`*`Liberty_installation_directory`*`>/usr/servers/engineServer/configDropins`.

You can edit the file to associate users and groups to the **Operator**, **Administrator**, **Developer**, or **Submitter** roles, as follows:

1. Copy the template file from the `templates` folder to a working folder.
2. Edit the template file in the working folder with the desired configuration.
3. Optionally, create a backup copy of the relevant configuration file present in the `overrides` directory in a different directory. Ensure you do not copy the backup file in the path where the template files are located.
4. Copy the updated template file to the `overrides` folder. Maintaining the original folder structure is not required.
5. Changes are effective immediately.

To enable all users to use the dynamic workload broker commands, remove the comment from the `special-subject` string, otherwise, specify the list of users or groups for each role.

Examples

In the following example, the **Operator** role is associated to user **user1**, the **Submitter** role is associated to all users belonging to **group1**, and the **Configurator** role is associated to all users authenticated by the server:

```
<server>
         <enterpriseApplication id="SchedulerEAR">
         <application-bnd>
         <security-role id="adminRole" name="Administrator">
         <user access-id="${user.twsuser.id}" name="${user.twsuser.id}" />
         <run-as userid="${user.twsuser.id}" password="${user.twsuser.password}"/>

        </security-role>
        <security-role id="operatorRole" name="Operator">
        <user name=?user1?/>

        </security-role>
        <security-role id="submitterRole" name="Submitter">
        <group name=?group1?/>

        </security-role>
        <security-role id="configuratorRole" name="Configurator">
        <special-subject type="ALL_AUTHENTICATED_USERS"/>
        </security-role>
        </application-bnd>
        </enterpriseApplication>
```

broker_role_mapping.xml file

Example

```
<server>
        <enterpriseApplication id="SchedulerEAR">
        <application-bnd>
        <security-role id="adminRole" name="Administrator">
        <user access-id="${user.twsuser.id}" name="${user.twsuser.id}" />
        <run-as userid="${user.twsuser.id}" password="${user.twsuser.password}"/>

        </security-role>
        <security-role id="operatorRole" name="Operator">

        </security-role>
        <security-role id="submitterRole" name="Submitter">

        </security-role>
        <security-role id="configuratorRole" name="Configurator">

        </security-role>
        </application-bnd>
        </enterpriseApplication>
```

Configuring command-line client access authentication

This section describes how to reconfigure the connection used by the command line client.

The command line client is installed automatically on the master domain manager and on fault-tolerant agents. On the master domain manager you use it to run all of the commands and utilities.

On any other workstation you use it to run one of the following commands:

- **composer**
- **conman**
- **evtdef**
- **logman**
- **optman**
- **planman**
- **sendevent**
- **wappman**

It is configured automatically by the installation process, but if you need to change the credentials that give access to the server on the master domain manager, or you want to use it to access a different master domain manager, modify the ***connection parameters*** as described in Connection parameters on page 124.

Note:

1. The ***connection parameters*** are not required to use the local **conman** program on a fault-tolerant agent.
2. The command-line client on the master domain manager uses exactly the same mechanism to communicate with the server as it does when it is installed remotely.

Connection parameters

About this task

The connection parameters can be provided in one of three ways:

Define them as local options in the `localopts` file

All fields except *username* and *password*, can be defined by editing the `TWA_home/TWS/localopts` properties file on the computer from which the access is required. See Setting local options on page 48 for a full description of the file and the properties.

In **localopts** there is a section for the general connection properties, which contains the following:

```
host = host_name
protocol = protocol
port = port number
proxy = proxy server
proxyport = proxy server port number
timeout = seconds
defaultws = master_workstation
useropts = useropts_file
```

In addition, there is a separate set of SSL parameters:

```
CLI SSL server auth = yes|no
CLI SSL cipher = cipher_class
CLI SSL server certificate =certificate_file_name
CLI SSL trusted dir =trusted_directory
```

Store some or all of them as user options in the `useropts` file

As a minimum, the **username** and **password** parameters can be defined in the `user_home/.TWS/useropts` file for the user who needs to make the connection. Also, if you need to personalize for a user any of the properties normally found in the `localopts` file, add the properties to the `useropts` file. The values in the `useropts` file always take precedence over those in the `localopts` file.

The minimum set of properties you would find in **useropts** is as follows:

```
username=user_ID
password=password
```

You can also add the JWT in the `useropts` file. See Setting user options on page 70 for a full description of the file and the properties.

Supply them when you use the command

When you use any of the commands you can add one or more of the connection parameters to the command string. These parameters take precedence over the parameters in **localopts** and **useropts**. This allows you, for example, to keep the parameters in the **localopts** file and just get users to supply the **username** and **password** parameters when they use one of the commands, avoiding the necessity to store this data in the **useropts** file for each user..

The parameters can either be supplied fully or partially in a file, to which you refer in the command string, or typed directly as part of the command string. The full syntax is as follows:

```
[-file <parameter_file>
|
[-host <host_name>]
[-port <port_number>]
[-protocol {http|https}]
[-proxy <proxy_name>]
[-proxyport <proxy_port_number>]
[-jwt JSON Web Token]
[-username <username>]
[-password <user_password>]
[-timeout <timeout>]
```

-file <*parameter_file*>

A file containing one or more of the connection parameters. Parameters in the file are superseded if the corresponding parameter is explicitly typed in the command.

-host <*host_name*>

The host name or IP address of the master domain manager to which you want to connect.

-port <*port_number*>

The listening port of the master domain manager to which you want to connect.

-protocol {http|https}

Enter either http or https, depending on whether you want to make a secure connection.

-proxy <*proxy_name*>

The host name or IP address of the proxy server involved in the connection (if any).

-proxyport <*proxy_port_number*>

The listening port of the proxy server involved in the connection (if any).

[-jwt *JSON Web Token*]

Specify the JWT to be used for authentication between the master domain manager and agents. You can retrieve the token from the Dynamic Workload Console. This parameter is mutually exclusive with the **username** and **password** parameters. The JWT authentication applies to the commands you launch in the shell where you specify the JWT.

You can use JWT when running the following commands:

- composer
- conman
- wappman

[-username *user_name*]

An HCL Workload Automation user with sufficient privileges to perform the operation. This parameter is mutually exclusive with the **jwt** parameter.

[-password *password*]

The password of the HCL Workload Automation user. This parameter is mutually exclusive with the **jwt** parameter.

-timeout <*timeout*>

The number of seconds the command line client is to wait to make the connection before giving a timeout error.

Note:

From the command line, neither the default workstation, nor the command line client SSL parameters can be supplied. These must always be supplied in either the `localopts` (see Setting local options on page 48) or the `useropts` file for the user (see Setting user options on page 70).

For monitoring commands, such as **conman showjobs**, **conman showresorces**, and so on, HCL Workload Automation uses the connection parameters of the user logged on to the computer

The command line client needs to assemble a full set of parameters, and it does so as follows:

1. First it looks for values supplied as parameters to the command
2. Then, for any parameters it still requires, it looks for parameters supplied in the file identified by the `-file` parameter
3. Then, for any parameters it still requires, it looks in the `useropts` file for the user
4. Finally, for any parameters it still requires, it looks in the `localopts` file

If a setting for a parameter is not specified in any of these places an error is displayed.

Entering passwords

About this task

Password security is handled as follows:

Password entered in `useropts` file

You type the connection password into the `useropts` file in unencrypted form. When you access the interface for the first time it is encrypted. This is the preferred method.

Password entered in the parameter file used by the command

You type the connection password into the parameter file in unencrypted form. It is not encrypted by using the command. Delete the file after use to ensure password security.

Password entered using the `-password` parameter in the command

You type the password in the command string in unencrypted form. It remains visible in the command window until you clear the command window contents.

Note: On Windows™ workstations, when you specify a password that contains double quotation marks (") or other special characters, make sure that the character is escaped. For example, if your password is `tws11"tws`, write it as `"tws11\"tws"` in **`useropts`**.

An active-active high availability scenario

Implement active-active high availability between the Dynamic Workload Console and the master domain manager so that a switch to a backup is transparent to Dynamic Workload Console users.

Use a load balancer between the Dynamic Workload Console servers and the master domain manager so that in the event the master needs to switch to a backup, the switch is transparent to console users.

Configure the master domain manager and backup master domain managers behind a second load balancer so that the workload is balanced across all backup master domain managers and the master domain manager. Load balancing distributes workload requests across all configured nodes to avoid any single node from being overloaded and avoids a single point of failure.

You might already have installed and configured a number of backup master domain managers, in addition to your master domain manager, that you use for dedicated operations and to alleviate the load on the master domain manager. For example, you might have one dedicated to monitoring activities, another for event management, and perhaps your scheduling activities are run on the master domain manager. Administrators must create engine connections for each of these and users have to switch between engines to run dedicated operations. Should one of them go down, users need to be notified about which engine to use as a replacement and switch to the replacement engine.

To simply this, configure a load balancer in front of the master domain manager and backup master domain managers so that users are unaware of when a switch occurs and administrators configure a single engine connection in single-sign on that points to the name or IP address and port number of the load balancer and not ever need to know the hostname of the current active master. The load balancer monitors the engine nodes and takes over the task of balancing the workload and the switch to a backup master domain manager is completely transparent to console instance users. Any backup master domain manager can satisfy HTTP requests, even those that can be satisfied only by the active master, such as requests on the plan, because the requests are proxied to and from the active master.

To complete the picture of a full high availability HCL Workload Automation environment, the RDBMS and the Dynamic Workload Console need to be configured in high availability. If your RDBMS includes a high availability disaster recovery (HADR) feature and it is enabled, you can configure the `datasource.xml` file on the WebSphere Application Server Liberty server of the master and backup components to add failover properties. The key-value pairs to set depend on your specific

RDBMS. As an example, Db2®'s datasource can be configured with the following set of properties in the XML element named **properties.db2.jcc**:

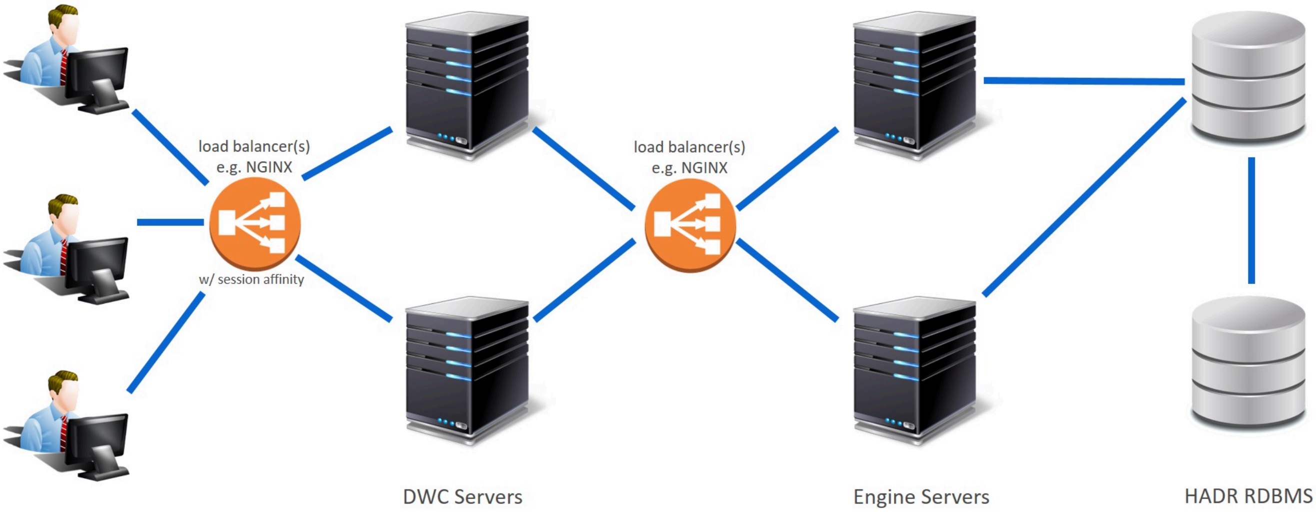
```
<properties.db2.jcc
            databaseName="TWS"
            user="…"
            password="…"
            serverName="MyMaster"
            portNumber="50000"
            clientRerouteAlternateServerName="MyBackup"
            clientRerouteAlternatePortNumber="50000"
     retryIntervalForClientReroute="3000"
     maxRetriesForClientReroute="100"
         />
```

For the Dynamic Workload Console, replicate it, and link the consoles to a load balancer that supports session affinity so that requests related to the same user session are dispatched to the same console instance.

Note: If your environment includes the HCL Workload Automation Agent for z/OS to schedule jobs (JCL) on the JES2 subsystem of z/OS, ensure that the value for the **host.bootstrap.port.sec** parameter specified in the `ports_variables.xml` file is the same on every workstation hosting the Dynamic Workload Console component in your environment. For more information about the location of the `ports_variables.xml` file, see Configuring HCL Workload Automation using templates on page 422.

The following is a sample of the high-level architecture of an active-active high availability system with two load balancers

Figure 1. And end-to-end environment configured for high availability



This environment configuration offers numerous benefits:

High availability

The load balancer monitors the nodes and takes care of balancing the workload across the nodes, eliminating the possibility of a node creating a single point of failure.

Scalability

As client requests increase, you can add additional backup master domain manager nodes to the configuration to support the increased workload.

User-friendly

Users are unaware of when a switch occurs to a different node. Administrators do not have to worry about creating new engine connections to run workloads on a different node.

Low overhead

This configuration does not require any manual intervention when nodes become unavailable. Additional flexibility is provided to console instance users who no longer have to run certain operations on dedicated nodes.

Optimization of hardware

Load balancing distributes session requests across multiple servers thereby utilizing hardware resources equally.

Note: When there are multiple Dynamic Workload Console servers connected to a load balancer and one of the servers becomes unavailable, load balancing takes place automatically, however, console users need to perform a page refresh and reopen any tabbed pages that were previously opened.

The following is intended to be a high-level view of the steps required to implement a scenario of this kind. It is an example and is not meant to be a verified procedure.

1. Install a load balancer on a workstation either within the HCL Workload Automation environment or external to the environment. Ports must be open between the load balancer and the Dynamic Workload Console nodes and the engine nodes.
2. Configure multiple Dynamic Workload Console instances in a cluster where the consoles share the same repository settings and a load balancer takes care of dispatching and redirecting connections among the nodes in the cluster. The load balancer must support **session affinity**.See the topic about configuring high availability across multiple Dynamic Workload Console nodes.
3. Exchange certificates between the load balancer and the Dynamic Workload Console and between the second load balancer and the master domain manager and backup master domain manager nodes.
4. Configure the load balancer configuration file with details about the Dynamic Workload Console, master domain manager, and backup master domain managers. The configuration file indicates which nodes (Dynamic Workload Console, master domain manager, and backup master domain managers) are available and the routes to be used to dispatch client calls to the Dynamic Workload Console server nodes.
5. Configure an engine connection that points to the name or IP address of the load balancer, and specify the incoming port number to the load balancer that corresponds to the outgoing port number to the master (default port number 31116). The load balancer must point to the HTTPS of the Dynamic Workload Console and the HTTPS of the master domain manager.
6. Configure an RDBMS in high availability and enable the HADR feature.

7. To configure the Dynamic Workload Console nodes in a cluster behind the load balancer, modify the `ports_config.xml` file as follows:

```
<httpEndpoint host="${httpEndpoint.host}" httpPort="${host.http.port}" httpsPort="${host.https.port}"
 id="defaultHttpEndpoint">
        <httpOptions removeServerHeader="true"/>
        <remoteIp useRemoteIpInAccessLog="true"/>
    </httpEndpoint>
```

This solution requires a load balancer that supports session affinity. Nginx is an example of a load balancer of this kind. The following is an abstract from a configuration file for an Nginx load balancer that demonstrates the configuration settings necessary to implement the high availability use case depicted in Figure 1: And end-to-end environment configured for high availability on page 128.

```
user  nginx;
worker_processes  10;  ## Default: 1
worker_rlimit_nofile 8192;

error_log  /var/log/nginx/error.log warn;
pid        /var/run/nginx.pid;


events {
  worker_connections  4096;  ## Default: 1024
}

http {
    include       /etc/nginx/mime.types;
    default_type  application/octet-stream;


    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';


    access_log  /var/log/nginx/access.log  main;


    sendfile        on;


    keepalive_timeout  65;


    upstream wa_console { ##DWC configuration
        ip_hash;
        server DWC1_HOSTNAME:DWC1_PORT max_fails=3 fail_timeout=300s;
        server DWC2_HOSTNAME:DWC2_PORT max_fails=3 fail_timeout=300s;
        keepalive 32;
    }

    upstream wa_server_backend_https {
       server MDM1_HOSTNAME:MDM1_PORT  weight=1;
       server MDM2_HOSTNAME:MDM2_PORT  weight=1;
    }


 server{
    listen          443 ssl;


    ssl_certificate /etc/nginx/certs/nginx.crt;
    ssl_certificate_key /etc/nginx/certs/nginxkey.key;
    ssl_trusted_certificate /etc/nginx/certs/ca-certs.crt;
    location /
    {
           proxy_pass https://wa_console;
        proxy_cache off;

        proxy_set_header Host $host;

        proxy_set_header Forwarded " $proxy_add_x_forwarded_for;proto=$scheme";
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header X-Forwarded-Host   $host;
        proxy_set_header X-Real-IP          $remote_addr;
        proxy_set_header X-Forwarded-Port  443;
    }
  }

 server{
    listen          9443 ssl;

    ssl_certificate /etc/nginx/certs/nginx.crt;
    ssl_certificate_key /etc/nginx/certs/nginxkey.key;
    ssl_trusted_certificate /etc/nginx/certs/ca-certs.crt;
    location /
    {
           proxy_pass https://wa_server_backend_https;
        proxy_cache off;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header X-Forwarded-Host   $host;
        proxy_set_header X-Real-IP          $remote_addr;

        proxy_set_header Connection "close";
    }
  }

}
```

where:

DWCx_HOSTNAME*:*DWCx_PORT

is the address of the Dynamic Workload Console.

MDMx_HOSTNAME*:*MDMx_PORT

is the address of the master domain manager.

HCL Workload Automation console messages and prompts

The HCL Workload Automation control processes (Netman, Mailman, Batchman, Jobman, and Writer) write their status messages (referred to as console messages) to standard list files. These messages include the prompts used as job and Job Scheduler dependencies. On UNIX® and Linux® operating systems, the messages can also be directed to the **syslog** daemon (**syslogd**) and to a terminal running the HCL Workload Automation console manager. These features are described in the following sections.

Setting sysloglocal on UNIX™

About this task

If you set **sysloglocal** in the local options file to a positive number, HCL Workload Automation's control processes send their console and prompt messages to the **syslog** daemon. Setting it to **-1** turns this feature off. If you set it to a positive number to enable system logging, you must also set the local option **stdlistwidth** to **0**, or a negative number.

HCL Workload Automation's console messages correspond to the following **syslog** levels:

LOG_ERR

Error messages such as control process abends and file system errors.

LOG_WARNING

Warning messages such as link errors and stuck job streams.

LOG_NOTICE

Special messages such as prompts and tellops.

LOG_INFO

Informative messages such as job launches and job and Job Scheduler state changes.

Setting **sysloglocal** to a positive number defines the syslog facility used by HCL Workload Automation. For example, specifying **4** tells HCL Workload Automationto use the local facility LOCAL4. After doing this, you must make the appropriate entries in the **/etc/syslog.conf** file, and reconfigure the syslog daemon. To use LOCAL4 and have the HCL Workload Automation messages sent to the system console, enter the following line in **/etc/syslog.conf**:

```
local4    /dev/console
```

To have the HCL Workload Automation error messages sent to the **maestro** and **root** users, enter the following command:

```
local4.err    maestro,root
```

The selector and action fields must be separated by at least one tab. After modifying **/etc/syslog.conf**, you can configure the **syslog** daemon by entering the following command:

```
kill -HUP `cat /etc/syslog.pid`
```

console command

About this task

You can use the conman **console** command to set the HCL Workload Automation message level and to direct the messages to your terminal. The message level setting affects only Batchman and Mailman messages, which are the most numerous. It also sets the level of messages written to the standard list file or files and the **syslog** daemon. The following command, for example, sets the level of Batchman and Mailman messages to **2** and sends the messages to your computer:

```
console sess;level=2
```

Messages are sent to your computer until you either run another **console** command, or exit conman. To stop sending messages to your terminal, enter the following conman command:

```
console sys
```

Automatic encryption for key product files

Key product files, such as the Symphony file, are automatically encrypted for all fresh installations using AES-256 or AES-128 cryptography starting from version 10.1.

Data breaches are becoming more and more common and pervasive in today's business world. Encryption is a key feature when it comes to protect sensitive data, such as the data at rest stored in your Symphony plan or message queues. For this reason, all fresh installations starting from this release automatically encrypt key product files using AES-256 or AES-128 cryptography.

Data at rest means data is not being accessed or used but instead stored on your computer, external hard drive, cloud storage, server, or database. Encryption at rest ensures that this data is protected and encrypted.

If you want HCL Workload Automation to encrypt files such as the Symphony file, messages queues, and the `useropts` file at runtime, you do not need to take any actions. By default, the product is automatically encrypted without your intervention. You can also define the folder containing the certificates and the certificates password using the **sslpassword** and **sslkeysfolder** parameters when installing the master domain manager and agents, both fault-tolerant agents and dynamic agents.

The following HCL Workload Automation elements are automatically encrypted:

- Symphony file
- messages queues
- `useropts` file
- `jmJobTableDir` directory on dynamic agents

You can also optionally encrypt the following HCL Workload Automation elements:

- SAP (r3batch) options file
- PeopleSoft options file
- secure command if you set the **useaeskeystore** argument. For more information, see the topic about the secure command in *HCL Workload Automation: Planning and Installation*.

Information about encryption keys is stored in the following `localopts` properties:

encrypt keystore file *file_name*

The path to the keystore PKCS12 file, containing the AES-256 or AES-128 key. The keystore is created automatically at installation time and the related path is inserted in this parameter. If you want to use a different keystore, you can create it and add the path in this option.

encrypt keystore pwd *password*

The path to the keystore stash file.

encrypt label

The label of the key in the keystore. When you modify a key label for key rotation, store the previous label in the **decrypt label list** property, so it can be retrieved if it is still used in the product. This property is case insensitive.

decrypt label list

The list of previously used aliases for key encryption. When you modify a key alias for key rotation, store the previous alias in this property. This storage method is useful if the obsolete key is still used in the product. Separate each value with a comma ",". Note that this property is commented. This property is case insensitive.

For more information about the `localopts` file, see Setting local options on page 48.

For more information about rotating the keys, see Encryption key rotation on page 134.

Encryption key rotation

Procedure to perform a key rotation.

About this task

You can optionally modify the existing encryption keys by performing a key rotation, for example if the existing keys expire or are no longer secure. Perform the following steps on the master domain manager and on each agent in the environment

1. Generate a new key by running the following keytool command:

   ```
   ./keytool -genseckey -alias new_alias_name -keyalg AES -keysize 256
   -storepass encrypt_keystore_pwd_in_clear -storetype PKCS12 -keystore encrypt_keystore_file
   ```

2. Change the `localopts` file as follows:
   a. Add the previous value of the **encrypt label** parameter to the **decrypt label list** parameter.
   b. Change the value of the **encrypt label** parameter to *new_alias_name*.

   For more information about the `localopts` file, see Setting local options on page 48.

   Result

   If the keystore does not exist, it is created. If it exists, the new key is added to the keystore.

Results

The current Symphony plan keeps using the previous key. To apply the new setting to the Symphony plan, run a JnextPlan command. The message boxes are encrypted immediately and the `useropts` file is encrypted as soon as you save the `localopts` file and launch a CLI command. Key product files are now encrypted with the new key.

Modifying jobmon service rights for Windows™

On Windows™ systems, the HCL Workload Automation jobmon service runs in the SYSTEM account with the right **Allow Service to Interact with Desktop** granted to it. You can remove this right for security reasons. However, if you do so, it prevents the service from launching interactive jobs that run in a window on the user's desktop. These jobs will be run, but are not accessible from the desktop or from HCL Workload Automation and do not have access to desktop resources. As a result, they may run forever or abend due to lack of resources.

Chapter 2. Configuring the Dynamic Workload Console

This chapter describes how to configure Dynamic Workload Console. It is divided into the following sections:

- Launching in context with the Dynamic Workload Console on page 136
- Configuring access to the Dynamic Workload Console on page 145
- Configuring the Dynamic Workload Console for Single Sign-On on page 149
- Configuring the Dynamic Workload Console to connect to an HCL Universal Orchestrator engine on page 154
- Configuring Dynamic Workload Console to connect to a remote SAP system on page 156
- Customizing your global settings on page 157
- Disable the What-if Analysis on page 176
- Configuring High Availability on page 177
- Configuring Dynamic Workload Console to view reports on page 177

Note: If, after installing, you have more than one instance of WebSphere Application Server Liberty managing any HCL Workload Automation products, ensure that they share the same LTPA token_keys.

Launching in context with the Dynamic Workload Console

Create a URL to launch the Dynamic Workload Console and have it directly open the results of a particular query.

By accessing the bookmark icon in the page of your interest, you can copy the URL of that page, and then include the copied URL in an external application, for example, to monitor jobs and job streams that are critical to your business, and to quickly and easily manage them.

Open your Dynamic Workload Console pages and access the information you need in just one click.

Launch in context: your environment at your fingertips.

Scenarios

The following main scenarios can be identified:

- Obtain the result of a monitor query on:
  - Jobs
  - Critical jobs
  - Job streams
  - Workstation
  - Workload
  - Existing tasks

For all the scenarios, you must create a basic URL.

Creating a basic URL

About this task

To create a basic URL, you need to define the URL to access the Dynamic Workload Console:

```
https://{WebUIHostName:adminSecurePort}
/console/?pageId=<pageID>
```

where:

WebUIHostname

It is the fully qualified hostname or the IP address of the computer where the Dynamic Workload Console is installed.

adminSecurePort

It is the number of the port on which the Dynamic Workload Console is listening.

pageId

It is the ID of the Dynamic Workload Console page that has to be launched.

Example

```
https://mypc:29443/console/?pageId=manage-roles
```

Advanced optional parameters

Depending on the query whose results you want to view, you can complete your URL with the following parameters:

Mandatory parameters

engineName

Specify the name of one or more engines to be used as filter.

objectType

Specify an object type as filter. The following are the supported object types in the Dynamic Workload Console:

- com.ibm.tws.objects.plan.JobInPlan
- com.ibm.tws.objects.plan.JobStreamInPlan
- com.ibm.tws.objects.plan.CriticalJobInPlan
- com.ibm.tws.objects.plan.WorkstattionInPlan
- com.ibm.tws.objects.plan.PromptInPlan
- com.ibm.tws.objects.plan.ResourceInPlan
- com.ibm.tws.objects.plan.DomainInPlan

plan

Specify a plan name as filter.

query

Specify a query to filter the results.

Note: Special characters must be replaced by encoded values. For example, # (number sign) must be replaced by %23, + (plus sign) must be replaced by %2B.

Example

A query like `@!@.@+state=#Waiting` must be written like this: `@!@.@%2Bstate=%23Waiting`

Optional parameters

columns

Specify the columns that you want to display in your result table. The following three options are available:

ALL

Display all columns.

DEFAULT

Diplay only the default columns.

Customized columns

Display the customized column to get a specific column result. For example, `“columns”: “Status,Internal Status”`.

If not specified, the default columns for this query are shown.

encrypt

Specify if you want to encrypt or not the engine name parameter. If true, the engine name has to be encrypted by using Base64 encode.

taskName

Specify the name of an existing task as filter.

Note: it is not recommended to specify both the taskName and query parameters in the URL. If both parameters are specified, the taskName parameter has the priority.

Monitor Jobs on distributed systems

The following is an example of URL to be launched to directly open the Monitor Workload page and obtain results about jobs on distributed systems.

To create a URL to monitor jobs on a distributed system, specify the following filters:

engineName

Specify the name of one or more engines to be used as filter.

objectType

Specify an object type as filter. The following are the supported object types in the Dynamic Workload Console:

- com.ibm.tws.objects.plan.JobInPlan
- com.ibm.tws.objects.plan.JobStreamInPlan
- com.ibm.tws.objects.plan.CriticalJobInPlan
- com.ibm.tws.objects.plan.WorkstattionInPlan
- com.ibm.tws.objects.plan.PromptInPlan
- com.ibm.tws.objects.plan.ResourceInPlan
- com.ibm.tws.objects.plan.DomainInPlan

plan

Specify a plan name as filter.

query

Specify a query to filter the results.

Note: Special characters must be replaced by encoded values. For example, # (number sign) must be replaced by %23, + (plus sign) must be replaced by %2B.

Example

A query like `@!@.@+state=#Waiting` must be written like this: `@!@.@%2Bstate=%23Waiting`

columns

Specify the columns that you want to display in your result table. The following three options are available:

ALL

Display all columns.

DEFAULT

Diplay only the default columns.

Customized columns

Display the customized column to get a specific column result. For example, `“columns”: “Status,Internal Status”`.

If not specified, the default columns for this query are shown.

encrypt

Specify if you want to encrypt or not the engine name parameter. If true, the engine name has to be encrypted by using Base64 encode.

Example:

```
https://mypc:9449/console?pageId=direct-query&properties={"query":"%2F%40%2F%40%23%2F%40%2F%40.%40","engineName
":"eJxzzUvPzEs1BAAKagKI","encrypt":true,"plan":"current-plan","objectType":"com.ibm.tws.objects.plan.JobInPlan"
,"columns":"Status,Internal Status,Folder (Job Stream),Job,Job Type,Workstation (Job),Job Stream,Workstation
 (Job Stream),Scheduled Time,Not Satisfied Dependencies,Priority,Job number,Earliest Start,Actual
 Start,Deadline"}
```

Monitor Jobs on z/OS® systems

The following is an example of URL to be launched to directly open the Monitor Workload page and obtain results about jobs on z/OS® systems.

To create a URL to monitor jobs on a z/OS® system, specify the following filters:

engineName

Specify the name of one or more engines to be used as filter.

objectType

Specify an object type as filter. The following are the supported object types in the Dynamic Workload Console:

- com.ibm.tws.objects.plan.JobInPlan
- com.ibm.tws.objects.plan.JobStreamInPlan
- com.ibm.tws.objects.plan.CriticalJobInPlan
- com.ibm.tws.objects.plan.WorkstattionInPlan
- com.ibm.tws.objects.plan.PromptInPlan
- com.ibm.tws.objects.plan.ResourceInPlan
- com.ibm.tws.objects.plan.DomainInPlan

plan

Specify a plan name as filter.

query

Specify a query to filter the results.

Note: Special characters must be replaced by encoded values. For example, # (number sign) must be replaced by %23, + (plus sign) must be replaced by %2B.

Example

A query like `@!@.@+state=#Waiting` must be written like this: `@!@.@%2Bstate=%23Waiting`

columns

Specify the columns that you want to display in your result table. The following three options are available:

ALL

Display all columns.

DEFAULT

Diplay only the default columns.

Customized columns

Display the customized column to get a specific column result. For example, `“columns”: “Status,Internal Status”`.

If not specified, the default columns for this query are shown.

encrypt

Specify if you want to encrypt or not the engine name parameter. If true, the engine name has to be encrypted by using Base64 encode.

Example:

```
https://
 mypc:9449/console?pageId=direct-query&properties={"query":"%40!%40","engineName":"eJyLyi92zUvPzEsFABGsA5M=","e
ncrypt":true,"plan":"current plan","objectType":"com.ibm.tws.objects.plan.JobInPlan","columns":"Status,Internal
 Status,Job Number,Job,Workstation,Job stream,Status Details,Scheduled Time,Job Identifier,Error Code,Time
 Dependent,Earliest Start,Planned Start,Actual Start,Deadline,Critical"}
```

Monitor Critical Jobs

The following is an example of URL to be launched to directly open the Monitor Workload page and obtain results about critical jobs.

To create a URL to monitor critical jobs, specify the following filters:

engineName

Specify the name of one or more engines to be used as filter.

objectType

Specify an object type as filter. The following are the supported object types in the Dynamic Workload Console:

- com.ibm.tws.objects.plan.JobInPlan
- com.ibm.tws.objects.plan.JobStreamInPlan
- com.ibm.tws.objects.plan.CriticalJobInPlan
- com.ibm.tws.objects.plan.WorkstattionInPlan
- com.ibm.tws.objects.plan.PromptInPlan
- com.ibm.tws.objects.plan.ResourceInPlan
- com.ibm.tws.objects.plan.DomainInPlan

plan

Specify a plan name as filter.

query

Specify a query to filter the results.

Note: Special characters must be replaced by encoded values. For example, # (number sign) must be replaced by %23, + (plus sign) must be replaced by %2B.

Example

A query like `@!@.@+state=#Waiting` must be written like this: `@!@.@%2Bstate=%23Waiting`

columns

Specify the columns that you want to display in your result table. The following three options are available:

ALL

Display all columns.

DEFAULT

Diplay only the default columns.

Customized columns

Display the customized column to get a specific column result. For example, `“columns”: “Status,Internal Status”`.

If not specified, the default columns for this query are shown.

encrypt

Specify if you want to encrypt or not the engine name parameter. If true, the engine name has to be encrypted by using Base64 encode.

Example:

```
https://mypc:9449/console?pageId=direct-query&properties={"query":"%2F%40%2F%40%23%2F%40%2F%40.%40","engineName
":"eJxzzUvPzEs1BAAKagKI","encrypt":true,"plan":"current-plan","objectType":"com.ibm.tws.objects.plan.CriticalJo
bInPlan","columns":"Risk level,Confidence Factor,Status,Internal Status,Folder (Job Stream),Job,Job
 Type,Workstation (Job),Job Stream,Workstation (Job Stream),Scheduled Time,Jobs Left on Critical Path,Critical
 Path Remaining Duration,Estimated Start,Estimated End,Earliest Start,Actual Start,Deadline,Critical Latest
 Start"}
```

Monitor Job Streams

The following is an example of URL to be launched to directly open the Monitor Workload page and obtain results about job streams.

To create a URL to monitor job streams, specify the following filters:

engineName

Specify the name of one or more engines to be used as filter.

objectType

Specify an object type as filter. The following are the supported object types in the Dynamic Workload Console:

- com.ibm.tws.objects.plan.JobInPlan
- com.ibm.tws.objects.plan.JobStreamInPlan
- com.ibm.tws.objects.plan.CriticalJobInPlan
- com.ibm.tws.objects.plan.WorkstattionInPlan
- com.ibm.tws.objects.plan.PromptInPlan
- com.ibm.tws.objects.plan.ResourceInPlan
- com.ibm.tws.objects.plan.DomainInPlan

plan

Specify a plan name as filter.

query

Specify a query to filter the results.

Note: Special characters must be replaced by encoded values. For example, # (number sign) must be replaced by %23, + (plus sign) must be replaced by %2B.

Example

A query like `@!@.@+state=#Waiting` must be written like this: `@!@.@%2Bstate=%23Waiting`

columns

Specify the columns that you want to display in your result table. The following three options are available:

ALL

Display all columns.

DEFAULT

Diplay only the default columns.

Customized columns

Display the customized column to get a specific column result. For example, `“columns”: “Status,Internal Status”`.

If not specified, the default columns for this query are shown.

encrypt

Specify if you want to encrypt or not the engine name parameter. If true, the engine name has to be encrypted by using Base64 encode.

Example:

```
https://mypc:9449/console?pageId=direct-query&properties={"query":"%2F%40%2F%40%23%2F%40%2F%40","engineName":"e
JxzzUvPzEs1BAAKagKI","encrypt":true,"plan":"current-plan","objectType":"com.ibm.tws.objects.plan.JobStreamInPla
n","columns":"Status,Internal Status,Folder,Job Stream,Workstation,Scheduled Time,Not Satisfied
 Dependencies,Total Jobs,Successful Jobs,Jobs Limit,Priority,Earliest Start,Actual Start,Deadline"}
```

Monitor Workstations

The following is an example of URL to be launched to directly open the Monitor Workload page and obtain results about workstations.

To create a URL to monitor workstations, specify the following filters:

engineName

Specify the name of one or more engines to be used as filter.

objectType

Specify an object type as filter. The following are the supported object types in the Dynamic Workload Console:

- com.ibm.tws.objects.plan.JobInPlan
- com.ibm.tws.objects.plan.JobStreamInPlan
- com.ibm.tws.objects.plan.CriticalJobInPlan
- com.ibm.tws.objects.plan.WorkstattionInPlan
- com.ibm.tws.objects.plan.PromptInPlan
- com.ibm.tws.objects.plan.ResourceInPlan
- com.ibm.tws.objects.plan.DomainInPlan

plan

Specify a plan name as filter.

query

Specify a query to filter the results.

Note: Special characters must be replaced by encoded values. For example, # (number sign) must be replaced by %23, + (plus sign) must be replaced by %2B.

Example

A query like `@!@.@+state=#Waiting` must be written like this: `@!@.@%2Bstate=%23Waiting`

columns

Specify the columns that you want to display in your result table. The following three options are available:

ALL

Display all columns.

DEFAULT

Diplay only the default columns.

Customized columns

Display the customized column to get a specific column result. For example, `“columns”: “Status,Internal Status”`.

If not specified, the default columns for this query are shown.

encrypt

Specify if you want to encrypt or not the engine name parameter. If true, the engine name has to be encrypted by using Base64 encode.

Example:

```
https://mypc:9449/console?pageId=direct-query&properties={"query":"%2F%40%2F%40","engineName":"eJxzzUvPzEs1BAAK
agKI","encrypt":true,"plan":"current-plan","objectType":"com.ibm.tws.objects.plan.WorkstationInPlan","columns":
"Link Status,Folder,Workstation,Agent Running,Writer Running,Start Time,Run Number,Limit,Domain,Type,Version"}
```

Existing task

The following is an example of URL to be launched to directly open the Monitor Workload page and obtain results about an existing task.

About this task

To create a URL to monitor an existing task, specify the following filters:

taskName

Specify the name of an existing task as filter.

Note: it is not recommended to specify both the taskName and query parameters in the URL. If both parameters are specified, the taskName parameter has the priority.

Example:

```
https://mypc:9449/console?pageId=direct-query&properties={"taskName":"MyTask"}
```

Configuring access to the Dynamic Workload Console

As soon as you finish installing the Dynamic Workload Console, you can launch it by logging in at the following link:

```
https://<your_ip_address>:9443/console/login.jsp
```

You can access the Dynamic Workload Console from any computer in your environment using a web browser through the secure HTTPS protocol and using the credentials specified at installation time.

By default, the Dynamic Workload Console is configured to use a local file-based user repository. Users defined in the user registry can log in to the Dynamic Workload Console and need to be associated to a role to be able access the Dynamic Workload Console features (see “Configuring roles to access the Dynamic Workload Console on page 146.)

If you use a central user registry that is based on the Lightweight Directory Access Protocol (LDAP) to manage users and groups and provide single sign-on, then you can set up an LDAP server and create an LDAP user registry to use with the Dynamic Workload Console. You can implement an LDAP user repository in place of the default file-based user registry by configuring the sample authentication templates provided in XML format. The following are the supported LDAP servers and the corresponding sample template that can be configured to replace the configuration file currently in use:

- File-based: `auth_basicRegistry_config.xml`
- IBM® Directory Server: `auth_IDS_config.xml`
- OpenLDAP: `auth_OpenLDAP_config.xml`
- Windows™ Server Active Directory: `auth_AD_config.xml`

In addition you can also add a line

See Configuring HCL Workload Automation using templates on page 422 for more information about the templates and the location.

Note: If two or more instances of Dynamic Workload Console share the same database repository for their settings, but they are not configured to be in a High Availability configuration, they all must be at the same fix pack level.

Configuring a user registry

To use LDAP user registry, users and groups must be created by the system administrator in the chosen LDAP server database.

Configuring user registries for the Dynamic Workload Console and all other HCL Workload Automation components is described in Configuring LDAP described in *Planning and Installation Guide*.

Configuring roles to access the Dynamic Workload Console

During the Dynamic Workload Console installation, new predefined roles are created. They determine which console panels are available to a user, and therefore which activities that user can perform from Dynamic Workload Console. More roles can be created and customized according to business needs. For more information about the creation of customized roles, see the section about customizing roles in *Dynamic Workload Console User's Guide*

Tip

It is not necessary to assign a role to every single user. If the user registry already contains groups of users that are properly defined for using the console, it is possible to assign roles to groups too. If groups are not available in the user registry, then the special role **all authenticated users** can be used to assign roles to *all* the users at once.

To assign roles to a default groups of users that are properly defined for using the console, add this property to the authentication file in use.

```
<jndiEntry jndiName="all.authenticated.users" value="my-group" />
```

.

Within the Dynamic Workload Console, you can create your own custom views to enable users to see all or a subset of HCL Workload Automation pages. To do it, you must have the **Administrator** role and perform the following steps:

1. Open the `authentication_config.xml` located in the following path:

   On UNIX operating systems

   DWC_DATA_dir`/usr/servers/dwcServer/configDropins/overrides`

   On Windows operating systems

   DWC_home`\usr\servers\dwcServer\configDropins\overrides`

2. Add the entity specifying username and password in users or groups.
3. Open **Dynamic Workload Console > Administration > Manage Roles**
4. Click `Entities` to associate the user or the group you have created to one of the roles from the list.
5. Add the entity and save.

The following lists the predefined roles created in WebSphere Application Server Liberty for accessing the HCL Workload Automation environments using Dynamic Workload Console:

API User

Users in this group can use only the Dynamic Workload Console APIs to perform the available actions. Logging to the Dynamic Workload Console through a Web browser would not give them access to any feature. For more details about the Dynamic Workload Console APIs, see `https://<DWC_hostname>:<port>/dwc/api`.

Administrator

Users with this role can see the entire portfolio and use all features of the Dynamic Workload Console.

Users with this group can also access and use all features of the Self-Service Catalog and theSelf-Service Dashboards mobile applications. From the Self-Service Catalog mobile application, these users can create and edit catalogs, create and edit services, add services to catalogs, submit services associated to job streams, and share catalogs and services with other users. From the Self-Service Dashboards mobile application, these users can create and edit dashboards to filter for jobs and workstations, display a dashboard of results, perform recovery actions on a single result.

From the Manage Roles panel, the administrator can add entities, manage pinned pages and shared boards.

Analyst

Users in this group can manage Dynamic Workload Console reports and user preferences.

Broker

Users in this group can define Broker settings, create and manage Broker jobs and resources, and monitor Broker computers and resources.

Developer

Users in this group can create, list, and edit workload definitions, workstations, and event rule definitions in the HCL Workload Automation database.

Mobile User

Users in this group can manage the Self-Service Catalog and theSelf-Service Dashboards mobile applications but the actions they can perform are limited to submitting service requests (job streams) from the Self-Service Catalog and , from the Self-Service Dashboards mobile application, displaying a dashboard of results and performing recovery actions on them.

Operator

Users in this group can see Dynamic Workload Console:

- All Monitor tasks.
- Jobs and job streams to be submitted on request
- Set User Preferences

The following table lists some entries of the navigation toolbar, and some activities that you can perform on the Dynamic Workload Console. Beside each item, the table shows the groups whose users are authorized to access them.

Table 21. Menu and Group Permissions

Menu Item	Groups with Permission
Quick Start	Administrator
All Configured Tasks	Administrator, Operator
Manage Workload Reports	Administrator, Analyst
Design -> Workload Definitions	Administrator Developer
Planning & Submission -> Workload Forecast	Administrator, Operator
Administration -> Workload Submission	Administrator Operator
Monitoring & Reporting	Administrator, Operator
Design -> Workload Definitions	Administrator
Reporting	Administrator, Analyst

Table 21. Menu and Group Permissions (continued)

Menu Item	Groups with Permission
Security ->Manage Engines	Administrator
Security -> Manage Settings	Administrator
Administration → Broker Settings; Design → Broker Design; Monitoring & Reporting → Broker Monitoring	Administrator, Broker

Configuring the Dynamic Workload Console for Single Sign-On

Single Sign-On (SSO) is a method of access control that allows a user to authenticate once and gain access to the resources of multiple applications sharing the same user registry.

This means that using SSO you can run queries on the plan or manage object definitions on the database accessing the engine without authenticating, automatically using the same credentials you used to log in to the Dynamic Workload Console.

The same is true when working with the Self-Service Catalog and Self-Service Dashboards apps from a mobile device. If the Dynamic Workload Console has been configured to use SSO, then these apps automatically use the same credentials used to log in to the Dynamic Workload Console.

After completing the installation or upgrade, you can set up Single Sign-On (SSO) for the Dynamic Workload Console and master domain manager using either an LTPA registry or a basic user registry. To achieve this, Dynamic Workload Console and master domain manager need to use the same user registry. Additionally, ensure that the contents of the `ltpa.keys` file are identical on both the Dynamic Workload Console and the master domain manager. The `ltpa.keys` file is located in the following path:

```
usr/servers/engineServers/resources/security
```

The Lightweight Directory Access Protocol (LDAP) is an application protocol for querying and modifying directory services running over TCP/IP - see the information about configuring a common LDAP for both the master and console in the post-installation section of the *Planning and Installation Guide* for more details.

If you configured Dynamic Workload Console to use Single Sign-On with an engine, then, the following behavior is applied:

If engine connection has the user credentials specified in its definitions

These credentials are used. This behavior regards also engine connections that are shared along with their user credentials.

If the user credentials are not specified in the engine connection

The credentials you specified when logging in to Dynamic Workload Console are used. This behavior regards also shared engine connections having unshared user credentials.

For detailed information about how to configure SSO using an LTPA token or an MP-JWT token, see How to configure the Dynamic Workload Console and the master domain manager for Single Sign-On on page 150.

How to configure the Dynamic Workload Console and the master domain manager for Single Sign-On

Configure the Dynamic Workload Console and the master domain manager for Single Sign-On.

About this task

You can configure Single Sign-On using a Lightweight Third-Party Authentication (LTPA) token or an MP-JWT token.

Note: When implementing a configuration in Single Sign-On, ensure you have not specified the engine credentials in the **Manage Engine** section.

Configuring the Dynamic Workload Console for Single-Sign-On with an LTPA token

About this task

To enable Single Sign-On between the Dynamic Workload Console and master domain manager, perform the following steps:

1. Configure an authentication provider for the Dynamic Workload Console as explained in the post-installation section of the *Planning and Installation Guide*.
2. Create the Access Control list for the authentication provider user or group. For example, to give full access on domain and folders to the LDAP group perform the following steps:
    a. From the Dynamic Workload Console open the **Manage Workload Security** panel and select **Give access to users and groups**.
    b. Select the LDAP group from the drop-down list and **FULLCONTROL** in the field **Role**.
    c. Select **Domain** and assign **ALLOBJECTS**.
    d. **Save and create new**
    e. Select the LDAP group from the drop-down list and **FULLCONTROL** in the field **Role**.
    f. Select **Folder** and assign the root by clicking `/`.
    g. **Save**
3. Ensure that the `ltpa.keys` file on both the Dynamic Workload Console and the master domain manager are identical, copying the file from one instance to the other. The file is located as follows:

    Dynamic Workload Console

    ```
    DWC_home/usr/servers/dwcServer/resources/security
    ```

    master domain manager

    ```
    TWA_home/usr/servers/engineServer/resources/security
    ```
4. Restart WebSphere Application Server Liberty on both the master domain manager and the Dynamic Workload Console by running stopAppServer and startAppServer.

Configuring the Dynamic Workload Console for Single Sign-On with an MP-JWT token

About this task

Perform the following steps:

1.
2. Create the Access Control list for the authentication provider user or group. For example, to give full access on folders to an LDAP group perform the following steps:
    a. From the Dynamic Workload Console open the **Manage Workload Security** panel and select **Give access to users and groups**.
    b. Select the LDAP group from the drop-down list and **FULL_CONTROL** in the field **Role**.
    c. Select **Folder** and assign the root by clicking `/`.
    d. **Save**

How to configure the Dynamic Workload Console 10.2.5 and a master domain manager 9.4.x for Single Sign-On

How to configure the Dynamic Workload Console 10.2.5 and a master domain manager 9.4.x for Single Sign-On.

Before you begin

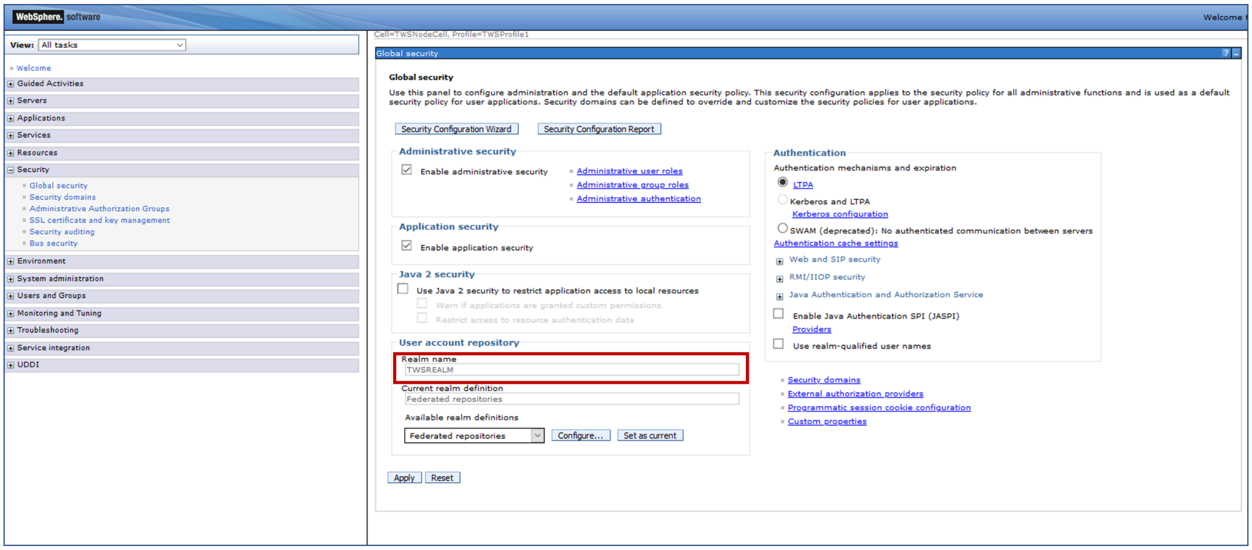
Ensure that the master domain manager V9.4.x is configured to use a Lightweight Directory Access Protocol (LDAP). The LDAP should be the same one already configured and used by the Dynamic Workload Console 10.2.5. For further information about how to configure an LDAP, see the section about configuring a user registry in *HCL Workload Automation: Planning and Installation*.

About this task

To configure the Dynamic Workload Console 10.2.5 and the master domain manager V9.4.x for Single Sign-On, perform the following steps:

1. Access the **WebSphere administrative console** of the master domain manager V9.4.x and go to **Global security** in the **Security** section.
2. In the Global security panel, take note of the value for the **Realm name** in the *User account repository section*. The realm name is required later in this section.

Figure 2. Realm name in the WebSphere administrative console



Cell=TWSNodeCell, Profile=TWSProfile1

Global security

Global security

Use this panel to configure administration and the default application security policy. This security configuration applies to the security policy for all administrative functions and is used as a default security policy for user applications. Security domains can be defined to override and customize the security policies for user applications.

Administrative security

☑ Enable administrative security

Application security

☑ Enable application security

Java 2 security

□ Use Java 2 security to restrict application access to local resources

□ Warn if applications are granted custom permissions

□ Restrict access to resource authentication data

User account repository

Realm name: TWSREALM

Current realm definition: Federated repositories

Available realm definitions: Federated repositories

Authentication

Authentication mechanisms and expiration

● LTPA

○ Kerberos and LTPA

○ SWAM (deprecated): No authenticated communication between servers

⊞ Web and SIP security

⊞ RMI/IIOP security

⊞ Java Authentication and Authorization Service

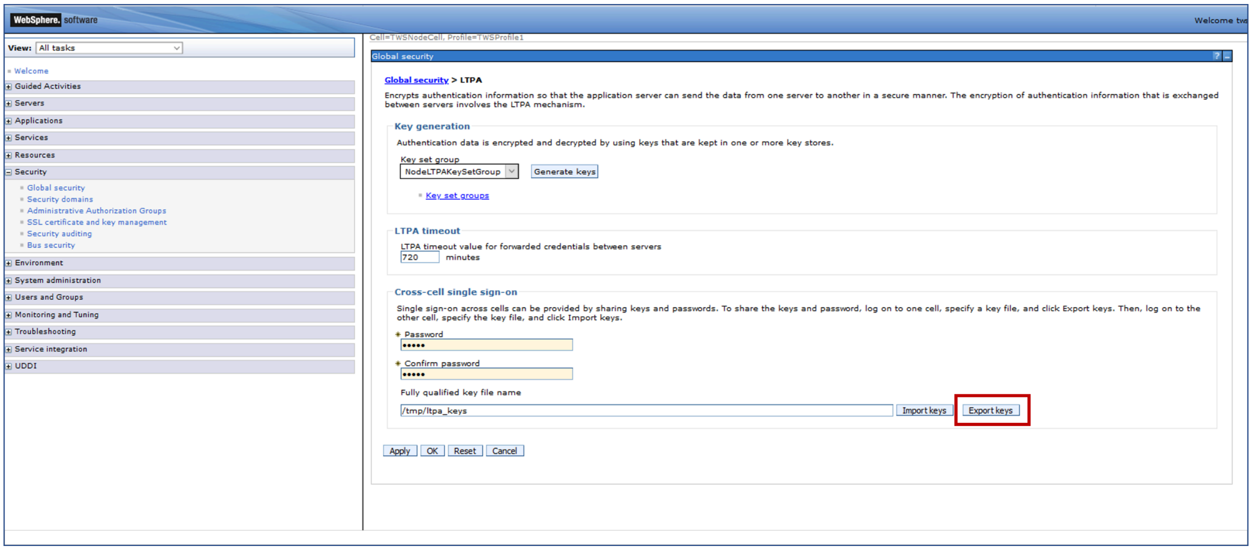
□ Enable Java Authentication SPI (JASPI)

□ Use realm-qualified user names

3. In *Authentication*, select **LTPA** as the authentication mechanism, and enter a password to export the ltpa keys.

   Note: Take note of the password. The password you enter is required later during the import.

Figure 3. Export of the ltpa keys file



Cell=TWSNodeCell, Profile=TWSProfile1

Global security

Global security > LTPA

Encrypts authentication information so that the application server can send the data from one server to another in a secure manner. The encryption of authentication information that is exchanged between servers involves the LTPA mechanism.

Key generation

Authentication data is encrypted and decrypted by using keys that are kept in one or more key stores.

Key set group: NodeLTPAKeySetGroup

LTPA timeout

LTPA timeout value for forwarded credentials between servers: 720 minutes

Cross-cell single sign-on

Single sign-on across cells can be provided by sharing keys and passwords. To share the keys and password, log on to one cell, specify a key file, and click Export keys. Then, log on to the other cell, specify the key file, and click Import keys.

* Password: •••••

* Confirm password: •••••

Fully qualified key file name: /tmp/ltpa_keys

4. Before replacing the existing ltpa on the Dynamic Workload Console 10.2.5, create a backup copy in a different directory. The existing ltpa keys file can be found in the following path:

```
DWC_DATA_dir/usr/servers/dwcServer/resources/security/
```

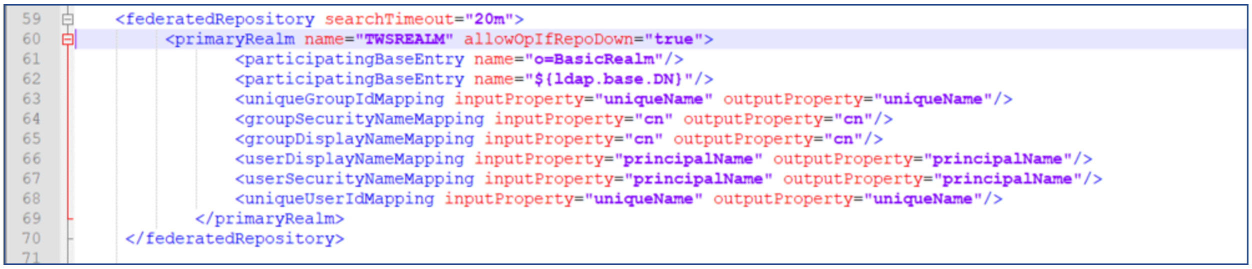
```
DWC_home\usr\servers\dwcServer\resources\security\
```

5. Rename the exported ltpa keys file to **ltpa.keys** and copy it to the same path as the existing file on the Dynamic Workload Console 10.2.5.
6. Open the authentication configuration file previously customized to enable the LDAP for the Dynamic Workload Console 10.2.5, and ensure that the realm name is the same as the one specified for the master domain manager V9.4.x (see Step 2 on page 151). The authentication configuration file is located in the following path:

   ```
   DWC_DATA_dir/usr/servers/dwcServer/configDropins/overrides/
   ```

   ```
   DWC_home\usr\servers\dwcServer\configDropins\overrides\
   ```

   Figure 4. Realm name in the authentication template

   ```
   <federatedRepository searchTimeout="20m">
       <primaryRealm name="TWSREALM" allowOpIfRepoDown="true">
           <participatingBaseEntry name="o=BasicRealm"/>
           <participatingBaseEntry name="${ldap.base.DN}"/>
           <uniqueGroupIdMapping inputProperty="uniqueName" outputProperty="uniqueName"/>
           <groupSecurityNameMapping inputProperty="cn" outputProperty="cn"/>
           <groupDisplayNameMapping inputProperty="cn" outputProperty="cn"/>
           <userDisplayNameMapping inputProperty="principalName" outputProperty="principalName"/>
           <userSecurityNameMapping inputProperty="principalName" outputProperty="principalName"/>
           <uniqueUserIdMapping inputProperty="uniqueName" outputProperty="uniqueName"/>
       </primaryRealm>
   </federatedRepository>
   ```

   Where *TWSREALM* is the default realm name.
7. Add the password in XOR format in the **ssl_config.xml** as follows:
   a. Copy the **ssl_config.xml** file from the following path:

      ```
      DWC_home/usr/servers/dwcServer/configDropins/defaults/
      ```

      ```
      DWC_home\usr\servers\dwcServer\configDropins\defaults\
      ```

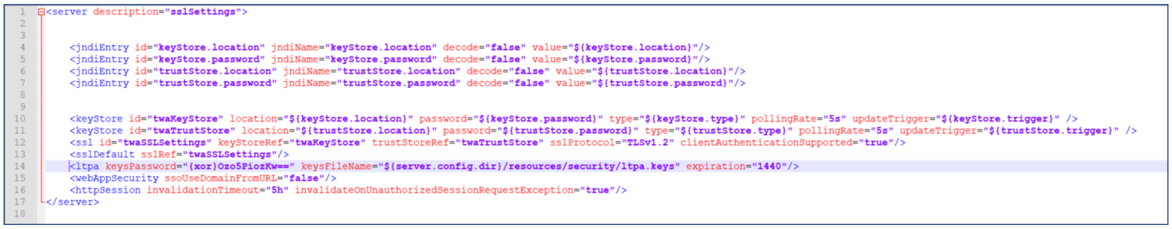
   b. Paste the **ssl_config.xml** file in the following path:

      ```
      DWC_DATA_dir/usr/servers/dwcServer/configDropins/overrides/
      ```

      ```
      DWC_home\usr\servers\dwcServer\configDropins\overrides\
      ```

c. Open the **ssl_config.xml** file and enter the password in XOR format. The password is the one you specified for the master domain manager V9.4.x during the export (see Step 3 on page 152).

Figure 5. Password in XOR format



```
<server description="sslSettings">


    <jndiEntry id="keyStore.location" jndiName="keyStore.location" decode="false" value="${keyStore.location}"/>
    <jndiEntry id="keyStore.password" jndiName="keyStore.password" decode="false" value="${keyStore.password}"/>
    <jndiEntry id="trustStore.location" jndiName="trustStore.location" decode="false" value="${trustStore.location}"/>
    <jndiEntry id="trustStore.password" jndiName="trustStore.password" decode="false" value="${trustStore.password}"/>


    <keyStore id="twaKeyStore" location="${keyStore.location}" password="${keyStore.password}" type="${keyStore.type}" pollingRate="5s" updateTrigger="${keyStore.trigger}" />
    <keyStore id="twaTrustStore" location="${trustStore.location}" password="${trustStore.password}" type="${trustStore.type}" pollingRate="5s" updateTrigger="${trustStore.trigger}" />
    <ssl id="twaSSLSettings" keyStoreRef="twaKeyStore" trustStoreRef="twaTrustStore" sslProtocol="TLSv1.2" clientAuthenticationSupported="true"/>
    <sslDefault sslRef="twaSSLSettings"/>
    <ltpa keysPassword="{xor}Ozo5PiozKw==" keysFileName="${server.config.dir}/resources/security/ltpa.keys" expiration="1440"/>
    <webAppSecurity ssoUseDomainFromURL="false"/>
    <httpSession invalidationTimeout="5h" invalidateOnUnauthorizedSessionRequestException="true"/>
</server>

```

8. Restart the Dynamic Workload Console 10.2.5.

Results

You successfully configured the Dynamic Workload Console 10.2.5 and the master domain manager V9.4.x for Single Sign-On.

Configuring the Dynamic Workload Console to connect to an HCL Universal Orchestrator engine

You can connect an on-premises Dynamic Workload Console V10.2.3 to an HCL Universal Orchestrator engine by performing the steps described in this topic. There are three different procedures:

- Connecting an already installed Dynamic Workload Console to an HCL Universal Orchestrator engine on page 154
- Connecting to an HCL Universal Orchestrator engine during the installation of the Dynamic Workload Console on page 155
- Connecting a Dynamic Workload Console deployed on Kubernetes to an HCL Universal Orchestrator engine on page 156

Connecting an already installed Dynamic Workload Console to an HCL Universal Orchestrator engine

1. From the directory in which the Dynamic Workload Console is installed, browse to the `configDropins/templates` folder, copy the `jwtsso.xml` file and paste it into the `configDropins/overrides` directory without changing any parameter.
2. Create a file named `unoca.crt` in a directory of the virtual machine that hosts the Dynamic Workload Console.
3. From the secret that has been deployed by HCL Universal Orchestrator, extract the `ca.crt` file and paste it into the `unoca.crt` file.
4. Import the certificate by running the following command:

```
keytool -importcert -file unoca.crt -alias uno
 -keystore /<dwc_data>/usr/servers/dwcServer/resources/security/TWSServerTrustFile.p12
```

5. Export the server certificate from the Dynamic Workload Console by running the following command:

```
keytool -export
 -keystore /<dwc_data>/usr/servers/dwcServer/resources/security/TWSServerKeyFile.p12 -alias
 server -file tls_dwc.pem -rfc
```

Note: Check the extracted certificate and verify that no `^Ms` is present in the file. If any `^Ms` is present, remove it by running the following command:

```
sed ‘s/\r//’ tls_dwc.pem > tws_dwc_clear.crt
```

6. Copy the extracted certificate and create a secret in the namespace that hosts the HCL Universal Orchestrator deployment.
7. In the `values.yaml` file of HCL Universal Orchestrator, add the name of the secret that you created in the previous step within the following parameter:

   ```
   global.dwcconsole.certSecretName
   ```

8. Define an access control list for every user that must be authorized to connect to an HCL Universal Orchestrator engine.
9. Run the `helm upgrade` command with the same parameters used during the installation of HCL Universal Orchestrator:

   ```
     helm upgrade <uno_release_name> <repo_name>/hcl-uno-chart -f values.yaml -n
    <uno_namespace>
   ```

Connecting to an HCL Universal Orchestrator engine during the installation of the Dynamic Workload Console

1. Create a file named `unoca.crt` in a directory of the virtual machine that is designated to host the Dynamic Workload Console.
2. Copy the `unoca.crt` file into the `additionalCAs` folder. For more information about the `additionalCAs` folder, see SSL configuration options.
3. Run the `dwcinst` command.
4. From the directory in which the Dynamic Workload Console is installed, browse to the `configDropins/templates` folder, copy the `jwtsso.xml` file and paste it into the `configDropins/overrides` directory without changing any parameter.
5. From the secret that has been deployed by HCL Universal Orchestrator, extract the `ca.crt` file and paste it into the `unoca.crt` file.
6. Import the certificate by running the following command:

   ```
   keytool -importcert -file unoca.crt -alias uno
    -keystore /<dwc_data>/usr/servers/dwcServer/resources/security/TWSServerTrustFile.p12
   ```

7. Export the server certificate from the Dynamic Workload Console by running the following command:

   ```
   keytool -export
    -keystore /<dwc_data>/usr/servers/dwcServer/resources/security/TWSServerKeyFile.p12 -alias
    server -file tls_dwc.pem -rfc
   ```

   Note: Check the extracted certificate and verify that no `^Ms` is present in the file. If any `^Ms` is present, remove it by running the following command:

```
sed ‘s/\r//’ tls_dwc.pem > tws_dwc_clear.crt
```

8. Copy the extracted certificate and create a secret in the namespace that hosts the HCL Universal Orchestrator deployment.
9. In the `values.yaml` file of HCL Universal Orchestrator, add the name of the secret that you created in the previous step within the following parameter:

   ```
   global.dwcconsole.certSecretName
   ```

10. Define an access control list for every user that must be authorized to connect to an HCL Universal Orchestrator engine.
11. Run the `helm upgrade` command with the same parameters used during the installation of HCL Universal Orchestrator:

    ```
     helm upgrade <uno_release_name> <repo_name>/hcl-uno-chart -f values.yaml -n
    <uno_namespace>
    ```

Connecting a Dynamic Workload Console deployed on Kubernetes to an HCL Universal Orchestrator engine

1. From the secret that has been deployed by HCL Universal Orchestrator, copy the `ca.crt` file.
2. Load the third-party certificate by following the procedure described here.
3. Upgrade the Dynamic Workload Console by running the `helm upgrade` command.
4. From the directory in which the Dynamic Workload Console is installed, browse to the `configDropins/templates` folder, copy the `jwtsso.xml` file and paste it into the `configDropins/overrides` directory without changing any parameter.
5. Copy the `ca.crt` file generated during the deployment of the Dynamic Workload Console, and use the file to create a new secret into the HCL Universal Orchestrator namespace.
6. Copy the name of the secret created in the previous step into the following section of the `values.yaml` file of HCL Universal Orchestrator:

   ```
   global.dwcconsole.certSecretName
   ```

7. Define an access control list for every user that must be authorized to connect to an HCL Universal Orchestrator engine.
8. Run the `helm upgrade` command with the same parameters used during the installation of HCL Universal Orchestrator:

   ```
    helm upgrade <uno_release_name> <repo_name>/hcl-uno-chart -f values.yaml -n
   <uno_namespace>
   ```

Configuring Dynamic Workload Console to connect to a remote SAP system

You can create a new SAP connection.

About this task

The purpose of the following scenario is to show how to create a new SAP connection in the context of an example.

The scope of the scenario is to create a new SAP connection named “S4HANA” with the workstation named “DYN_AGENT”, using an options file named “FILE_AGENT1_r3batch.opts”.

1. Create a new SAP connection as follows:
    a. From the **Administration** menu, click **Manage SAP Connections** page, and then select the engine.
    b. Click **New Connection**.
    c. In **SAP Connection Name**, type `S4HANA`.
    d. In **Workstation**, click the **lookup** icon, search the workstation named "DYN_AGENT", and select it.
    e. In **Option file**, click the **lookup** icon, search the option file named "FILE_AGENT1_r3batch.opts", and select it.
    f. Test the connection.
2. Click **Ok** to save it.

Results

You successfully created a SAP connection named “S4HANA”, and now you can start creating, scheduling, and control your SAP jobs directly from the HCL Workload Automation.

Customizing your global settings

How to customize global settings.

About this task

To customize the behavior of the Dynamic Workload Console, you can optionally configure some advanced settings. These settings are specified in a customizable file named `TdwcGlobalSettings.xml.template`.

By default, the customizable file is copied into the following path after you install the Dynamic Workload Console:

On Windows operating systems:

DWC_home`\usr\servers\dwcServer\registry\TdwcGlobalSettings.xml.template`

On UNIX and Linux operating systems:

DWC_home`/usr/servers/dwcServer/registry/TdwcGlobalSettings.xml.template`

If you have Administrator privileges, you can modify the file to replace default values with customized ones and enable commented sections. To enable commented sections, remove the tags that enclose the section. You then save the file locally with the name `TdwcGlobalSettings.xml`.

You can add and modify some customizable information, such as:

- The URLs that link to videos in the Dynamic Workload Console. For example, you can link to a company intranet server to view help videos rather than to a public video site.
- The maximum number of objects to be shown in the graphical views.
- The setting to display the plan view in a new window.
- The auto refresh interval for the **Show Plan View** graphical view.
- The creation of predefined tasks.

- The URLs where you can store customized documentation about your jobs or job streams to associate customized documentation to them.
- The current user registry in use.
- The timeout to read and write information on a HCL Workload Automation for Z engine.
- The maximum number of objects to be retrieved with a query, the maximum number of rows to display in a table, and the maximum number of direct queries to maintain in history.
- Allowing or preventing users from sharing tasks and engine connections.
- The display of all dependencies, both satisfied and unsatisfied.
- The use of audit files to track activities in the Self-Service Catalog and Self-Service Dashboards mobile applications.
- Displaying or hiding all predecessors from the What-if Analysis Gantt view.

This file is accessed at each login, and all configurations specified in the file are immediately applied, except for the **precannedTaskCreation** property. This property is read only when a user logs in for the first time and is then used whenever this user logs in again.

You can use any text or XML editor to edit this file, but ensure that you save it is as a valid XML file.

The file is organized into sections that group similar properties. An explanation of each section is available in the file. For more information, see TdwcGlobalSettings.xml sample on page 171.

Sections can also be repeated multiple times in the same file and applied differently to different user roles. To apply a section only to the users belonging to a role, the section must be included within the tags `<settings role="user_role">` and `</settings>`, where:

<user_role>

The user for which the enclosed configuration must be applied. The default value is all users, unless otherwise specified.

Only one **settings** section can be specified for each role. If a user has more than one role, the settings associated to the higher role are used.

To edit the file, proceed as follows:

1. Stop WebSphere Application Server Liberty Base using the following command:

   UNIX™

   Stop the application server

   ```
   ./stopAppServer.sh  [-direct]
   ```

   Windows™

   Stop the application server

   ```
   stopAppServer.bat [-direct
               [-wlpHome <installation_directory>]
               [-options <parameters>]]
   ```

as described in the section about starting and stopping WebSphere Application Server Liberty Base in *Administration Guide*.

2. Log is as root or Administrator to the Dynamic Workload Console.
3. Browse to

   On Windows operating systems:

   DWC_home`\usr\servers\dwcServer\registry\TdwcGlobalSettings.xml.template`

   On UNIX and Linux operating systems:

   DWC_home`/usr/servers/dwcServer/configDropins/templates/TdwcGlobalSettings.xml.template`

4. Edit the file as necessary, rename it to `TdwcGlobalSettings.xml` and save it.
5. Start WebSphere Application Server Liberty Base using the following command:

   UNIX™

   Start the application server

   ```
   ./startAppServer.sh  [-direct]
   ```

   Windows™

   Start the application server

   ```
   startAppServer.bat [-direct]
   ```

   as described in the section about starting and stopping WebSphere Application Server Liberty Base in *Administration Guide*.

Example:

```
<?xml version"1.0"?>
<tdwc>
.
.
<settings>
<graphViews>
<property name="planViewNewWindow" value="true"/>
</graphViews>
</settings>

<settings  role="TWSWEBUIOperator">
<graphViews>
<property name="planViewNewWindow" value="false"/>
</graphViews>
</settings>
.
.
</tdwc>
```

To view the complete syntax for the file, see TdwcGlobalSettings.xml sample on page 171.

Customize video URLs

This section shows how you should customize your URLs that link video content in the Dynamic Workload Console so that you can link to a company intranet server to view help videos rather than a public video site.

The `_baseURL` prefix will be added to all your video URLs . If you do not specify a link for your video the default setting will automatically be used.

```
<?xml version"1.0"?>
<tdwc>
.
.
<settings>
-<videoGallery>
<property name="_baseURL" value=""></property>
<property name="depLoop" value=""></property>
<property name="highlightRelDep" value=""></property>
<property name="viewDepPrompt" value=""></property>
<property name="usingImpactView" value=""></property>
<property name="createUseTasks" value=""></property>
<property name="weAddRemoveFile" value=""></property>
<property name="weCreateDeps" value=""></property>
<property name="weAddJob" value=""></property>
<property name="weHighlightDeps" value=""></property>
<property name="weCreateJCL" value=""></property>
</videoGallery>
```

Override graphical view limits

This section contains the configuration parameters that apply to the graphical views in the plan, such as the maximum number of objects shown in each view.

planViewMaxJobstreams

The maximum number of job streams displayed in the Plan View. Default value is **1000**. Values greater than **1000** are not supported.

preProdPlanViewMaxJobstreams

The maximum number of job streams displayed in the preproduction plan view. Default value is **1000**. Values greater than **1000** are not supported.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
<graphViews>
<property name="planViewMaxJobstreams" value="1000"></property>
<property name="preProdPlanViewMaxJobstreams" value="1000"></property>
</graphViews>
 </settings>
.
```

```
.
</tdwc>
```

See TdwcGlobalSettings.xml sample on page 171 to view the complete syntax for the file.

For more information about how to customize global settings, see Customizing your global settings on page 157.

Plan View in new window

This section is used to prevent Internet Explorer 7 from freezing while using the Plan View. To solve the problem, set value to **true**.

planViewNewWindow

Set it to **true** if you want the plan view to be displayed in a new window each time it is launched. Default value is **false**.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
<graphViews>
<property name="planViewNewWindow" value="true"/>
</graphViews>
.
.
 </settings>
</tdwc>
```

See TdwcGlobalSettings.xml sample on page 171 to view the complete syntax for the file.

For more information about how to customize global settings, see Customizing your global settings on page 157.

Plan View auto refresh interval

Use this section to change the default setting of the auto refresh interval for the Show Plan View graphical view for all users. By default, the auto refresh interval is `300 seconds` (five minutes).

PlanViewAutorefresh

The graphical representation of the Plan View is automatically refresh every 300 seconds by default. To change this setting, edit the value assigned to the **DefaultTime** property. The minimum value you can set is 30 seconds. Any value specified below this value is reset to 30 seconds. You must restart the Dynamic Workload Console application server after modifying this value.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
<PlanViewAutorefresh>
<property name="DefaultTime" value="300"/>
</PlanViewAutorefresh>
.
.
```

```
 </settings>
</tdwc>
```

See TdwcGlobalSettings.xml sample on page 171 to view the complete syntax for the file.

For more information about how to customize global settings, see Customizing your global settings on page 157.

Disable and customize NewsFeed function

This section contains the configuration details to be constantly up-to-date with product information.

FeedURL

Contains the URL from which you receive news and updates. Default value is:https://community.ibm.com/community/user/legacy

FeedType

A string that identifies the format of update information. Default value is **JSONP**.

PollInterval

The interval in seconds between two checks for updates. Default value is **600**.

PollInitialDelay

An initial delay in seconds before the first attempt to read the news feeds. After the initial load, the poll interval is used. Default value is **120**.

NewsFeed

Property used to add further customized news feeds. Specify the format and address of the file that contains the customized communication. Supported formats are RSS 2.0 and ATOM 1.0. You must write the communication in ATOM 1.0 or RSS 2.0 format and store this file in the an HTTP server complying with the *same origin policy*. For browser security reasons, this policy permits to access information only on server using the same protocol, hostname and port number as the one to which you are connected. Optionally, if you want to store your customized feed on an external server, you must configure an HTTP reverse proxy server mapping the external server address.

```
<property name="NewsFeed" type="RSS"
value="http://DWC_hostname:portnumber.com/news.rss" />
```

Note: To specify multiple feeds, you must specify multiple **NewsFeed** properties.

NewsFeedCategory

The name of the customized information. It can be used to identify informational, warning or alert messages, for example. The path to an image can also be added to better identify the information with an icon.

To add more category images, specify a list of properties named **NewsFeedCategory**, for example:

```
<property name="NewsFeedCategory" value="my company info"
icon="http://www.my.company.com/info.png" />
<property name="NewsFeedCategory" value="my company alert"
icon="http://www.my.company.com/alert.png" />
```

If no customized feed is specified, the default feed is used, which retrieves the latest product information from official support sites. To disable any notification, comment the entire section. To disable only external notifications about product information updates, assign an empty string as value to the `FeedURL` property of `JSONP` feed like:

```
 <property name="FeedURL" type="JSONP" value="" />
```

Example:

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
<NewsFeed>
<property name="NewsFeed" type="RSS"
value="http://www.DWC_hostname:portnumber.com/my_rss.xml" />
<property name="NewsFeed" type="ATOM"
value="http://www.DWC_hostname:portnumber.com/my_atom.xml" />

<property name="PollInterval" value="600" />
<property name="PollInitialDelay" value="1" />

<property name="FeedURL" type="JSONP" value="" />

<property name="NewsFeedCategory"
value="my company info" icon="http://www.DWC_hostname:portnumber.com
/info.png" />
<property name="NewsFeedCategory"
value="my company alert" icon="http://www.DWC_hostname:portnumber.com
/alert.png" />

</NewsFeed>
 </settings>
.
.
</tdwc>
```

See TdwcGlobalSettings.xml sample on page 171 to view the complete syntax for the file.

For more information about how to customize global settings, see Customizing your global settings on page 157.

Disable and customize the creation of predefined tasks

This section defines the environment for which predefined tasks are created.

precannedTaskCreation

Some predefined tasks are created by default and are available when you log in to the console. There is a predefined Monitor task for every object, for both z/OS® and distributed engines. Default value is **all**. To change this setting, use one of the following values:

all

All predefined tasks are created. This is the default.

distributed

Only predefined tasks for distributed engines are created

zos

Only predefined tasks for z/OS engines are created

none

No predefined task is created.

```
<?xml version"1.0"?>
<tdwc>
.
.
   <settings>
     <application>
      <property name="precannedTaskCreation" value="all"/>
     </application>
   </settings>
.
.
</tdwc>
```

See TdwcGlobalSettings.xml sample on page 171 to view the complete syntax for the file.

For more information about how to customize global settings, see Customizing your global settings on page 157.

Add customized URL to job and job streams

This section contains URLs where you can store customized documentation about your jobs or job streams. By default, this setting is not specified. If you want to associate customized documentation to a job or job stream, use this setting to specify the external address where this information is located.

If you want to specify a URL where customized documentation for a job and job stream is stored, uncomment the section lines, specify the required URL, and optionally assign a name to the UI label by specifying a value for the customActionLabel property. By default this name is **Open Documentation**. This label is then displayed in the **More Actions** menus in Monitor Jobs and Monitor Job Streams tasks, as well as in the graphical views of the plan (in the object's tooltips, context menus and properties). In this example, selecting **Open Documentation** accesses the relevant documentation making it possible to open the documentation while monitoring your job or job stream in the plan.

To implement this setting, assign values to the following keywords:

customActionLabel

The name of the action displayed in menus, object properties, and tooltips to access customized documentation about your jobs or job streams. By default this name is "Open Documentation" unless you customize the name with this keyword.

jobUrlTemplate

The address of your job documentation. No default value available.

jobstreamUrlTemplate

The address of your job stream documentation. No default value available.

Consider the following example:

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
  <twsObjectDoc>
    <property name="jobstreamUrlTemplate"
      value="http://www.yourhost.com/tws/docs/${js_encoded_folder_path}${js_name_w}"/>
    <property name="jobUrlTemplate"
      value="http://www.yourhost.com/docs/jobs/${job_name_w}"/>
    <property name="customActionLabel" value="Your Custom Label Name"/>
  </twsObjectDoc>
 </settings>
.
.
</tdwc>
```

See TdwcGlobalSettings.xml sample on page 171 to view the complete syntax for the file.

These properties must be valid URLs, containing one or more of the variables listed in the table below.

If you use any of the following special characters in the URL, you must write them as follows:

Table 22. Syntax for special characters

Special characters	Write them as...
quote (")	\"
apostrophe (')	&apos;
ampersand (&)	&amp;
less than (<)	&lt
greater than (>)	&gt
backslash (\)	\\

Multiple variables can be included in a URL and must be specified using the following syntax: `${variable}`:

Table 23. Variables used in the URL definition

Name	Object	Description
job_number_w	Job z/OS®	The number of the job
job_wkst_w	Job	The name of the workstation on which the job runs and the folder where it is stored, if any.
job_jsname_w	Job	The name of the job stream that contains the job and the folder where it is stored, if any.

Table 23. Variables used in the URL definition (continued)

Name	Object	Description
job_jswkst_w	Job	The name of the job stream that contains the job and the folder where it is stored, if any.
job_actualarrival_w	Job z/OS®	The actual start time of the job (date format: YYYY-MM-DDThh:mm:ss)
job_actualend_w	Job z/OS®	When the job actually completed (date format: YYYY-MM-DDThh:mm:ss)
job_starttime_w	Job	The start time of the job (date format: YYYY-MM-DDThh:mm:ss)
job_id_w	Job	The ID of the job
job_returncode_w	Job	The return code of the job
js_name_w	Job stream	The name of the job stream that contains the job
js_wkst_w	Job stream	The name of the job stream that contains the job and the folder where it is stored, if any.
js_id_w	Job stream	The job stream ID
js_latest_start_w	Job stream	The latest time at which a job stream can start (date format: YYYY-MM-DDThh:mm:ss)
engine_name_w	Engine	The name of the engine connection
engine_host_w	Engine	The hostname of the engine connection
engine_port_w	Engine	The port number of the engine connection
engine_plan_w	Engine	The ID of selected plan
engine_serv_w	Engine	The remote server name of the engine connection

User registry

Use this section to configure some properties related to the User Registry in use.

groupIdMap

The property groupIdMap is related to the groups of User Registry, and can be modified to map and display the specified value of each group. By default the common name of the group is displayed.

importSettingsMaxFileSize

The property importSettingsMaxFileSize is related to the "Manage settings" > "Import Settings" functionality and defines the max file size of the uploaded TDWCSettings.xml. KB is the unit of measure, and by default, it is set to 102400 KB (100 MB). If you need to upload a property file bigger than 100MB, you can increase this value, but for security purposes, it is strongly suggested to revert the file size back to the default value once the import has been performed.

Examples:

```
<?xml version"1.0"?>
<tdwc>
.
.
<settings>
<security>
<property name="groupIdMap" value="cn"></property>
<property name="importSettingsMaxFileSize" value="102400"></property>
</security>
</settings>
.
.
</tdwc>
```

Therefore, if you need to change the default value `"cn"` to "`racfid`", you can define this property as follows:

```
<property name="groupIdMap" value="racfid"></property>
```

See the section about the TdwcGlobalSettings.xml sample in *Administration Guide* to view the complete syntax for the file.

or see the section about user settings in *Dynamic Workload Console User's Guide* to manage Dynamic Workload Console settings.

For more information about how to customize global settings, see Customizing your global settings on page 157.

z/OS http connections

Use this section to configure the timeout to read and write information on IBM® Z® Workload Scheduler engine. When you connect to the IBM® Z® Workload Scheduler engine to retrieve a list of defined objects, you receive an error message if the list is not returned within the timeout period. The value is expressed in milliseconds.

Example:

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
<http>
<property name="zosHttpTimeout" value="90000" />
</http>
.
.
 </settings>
</tdwc>
```

See TdwcGlobalSettings.xml sample on page 171 to view the complete syntax for the file.

For more information about how to customize global settings, see Customizing your global settings on page 157.

Limit the number of objects retrieved by queries

Use this section to configure: the number of results displayed for Monitor tasks, the maximum number of rows to display on each page, and the number of direct queries to maintain in history.

If you want to limit the number of results produced by your queries, you can specify the maximum number of items that must be retrieved using the monitorMaxObjectsPM property.

Note: monitorMaxObjectsPM property only limits the number of results for archived plans queries. The property does not affect current plan queries.

For Multiple engine tasks, this limit is applied to each engine included in the query. Therefore, if you specify a limit of 500 results and, for example, you run a Monitor jobs on multiple engine task on three engines, the results produced by your query will be no more than 500 *for each engine*, for a maximum of 1500 rows.

Note: This setting does not apply to Monitor critical jobs tasks.

To set the maximum number of rows to display in a table view, configure the maxRowsToDisplay property.

To set the maximum number of direct queries to maintain in history, configure the maxHistoryCount property. These queries are available from the pull-down for the Query field on the Monitor Workload page.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
  <monitor>
    <property name="monitorMaxObjectsPM" value="2000"></property>
  </monitor>

  <ph rev="v92"><monitor>
    <property name="maxRowsToDisplay" value="25"></property>
  </monitor>

  <monitor>
    <property name="maxHistoryCount" value="100"></property>
  </monitor>
</ph>
 </settings>

<settings>
        <search>
                  <property name="search_max_limit" value="1500"></property>
        </search>
    </settings>
.
.
</tdwc>
```

See TdwcGlobalSettings.xml sample on page 171 to view the complete syntax for the file.

For more information about how to customize global settings, see Customizing your global settings on page 157.

Limit task and engine sharing

Use this section to prevent users from sharing tasks and engines.

By default there is no limit to task and engine sharing and all users are authorized to share their tasks and engine connections. If you want to change this behavior, preventing users from sharing tasks and engines, set this property to **true**.

The property default value is **false**, set it to **true** to enable the limit:

limitShareTask

Set to true to prevent users from sharing tasks.

limitShareEngine

Set to true to prevent users from sharing engine connections.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
  <security>
   <property name="limitShareTask"     value="false" />
   <property name="limitShareEngine"   value="false" />
  </security>
 </settings>
.
.
</tdwc>
```

See TdwcGlobalSettings.xml sample on page 171 to view the complete syntax for the file.

For more information about how to customize global settings, see Customizing your global settings on page 157.

Show all dependencies

This section defines whether to show all dependencies displayed, regardless of their being satisfied or not.

ShowDependencies

When you open the dependencies panel from Monitor jobs and Monitor job streams task results, by default only **Not Satisfied** dependencies are shown. Uncomment this section and leave the value set to "**true**" to have all dependencies displayed, regardless of their being satisfied or not. Possible values are:

true

All dependencies displayed, regardless of their being satisfied or not.

false

Only not satisfied dependencies are displayed.

```
<?xml version"1.0"?>
<tdwc>
```

```
.
.
 <settings>
  <ShowDependencies>
   <property name = "AlwaysShowAllDependencies"
             value="true"></property>
  </ShowDependencies>
 </settings>
.
.
 </tdwc>
```

See TdwcGlobalSettings.xml sample on page 171 to view the complete syntax for the file.

For more information about how to customize global settings, see Customizing your global settings on page 157.

Auditing mobile app activity

This section defines whether to track activities performed in the Self-Service Dashboards application in an auditing log file.

For information about the name and location of the log file, see the logs and traces section in the *Troubleshooting Guide*.

SSAuditing

This value is set to "**true**" by default so that operations performed in the Self-Service Dashboards application are written to a log file. The log file contains information such as creation, modification and deletion dates, the operations performed in the mobile apps, and the user performing the operations. Possible values are:

true

Operations performed in the Self-Service Dashboards application are tracked in an auditing log file.

false

Operations performed in the Self-Service Dashboards application are not tracked in an auditing log file.

SSAuditingLogSize

The maximum size of a log file in KB. When a log file reaches the maximum size, the system rolls that log file over and creates a new file. By default, the maximum size of a log file is 100 KB.

SSAuditingLogFiles

The default number of log files to create. When this number is met and the latest log file reaches its maximum size, the system deletes the oldest log file and rolls the latest file over and creates a new file.

```
<?xml version"1.0"?>
<tdwc>
.
.
 <settings>
 <SSCMAuditing>
            <property name = "SSAuditing"         value="true"></property>
            <property name = "SSAuditingLogSize"  value="100"></property>
            <property name = "SSAuditingLogFiles" value="2"></property>
 </settings>
 .
 .
</tdwc>
```

See TdwcGlobalSettings.xml sample on page 171 to view the complete syntax for the file.

For more information about how to customize global settings, see Customizing your global settings on page 157.

Modifying the number of archived plans displayed in the Dynamic Workload Console

You can modify the number of archived plans displayed in the Monitor Workload view of the Dynamic Workload Console. The default number is 30 plans.

To modify the default number, configure the following property in the **TdwcGlobalSettings.xml** file:

```
<monitor>
    <property name="maxArchivedPlan"value="30"></property>
  </monitor>
```

See TdwcGlobalSettings.xml sample on page 171 to view the complete syntax for the file.

For more information about how to customize global settings, see Customizing your global settings on page 157.

Show or hide predecessors from What-if Analysis Gantt view

When you have hundreds of predecessors, you can optimize performance by excluding them from the What-if Analysis Gantt view. By default, all predecessors are loaded into the What-if Analysis Gantt view. To exclude them, uncomment this section and leave the default setting of the property **whatIfAutoLoadPreds** to "`false`" . To revert back to the default behavior either set the property to "`true`" or comment the section again in the **TdwcGlobalSettings.xml** file.

To modify the default setting, configure the following property in the **TdwcGlobalSettings.xml** file:

```
      <WhatifAnalysis>
            <property name = "whatIfAutoLoadPreds" value="false"></property>
      </WhatifAnalysis>
```

See TdwcGlobalSettings.xml sample on page 171 to view the complete syntax for the file.

For more information about how to customize global settings, see Customizing your global settings on page 157.

TdwcGlobalSettings.xml sample

The following example is a sample of the file:

```
 <?xml version="1.0"?>
<tdwc>

   <!--
   #######################################################################################
   ##################         SETTINGS FOR ALL USERS               #######################
   #######################################################################################
   -->
   <settings>

   <!--
   #######################################################################################
   ##################       CUSTOMIZE LINKS TO VIDEOS             #######################
   #######################################################################################
   -->
   <!--
   This section shows how you should customize your URLs that link video content in
   the Dynamic Workload Console so that you can link to a company intranet server
   to view help videos rather than a public video site.

   ### This prefix "_baseURL" will be added to all video URLs ###

   ### Links to videos, missing entries or empty (blank) values are not considered ###

   #### Graphical view: detect loop                        ####
   #### Graphical view: highlight and release dependencies ####
   #### Graphical view: reply to prompt dependency         ####
   #### Graphical view: using the impact view              ####
   #### Table: creating and using tasks                    ####
   #### Workload editor: add and remove a file dependency  ####
   #### Workload editor: create a dependency               ####
   #### Workload editor: add a job                         ####
   #### Workload editor: highlight dependencies            ####
   #### Workload editor: creating a z/OS job               ####
   -->

   <!--
       <videoGallery>
           <property name="_baseURL" value=""></property>
           <property name="depLoop" value=""></property>
           <property name="highlightRelDep" value=""></property>
           <property name="viewDepPrompt" value=""></property>
           <property name="usingImpactView" value=""></property>
           <property name="createUseTasks" value=""></property>
           <property name="weAddRemoveFile" value=""></property>
           <property name="weCreateDeps" value=""></property>
           <property name="weAddJob" value=""></property>
           <property name="weHighlightDeps" value=""></property>
           <property name="weCreateJCL" value=""></property>
       </videoGallery>
   -->
   <!--
   #######################################################################################
   ##################           GRAPHICAL VIEW SETTINGS            #######################
   #######################################################################################
   -->
   <!--
   This section specifies the maximum number of objects shown in each graphical view.
   Default value is 1000 for all properties.
   -->
   <!--
       <graphViews>
           <property name="planViewMaxJobstreams" value="1000"></property>
           <property name="preProdPlanViewMaxJobstreams" value="1000"></property>
       </graphViews>
   -->
   <!--
###################################  AutoLayout configuration  ########################

 nodeSep: a number of pixels representing the separation between adjacent nodes in the same rank. Default is 30
 edgeSep: a number of pixels representing the separation between adjacent edges in the same rank. Default is 0
 rankSep: a number of pixels representing the separation between ranks. Default is 0
 rankDir: direction of the layout (Possible values are: "TB" (top-to-bottom),"BT" (bottom-to-top),"LR" (left-to-right),"RL" (right-to-left)). Default is TB
 marginX: a number of pixels representing the separation between adjacent nodes in the same rank. Default is 100
 marginY: number of pixels to use as a margin around the top and bottom of the graph. Default is 100
 setLinkVertices: If set to true the layout will adjust the links by setting their vertices. It defaults to false.
      If the option is defined as a function it will be used to set the vertices of links at the end of the layout.
   -->
<!--
  <AutoLayout>
        <property name="nodeSep" value="30"/>
  <property name="edgeSep" value="0"/>
  <property name="rankSep" value="0"/>
  <property name="rankDir" value="TB"/>
  <property name="marginX" value="100"/>
  <property name="marginY" value="1500"/>
  <property name="marginY" value="1500"/>
  <property name="setLinkVertices" value="false"/>
    </AutoLayout>
   -->
   <!--
   #######################################################################################
   #####################           PLAN VIEW IN NEW WINDOW          ######################
   #######################################################################################
   -->
   <!--
   This section is used to prevent Internet Explorer 7 from freezing while using the Plan View. To solve the problem, set value         to true.
   Default value is false
```

```
    -->
    <!--
        <graphViews>
            <property name="planViewNewWindow" value="true"/>
        </graphViews>
    -->
    <!--
    #######################################################################################
    #############       DISABLE /CUSTOMIZE CREATION OF PREDEFINED TASKS         ###########
    #######################################################################################
    -->
    <!--
    To avoid or customize the creation of predefined tasks at first logon.
    Possible values are:
        all             both distributed and z/OS tasks are created. This is the default value
        none            no task is created
        distributed     only distributed tasks are created
        zos             only z/OS tasks are created
    -->
    <!--
        <application>
            <property name="precannedTaskCreation" value="all"/>
            <property name="updateWorkstationMaxNumber" value="20"/>
        </application>
    -->

    <!--
        <PositionSorting>
            <property name="enabled" value="true"></property>
        </PositionSorting>
    -->

    <!--
    #######################################################################################
    #############       ADD A CUSTOM DOCUMENTATION URL TO JOB/JOBSTREAM         ###########
    #######################################################################################
    -->
    <!--
    This section contains URLs where you can store customized documentation about your jobs or job streams.
    By default this setting is not specified. If you want to associate customized documentation to a job or
    job stream, use this setting to specify the external address where this information is located.
    If you want to specify a URL to be opened as related documentation for jobs and job streams,
    uncomment the section lines so that a new action, Open Documentation, is inserted in the More Actions
    menu for Monitor Jobs and Monitor Job Streams tasks. The new action links to the specified URL

   You can customize the URL template by using variables. The variables have the following syntax
                    ${<variable_name>}

    For the complete list of variables, please refer to the documentation.

    -->
    <!--
        <twsObjectDoc>
            <property name="jobstreamUrlTemplate" value="http://www.yourhost.com/tws/docs/jobstream/${js_name_w}" />
            <property name="jobUrlTemplate"       value="http://www.yourhost.com/docs/jobs/${job_name_w}" />
            <property name="customActionLabel"    value="Custom Action" />
        </twsObjectDoc>
    -->

<!--
    #######################################################################################
    #############                        USER REGISTRY                          ###########
    #######################################################################################
       In this section you can configure properties related to the User Registry in use.

       The property groupIdMap is related to the groups of User Registry, and can be modified
       to map and display the specified value of each group. By default the common name
       of the group is displayed.

       The property importSettingsMaxFileSize is related to the "Manage settings" > "Import Settings"
       functionality and defines the max file size of the uploaded TDWCSettings.xml.
       KB is the unit of measure, and by default, it is set to 102400 KB (100 MB).
       If you need to upload a property file bigger than 100MB, you can increase this value, but
       for security purposes, it is strongly suggested to revert the file size back to the default
       value once the import has been performed.
-->

    <!--
        <security>
            <property name="groupIdMap" value="cn"></property>
            <property name="importSettingsMaxFileSize" value="102400"></property>
        </security>
    -->

    <!--
        #######################################################################################
        ##################                Z/OS HTTP CONNECTIONS                     ###########
        #######################################################################################

    Use this section to increase or decrease timeout for http connection in Z/OS
    environment. Change this setting if you receive a connection timeout
    using plugin actions/picklists.

    The setting is in milliseconds.
    -->
        <!--
            <http>
                <property name="zosHttpTimeout" value="90000" />
            </http>
        -->
    <!--
    #######################################################################################
    #################    LIMIT THE NUMBER OF OBJECTS RETURNED BY THE QUERIES   ###########
    #######################################################################################

        Use this section to configure: the number of results displayed for Monitor tasks, the maximum number of rows
        to display on each page, and the number of direct queries to maintain in history.
        This setting applies to all tasks except for Monitor critical jobs and Monitor jobs on multiple engines.
        If you want to limit the number of results produced by your queries, you can specify the maximum number of items that must be retrieved.
        To set the maximum number of rows to display in a table view, configure the maxRowsToDisplay property.
        To set the maximum number of direct queries to maintain in history, configure the maxHistoryCount property.
        These queries are available from the pull-down for the Query field on the Direct Query page.
        The property maxRowInfoNumber indicates the maximum number of objects to which actions can be performed in the plan.

        -->



        <!--
        <monitor>
          <property name="monitorMaxObjectsPM" value="2000"></property>
    </monitor>
    <monitor>
        <property name="maxRowsToDisplay" value="25"></property>
    </monitor>
    -->
    <!--
     You modify the number of archived plans displayed in the Monitor Workload view of the Dynamic
     Workload Console. The default number is 30 plans.
    -->
    <!--
    <monitor>
        <property name="maxArchivedPlan"value="30"></property>
    </monitor>
    -->
    <!--
    <monitor>
        <property name="maxHistoryCount" value="100"></property>
    </monitor>
    -->
 <!--
  Custom SQL report HTML format maximum limit. The default limit is 10000.
 -->
 <!--
  <monitor>
        <property name="SQL_REPORT_HTML_FORMAT_RESULT_MAX_NUMBER" value="10000"></property>
    </monitor>
    -->
     <!--
  <monitor>
        <property name="PROMPT_FILTER" value="true"></property>
    </monitor>
 -->
    <!--
    <settings>
        <search>
            <property name="search_max_limit" value="500"></property>
        </search>
    </settings>
    -->
    <!--
    <monitor>
        <property name="maxRowInfoNumber" value="100"></property>
    </monitor>
    -->
    <!--
    Increase the share maximum limit to share engines, boards, queries, and data sources.
    -->
    <!--
    <ShareLimit>
    <property name = "MaxShareCount" value="1000"></property>
    </ShareLimit>
    -->
    <!--
    ##################################################################################
```

```
    ##################          LIMIT TASK AND ENGINE SHARING              ###########
    ##################################################################################

    Use this section to prevent users from sharing tasks and engines.
    By default there is no limit to task and engine sharing and all users are authorized to share
    their tasks and engine connections. If you want to change this behavior, preventing users from
    sharing tasks and engines, set this property to true. The property default value is false,
    set it to true to enable the limit:

    -->
    <!--
        <security>
            <property name="limitShareTask"     value="false" />
            <property name="limitShareEngine"   value="false" />
        </security>
    -->

<!--
    #######################################################################################
    ##################      CHANGE DEFAULT BEHAVIOR FOR DEPENDENCIES PANEL      ###########
    #######################################################################################

    Use this section to change the default behavior of the UI when displaying
    dependencies in the dependencies panel. By setting this value to true, by default,
    all dependencies are displayed, and not just the unsatisfied ones.
-->
<!--
        <ShowDependencies>
            <property name = "AlwaysShowAllDependencies" value="true"></property>
        </ShowDependencies>
-->

<!--
    ######################################################################################
    ##################    CHANGE DEFAULT BEHAVIOR FOR SSC AND SSD AUDITING     ###########
    ######################################################################################
    Use this section to change the default behavior of the auditing of activities performed
    using the Self-Service Catalog and the Self-Service Dashboards applications. By default,
    auditing is enabled. You can also set the maximum size of the log file before it rolls
    over to a new log file, and the maximum number of log files maintained.
    Note: This section is valid only for the Self-Service Catalog V9.x, not for the latest one.
-->
 <!--
        <SSCMAuditing>
            <property name = "SSAuditing"         value="true"></property>
            <property name = "SSAuditingLogSize"  value="100"></property>
            <property name = "SSAuditingLogFiles" value="2"></property>
        </SSCMAuditing>
 -->

    <!--
    ######################################################################################
    ##################             URL FOR AGENT LICENSE                       ###########
    ######################################################################################
    Use this section to change the default Agent License URL.
    -->
    <!--
        <AgentLicense>
            <property name = "URL" value="Workoad Automation SaaS agent license document"></property>
        </AgentLicense>
    -->

    </settings>

    <!--
    #######################################################################################
    ##################       SETTINGS FOR ALL Administrators users    #####################
    #######################################################################################
    -->
    <settings role="Administrator">
    <!-- Put here setting to be applied only to users with Administrator role  -->
    </settings>
    <!--
    #######################################################################################
    ##################        SETTINGS FOR ALL Operators users      #######################
    #######################################################################################
    -->
    <settings role="Operator">
    </settings>
    <!--
    #######################################################################################
    ##################     SETTINGS FOR ALL Configurator users      #######################
    #######################################################################################
    -->
    <settings role="Configurator">
    </settings>
    <!--
    #######################################################################################
    ##################      SETTINGS FOR ALL Developer users        #######################
    #######################################################################################
    -->
    <settings role="Developer">
    </settings>
    <!--
    #######################################################################################
    ##################        SETTINGS FOR ALL Analyst users        #######################
    #######################################################################################
    -->
    <settings role="Analyst">
    </settings>

</tdwc>
```

For more information about how to customize global settings, see Customizing your global settings on page 157.

Disable the What-if Analysis

You can disable the What-if Analysis in your environment by setting the **optman** `enWhatIf | wi` global option to *no* (default value is *yes*).

The `enWhatIf | wi` global option interacts with the `enWorkloadServiceAssurance | wa` global option, which enables or disables privileged processing of mission-critical jobs and their predecessors. For details about this interaction, see the following table.

Table 24. Interaction between `enWorkloadServiceAssurance` and `enWhatIf` global options

Options	Interaction
**`enWorkloadServiceAssurance	wa`** is set to *yes* **`enWhatIf	wi`** is set to *yes*	Both the Workload service assurance and the What-if Analysis features are fully enabled in your environment.
**`enWorkloadServiceAssurance	wa`** is set to *yes* **`enWhatIf	wi`** is set to *no*	The Workload service assurance is enabled. The What-if Analysis feature is disabled and an exception is issued if you try to use it.
**`enWorkloadServiceAssurance	wa`** is set to *no* **`enWhatIf	wi`** is set to *yes*	The Workload service assurance is partially enabled, just to allow the What-if Analysis feature to work properly. This means that: • The Workload service assurance is disabled and an exception is issued if you try to use it. • No critical job is added to the plan.
**`enWorkloadServiceAssurance	wa`** is set to *no* **`enWhatIf	wi`** is set to *no*	Both the Workload service assurance and the What-if Analysis features are disabled in your environment.

Configuring High Availability

How to configure, change, and share your settings repository.

Performance can be highly improved by configuring multiple Dynamic Workload Console instances in a High Availability configuration, so as to have multiple console instances working at the same time and with the same repository settings.

The Dynamic Workload Console is set to be always in High Availability and a front-end Network Dispatcher must be set up to handle and distribute all incoming session requests.

If you use a Dynamic Workload Console in High Availability configuration, when you connect to a Dynamic Workload Console you are not actually connecting to a specific console but to a load balancer that dispatches and redirects the connections among the nodes in the configuration. Therefore, for example, if a node fails, new user sessions are directed to other active nodes in the configuration.

To implement this kind of configuration, the Administrator must ensure that the `datasource.xml`, located in the following path `opt/wa/DWC/DWC_DATA/usr/servers/dwcServer/configDropins/overrides`, has the same configuration on every Dynamic Workload Console in the cluster.

Configuring Dynamic Workload Console to view reports

This topic describes the configuration steps that you perform to be able to see the reports from the Dynamic Workload Console, if are using an Oracle database.

To access the databases where reports are stored, you must have the following prerequisites:

- A user ID and password to access the database
- A working connection between the Dynamic Workload Console and the database

Perform the following step on the system where the HCL Workload Automation engine is running:

- Configuring for an Oracle database on page 177
- Configuring for a Db2 database in HADR on page 179

Configuring for an Oracle database

About this task

Actions taken on HCL Workload Automation engine:

For Oracle, the IT administrator, or the HCL Workload Automation IT administrator, or both working together, do the following:

1. Use the **TWS Oracle user** specified during the master domain manager installation or perform the following steps to create a new user:
    a. Create a database user authorized to access the database and specify a password.
    b. Launch the following script:

```
<TWA_home>/TWS/dbtools/Oracle/scripts/dbgrant.bat/.sh
 <ID_of_user_to_be_granted>
 <database_name>
 <database_admin_user> <password>
```

where the variables are as follows:

<TWA_home>

The HCL Workload Automation instance directory

<ID_of_user_to_be_granted>

The ID of the user created in step 1.a on page 177, who is going to be granted the access to the reports

<database_name>

The name of the database, as created when the master domain manager was installed

<database_schema_owner> <password>

The user ID and password of the database schema owner.

2. Define a valid connection string to the database:
   a. Browse to the following path:

      On Windows operating systems

      <TWA_home>`\usr\servers\engineServer\resources\properties`

      On UNIX operating systems

      <TWA_DATA_DIR>`/usr/servers/engineServer/resources/properties`

   b. Ensure that the following property is set in the `TWSConfig.properties` file to point to the Oracle JDBC URL:

      ```
      com.ibm.tws.webui.oracleJdbcURL
      ```

      For example:

      ```
      com.ibm.tws.webui.oracleJdbcURL=
                          jdbc:oracle:thin:@//9.132.235.7:1521/orcl
      ```

      The Oracle JDBC URL is also to be specified in the **PARAM_DataSourceUrl** property in the `.\config\common.properties` file. The `common.properties` file is required when setting up for command line reporting. For more information about this file, see Setting up for command line audit reporting on page 377 and the section about setting up for command line batch reporting in *HCL Workload Automation: User's Guide and Reference*.

   c. Restart WebSphere Application Server Liberty .

Actions taken on the Dynamic Workload Console:

1. Log on to the Dynamic Workload Console.
2. In the navigation bar, select **Administration > Manage Engines**. The Manage Engines panels opens.
3. Select the engine you defined or create another engine. The Engine Connection properties panel is displayed.
4. In Database Configuration for Reporting, do the following:

   a. Check **Enable Reporting** to enable the engine connection you selected to run reports.
   b. In **Database User ID and Password**, specify the database user and password that you authorized to access reports.

Configuring for a Db2 database in HADR

Configure Db2 for generating Dynamic Workload Console reports when Db2 is configured for HADR

About this task

You can optionally configure Db2 in HADR configuration so that you can connect to the secondary database if the primary database becomes unreachable.

To configure Db2 for generating reports when Db2 is configured for HADR, perform the following steps

1. Stop WebSphere Application Server Liberty, as described in Application server - starting and stopping on page 428.
2. Browse to the following paths, depending on your operating system:

   On UNIX operating systems

   TWA_home`/usr/servers/engineServer/configDropins/templates`

   On Windows operating systems

   TWA_home`\usr\servers\engineServer\configDropins\templates`
3. Copy the `datasource.xml` file to a working location and edit it by adding the following strings:

```
<variable name="db.clientRerouteAlternateServerName" value="secondary_database"/>
<variable name="db.clientRerouteAlternatePortNumber" value="secondary_database_port_number"/>
```
4. You also need to add the call to the above variables. To perform this operation, modify the string:

```
<jndiEntry value="jdbc:db2://${db.serverName}:${db.portNumber}/${db.databaseName}" jndiName="db.url"/>
```

   as follows:

```
:clientRerouteAlternateServerName=${db.clientRerouteAlternateServerName};clientRerouteAlternatePortNumb
er=${db.clientRerouteAlternatePortNumber};"
```

   . The resulting string must be as follows:

```
<jndiEntry
 value="jdbc:db2://
${db.serverName}:${db.portNumber}/${db.databaseName}:clientRerouteAlternateServerName=${db.clientReroute
AlternateServerName};clientRerouteAlternatePortNumber=${db.clientRerouteAlternatePortNumber};"
 jndiName="db.url"/>
```
5. Copy the edited file to the following paths:

   On UNIX operating systems

   TWA_DATA_DIR`/usr/servers/engineServer/configDropins/overrides`

On Windows operating systems

TWA_home`\usr\servers\engineServer\configDropins\overrides`

6. Start WebSphere Application Server Liberty, as described in Application server - starting and stopping on page 428.

Results

You have now configured your Db2database in HADR.

For more information about HADR configuration, see An active-active high availability scenario on page 127. For more information about templates, see Configuring HCL Workload Automation using templates on page 422.

Chapter 3. Configuring user authorization (Security file)

This chapter describes how to manage the authorizations to access scheduling objects assigned to HCL Workload Automation users.

Getting started with security

The way HCL Workload Automation manages security is controlled by a configuration file named ***security file***. This file controls activities such as:

- Linking workstations.
- Accessing command-line interface programs and the Dynamic Workload Console.
- Performing operations on scheduling objects in the database or in the plan.

The security file for a fresh installation is located in the following path:

For a fresh installation of version 9.5.*x* or later

```
TWA_DATA_DIR
```

```
TWA_home\TWS
```

Upgraded environment originating from a version earlier than 9.5:

```
TWA_home/TWS
```

```
TWA_home\TWS
```

The security file contains some predefined access definitions:

- A full access definition for the user who installed the product, *TWS_user*.
- An access definition for the system administrator (root on UNIX™ or Administrator on Windows™).
- The following access definitions for the Dynamic Workload Console:
    - Analyst
    - Administrator
    - Operator
    - Developer

As you continue to work with the product, you might want to add more users with different roles and authorization to perform specific operations on a defined set of objects.

By default, the security model enabled when you perform a fresh installation is role-based. You can update your *security file* according to the role-based security model. The role-based security model allows you to update your *security file* with the security objects (domains, roles, and access control lists) that you define in the master domain manager database. You can define your security objects by using the **Manage Workload Security** interface from Dynamic Workload Console or the **composer** command-line program. The role-based security model is enabled through the setting of the **optman** enRoleBasedSecurityFileCreation global option. By default this option is set to `yes`. To use the classic security model instead, change the value to `no`. For details about updating the security file according to the role-based security model, see Role-based security model on page 182. For more information about the enRoleBasedSecurityFileCreation global option, see Global options - detailed description on page 27.

If you are upgrading HCL Workload Automation version 9.3 or earlier, you might want to continue to use the classic security model that allows you to update the security file by using dumpsec and makesec commands from the command line. To continue to use the classic security model, the enRoleBasedSecurityFileCreation global option must be set to `no`. A new security file is then created and updated with the security objects (domains, roles, and access control lists) that you define in the master domain manager database by using the **Manage Workload Security** interface from Dynamic Workload Console or the **composer** command-line program. For details about updating the security file according to the classic security model, see Classic security model on page 208.

Changes to enRoleBasedSecurityFileCreation global option are effective immediately. For details about the enRoleBasedSecurityFileCreation global option, see Global options - detailed description on page 27.

Note: The role-based security model and the classic security model are mutually exclusive.

Starting from version 9.5, Fix Pack 3, the term **$SLAVES**, which applies to all fault-tolerant agents in both the classic and role-based security models, was replaced with the term **$AGENTS** with the same scope. No change is required to your existing scripts nor environments.

Role-based security model

The security objects that you define by using the **Manage Workload Security** interface from Dynamic Workload Console, or the **composer** command-line program, are:

Access control lists

Each access control list is defined assigning roles to users or groups, on a specific security domain or folder.

Folders

Each folder has its own level of authorization that defines the set of actions that users or groups can perform on each folder.

Security roles

Each role represents a certain level of authorization and includes the set of actions that users or groups can perform.

Security domains

Each domain represents the set of scheduling objects that users or groups can manage.

You save the definitions of your security objects in the master domain manager database. If the role-based security model is enabled for your system (see Getting started with security on page 181), whenever you need to update the security objects, your *security file* is updated and converted into an encrypted format (for performance and security), replacing the previous file. The system uses this encrypted *security file* from that point onwards.

Each time a user runs HCL Workload Automation programs, commands, and user interfaces, the product compares the name of the user with the user definitions in the *security file* to determine if the user has permission to perform those activities, on the specified scheduling objects, in a certain security domain.

When the security file is updated on the master domain manager, the security settings on the master domain manager are automatically synchronized with the backup master domain manager.

Note: The role-based security model does not support centralized security management on fault-tolerant agents. On fault-tolerant agents, the security is managed locally on each workstation.

Configuring role-based security from Dynamic Workload Console

About this task

This section explains how to create and modify the security objects by using the **Manage Workload Security** interface from Dynamic Workload Console.

To create or modify security objects, you must have permission for the **modify** action on the object type **file** with attribute **name=security**.

When working with the role-based security from the Dynamic Workload Console, be aware that access to security objects is controlled by an "optimistic locking" policy. When a security object is accessed by user "A", it is not actually locked. The security object is locked only when the object update is saved by user "A", and then it is unlocked immediately afterward. If in the meantime, the object is accessed also by user "B", he receives a warning message saying that the object has just been updated by user "A", and asking him if he wants to override the changes made by user "A", or refresh the object and make his changes to the updated object.

Managing access control list

About this task

Create an access control list by assigning security roles to users or groups, in a certain security domain or in one or more folders.

You can:

- Give access to user or group.
- View access for user or group.
- View access for Security Domain or folders.
- Manage accesses.

Give access to user or group

About this task

To give access to users or groups complete the following procedure:

1. From the navigation toolbar, click **Administration**.
2. In the **Workload Environment Design**, select **Manage Workload Security**.
   Result
   The Manage Workload Security panel opens.
3. From the drop-down list, select the HCL Workload Automation engine on which you want to manage security settings.
4. In the Access Control List section, click **Give access to user or group**.
   Result
   The Create Access Control List panel opens.
5. Enter the user name or the group name, the assigned roles, and the security domain or enter the folders assigned. For each Access Control List you can associate one or more folders.
6. Click **Save** to save the access definition in the database.
7. Click **Save and Create New** to save the access definition in the database and proceed to create a new access definition.
8. Click **Save and Exit** to save the access definition in the database and return to the Manage Workload Security panel.

Results

The access definition has now been added to the database. If the **optman** enRoleBasedSecurityFileCreation global option is set to *yes*, the access definition is activated in your security file.

View access for user or group

About this task

From Manage Workload Security, you can also view the access for users or groups.

1. In the Access Control List section of the Manage Workload Security panel, click **View access for user or group**.
   Result
   The input field for the user or group name is displayed.
2. Enter the user or group name and click **View**.
   Result
   The user or group access, with the assigned roles, to the related security domains is displayed.

View access for Security Domain

About this task

From Manage Workload Security, you can also view the access to a certain security domain.

1. In the Access Control section of the Manage Workload Security panel, click **View access for Security Domain**.
   Result
   The input field for the security domain name is displayed.
2. Enter the security domain name and click **View**.
   Result
   The list of users or groups, with the assigned roles, that have access to the specified security domain is displayed.

Manage accesses

About this task

From Manage Workload Security, you can also **remove** and **edit** existing access control lists.

1. In the Access Control List section of the Manage Workload Security panel, click **Manage Accesses**.
   Result
   The list of users or groups, with the assigned roles, that have access to the different security domains is displayed.
2. Select the access control list that you want to manage.
3. Select the action that you want to run on the selected access control list.

   If you select the **edit** action, you can change only the roles associated with the access control list. You cannot change the associated domain. If you want to change the domain, you must **remove** the access control list and redefine the access control list with a new domain.

Managing folders

About this task

Folders help you to organize jobs and job streams into different categories. You can create folders with different levels of authorization that define the set of actions that users or groups can perform on each folder. More than one folder can be associated to the same Access Control List, and the level of security is also applied to the sub-folders.

You can also grant a user administrator privileges on a folder and its sub-folders so that this user can then create access control lists, with a dedicated role to manage the objects contained in the folder. See Granting administrator permissions to a user on a folder on page 186.

Creating, renaming, or deleting a folder

About this task
To create, rename, or delete a folder:

1. From the navigation toolbar, click **Administration**.
2. In the **Security**, select **Manage Workload Security**.
3. From the drop-down list, select the HCL Workload Automation engine on which you want to manage security settings.
   Result
   The Manage Workload Security panel opens.
4. In the folders section, click **Manage Folder**.
   Result
   The **Manage Folders** panel opens. From this panel you can:
   - Use the search box to search folders and job streams in the current view.
   - Create a folder or subfolder, rename or delete a folder.

Granting administrator permissions to a user on a folder

About this task

The HCL Workload Automation administrator can grant administrator permissions to a user on a folder so that the user can freely define access control lists for other users on the same folder or any sub-folders. Users can then access the objects in the folder or sub-folders in accordance with the access permissions they have on the objects.

Tip: Users with the FULLCONTROL security role assigned automatically have administrator rights on folders.

The following scenario demonstrates how Tim, the HCL Workload Automation administrator, grants Linda, the application administrator (`app1_admin` user), permissions on the folder named `/PRD/APP1/`, and how Linda grants access to Alex, the application user, to work with the objects defined in `/PRD/APP1/FINANCE`:

1. Tim, the HCL Workload Automation administrator, grants Linda, the `app1_admin` user, administrator permissions on the folder, `/PRD/APP1/`, through the definition of an access control list and by modifying her currently assigned role, `APPADMIN`. Optionally, Tim can create a new role with the appropriate permissions to achieve the same result.
    a. From the **Manage Workload Security** page, Tim selects **Manage roles**.
    b. He then selects her current role from the list, `APPADMIN` and clicks **Edit**.
    c. He gives this role administrator permissions on folders by selecting **Delegate folder permission (folder - acl)** in the **Administrative Tasks** section and clicks **Save and Exit**.
    d. Tim then creates an access control list for Linda, the `app1_admin` user. From the **Manage Workload Security** page, Tim selects **Give access to users or groups**.
    e. From the **Create Access Control** List page, Tim selects **User name** from the drop-down and enters Linda's user name, `app1_admin` in the text box.
    f. In the **Role** text box, Tim enters the `APPADMIN` role he modified earlier.
    g. In the text box next to the **Folder** selection, Tim enters the folder path of the folders on which he wants to grant Linda permissions, `/PRD/APP1/`.

    Result

    Linda, the `app1_admin` user, with the `APPADMIN` role assigned, can now access the entire `/PRD/APP1/` hierarchy, can create new folders in this path, and can assign access to these folders to other users.
2. Linda needs to give application users such as Alex, access to the objects in the `/PRD/APP1/FINANCE` folder. She creates a new access control list on the folder for the application user and assigns a role to this user.
    a. From the **Manage Workload Security** page, Linda selects **Give access to users or groups** from the **Access Control List** section.
    b. On the **Create Access Control List** page, Linda selects **User name** from the drop-down and enters the user name for Alex, the application user, `app1_user`.
    c. Since Linda cannot create new roles, she specifies an existing role in the Role text box. Only Tim, the HCL Workload Automation administrator, can create new roles.
    d. In the text box next to the **Folder** selection, Linda enters the folder path to the new sub-folder she created and to which Alex requires access: `/PRD/APP1/FINANCE`.

    Result

    Alex now can access the `/PRD/APP1/FINANCE` folder. He does not have access permissions on the `/PRD/APP1` folder.

Managing security domains

Managing security domains

About this task

A security domain represents the set of objects that users or groups can manage. For example, you can define a domain that contains all objects named with a prefix 'AA'. If you want to specify different security attributes for some or all of your users, you can create additional security domains based on specific matching criteria.

You can filter objects by specifying one or more attributes for each security object type. You can include or exclude each attribute from the selection. For example, you can restrict access to a set of objects having the same name or being defined on the same workstation, or both.

For the attributes that you can specify for each security object type, seeAttributes for security object types on page 203.

For the values that you can specify for each object attribute, see Specifying object attribute values on page 204.

You can create new security domains or manage existing security domains.

Create new security domain

About this task

To create a new security domain from the Dynamic Workload Console, complete the following procedure:

1. From the navigation toolbar, click **Administration**.
2. In the **Security**, select **Manage Workload Security** .

   Result

   The Manage Workload Security panel opens.
3. From the drop-down list, select the HCL Workload Automation engine on which you want to manage security settings.
4. In the Security Domains section, click **Create new Security Domain**.

   Result

   The security domain creation panel opens.
5. Enter the name of the security domain that you are creating and, optionally, the domain description.
6. Select the type of security domain that you want to define:

   Simple

   To define a filtering rule that applies to all object types. Events and actions are excluded from this filtering rule.

   Complex

   To define different filtering rules for different object types.
7. Use object filtering to select the set of security objects that users or groups can manage in the security domains that you are defining. You can use the wildcard character (*) when defining object attributes.

8. Click **View** to see the mapping between the set of security objects that you are assigning to the domain and the corresponding set of security objects in the classic security model.
9. Click **Save** to save the security domain definition in the database.
10. Click **Save and Exit** to save the security domain definition in the database and then exit.

Results

The security domain has now been added to the database. If the **optman**enRoleBasedSecurityFileCreation global option is set to *yes*, the security domain is activated in your security file.

Edit security domain

About this task

From Manage Workload Security, you can also **remove**, **edit**, and **duplicate** existing security domains.

1. In the Security Domains section of the Manage Workload Security panel, click **Manage Security Domain**.
   Result
   The list of the available security domains is displayed.
2. Select the security domains that you want to manage.
3. Select the action that you want to run on the selected security domains.

Managing security roles

About this task

A security role represents a certain level of authorization and includes the set of actions that users or groups can perform on a set of object types.

For the list of actions that users or groups can perform on the different objects, for each HCL Workload Automation task, see Actions on security objects on page 197.

A set of predefined security roles is available in the master domain manager database after the product has been installed:

- A full access definition for the user who installed the product, TWS_user with the default security role assigned named `FULLCONTROL`.
- An access definition for the system administrator, root on UNIX or Administrator on Windows.

You can create new security roles or manage existing security roles.

Create new role

About this task

To create a new security role from the Dynamic Workload Console, complete the following procedure:

1. From the navigation toolbar, click **Administration**.
2. In the **Security** select **Manage Workload Security**.

   Result

   The Manage Workload Security panel opens.
3. From the drop-down list, select the HCL Workload Automation engine on which you want to manage security settings.
4. In the Roles section, click **Create new role**.

   Result

   The Create Role panel opens.
5. Enter the name of the security role that you are creating and, optionally, the role description.
6. For each of the HCL Workload Automation tasks, assign the level of access for performing certain actions on specific object types to the security role. You can assign a predefined or a custom level of access.
7. Click **Show Details** to see the permissions associated to a predefined level of access, or to define your custom level of access. Tooltips are available to explain what a certain permission means for a particular object type.
8. Click **View** to see the mapping between the set of permissions that you are assigning and the corresponding set of permissions in the classic security model.
9. Click **Save** to save the security role definition in the database.
10. Click **Save and Exit** to save the security role definition in the database and return to the Manage Workload Security panel.

Results

The security role has now been added to the database. If the **optman** enRoleBasedSecurityFileCreation global option is set to *yes*, the security role is activated in your security file.

Manage roles

About this task

From Manage Workload Security, you can also **remove**, **edit**, and **duplicate** existing roles.

1. In the Roles section of the Manage Workload Security panel, click **Manage roles**.

   Result

   The list of the available security roles is displayed.
2. Select the security roles that you want to manage.
3. Select the action that you want to run on the selected roles.

Configuring role-based security with composer command-line

About this task

This section explains how to create or modify the security objects in the database, by using the **composer** command line interface.

To define security objects in the database, see:

Access control list definition on page 190

Security domain definition on page 191

Security role definition on page 194

To manage security objects in the database, see the section about managing objects with composer command-line, in the *User's Guide and Reference*.

To define or modify security objects, you must have permission for the **modify** action on the object type **file** with attribute **name=security**.

Security access control list definition

In the role-based security model, an access control list assigns security roles to users or groups, in a certain security domain or on a specific folder or folder hierarchy. You can include multiple security access control list definitions in the same text file, along with security domain definitions and security role definitions.

Each security access control list definition has the following format and arguments:

Syntax

accesscontrollist for *security_domain_name*
   user_or_group_name [*security_role*[, *security_role*]...]
  [*user_or_group_name* [*security_role*[, *security_role*]...]]...
 end

[**securitydomain** ...]

[**securityrole** ...]

accesscontrollist folder *folder_name*
   user_or_group_name [*security_role*[, *security_role*]...]
  [*user_or_group_name* [*security_role*[, *security_role*]...]]...
 end

Arguments

security_domain_name

Specifies the name of the security domain on which you are defining the access control list.

***user_or_group_name* [*security_role*[, *security_role*]**

Assigns one or more security roles to a certain user or group, on the specified security domain.

folder_name

Specifies the name of the folder to which you can associate an access control list. If the access control list is associated to a folder, then the security roles are valid for all of the objects contained in the folder. When specifying folder names, ensure you include a forward slash (/) before the folder name. Include a forward slash after the folder name to indicate that the access control list is defined only on the folder specified, excluding any sub-folders. A folder name without a final forward slash indicates that the access control list is defined on the folder, as well as on any sub-folders.

Associating an access control list to a folder is a quick and easy method to grant access to all of the objects defined in a folder. If, instead, you need to restrict access to a subset of objects in the folder (for example, objects with a certain name, or specific userlogon, cpu or jcl), then using an access control list associated to a security domain is more effective. With security domains you can filter objects by specifying one or more attributes for each security object type.

See the following composer commands documented in the *User's Guide and Reference* when working with folders: Chfolder, Listfolder, Mkfolder, Rmfolder, and Renamefolder.

Example

Examples

The following example defines:

- An access control list on the `SECDOM1` domain
- An access control list on `SECDOM2` domain
- An access control list on the folder `/FOL1/FOL2/`
- An access control list on the folder `/APPS/APP1` and any sub-folders, if present, for example, `/APPS/APP1/APP1A`.

```
ACCESSCONTROLLIST FOR SECDOM1
  USER1 SECROLE1, SECROLE2, SECROLE3
  USER2 SECROLE4
  USER3 SECROLE2, SECROLE4
END

ACCESSCONTROLLIST FOR SECDOM2
  USER1 SECROLE1, SECROLE2
  USER2 SECROLE3
END

ACCESSCONTROLLIST FOLDER /FOL1/FOL2/
             USER1 SECROLE1
END
ACCESSCONTROLLIST FOLDER /APPS/APP1
             USER1 SECROLE1
END
```

Security domain definition

In the role-based security model, a security domain represents the set of objects that users or groups can manage. For example, you can define a domain that contains all objects named with a prefix 'AA'. If you want to specify different security attributes for some or all of your users, you can create additional security domains based on specific matching criteria. You can filter objects by specifying one or more attributes for each security object type. You can include or exclude each attribute from the selection. For example, you can restrict access to a set of objects having the same name or being defined on the same workstation, or both.

You can include multiple security domain definitions in the same text file, along with security role definitions and access control list definitions.

By default, a security domain named ALLOBJECTS is available. It contains all scheduling objects and cannot be renamed nor modified. If you try to rename it, a copy of the domain is created with the new name.

Each security domain definition has the following format and arguments:

Syntax

Each attribute can be included or excluded from the selection using the plus (+) and tilde (~) symbols.

securitydomain *security_domain_name*
[**description** "*description*"]
[**common** [[+|~*object_attribute* [= *value* | @[, *value* | @]...]]]
object_type [[+|~]*object_attribute* [= *value* | @[, *value* | @]...]]
[*object_type* [[+|~]*object_attribute* [= *value* | @[, *value* | @]...]]]...
end

[**securityrole** ...]

[**accesscontrollist** ...]

Arguments

securitydomain *security_domain_name*

Specifies the name of the security domain. The name must start with a letter, and can contain alphanumeric characters, dashes, and underscores. It can contain up to 16 characters.

description *”description”*

Provides a description of the security domain. The description can contain up to 120 alphanumeric characters. The text must be enclosed within double quotes.

common [[+|~]*object_attribute* [= *value* | @[, *value* | @]...]]

Provides object attributes that are common to all the security object types.

***object_type* [[+|~]*object_attribute* [= *value* | @[, *value* | @]...]]**

For each object type, specifies the attributes that apply to that object type and the related values. Each attribute can be included or excluded from the selection using the plus (+) and tilde (~) symbols. Wildcard (@) is supported for the attribute value: *object_attribute* =@ means that all the objects matching the object attribute must be included in the domain. For the use of wildcard (@), see the examples below.

For the attributes that you can specify for each security object type, see the section about managing security with the Dynamic Workload Console, in the *Dynamic Workload Console User's Guide*.

For the values that you can specify for each object attribute, see the section about managing security with the Dynamic Workload Console, in the *Dynamic Workload Console User's Guide*.

Example

Examples

The following example defines a security domain named `SECDOM1` and a security domain named `SECDOM2`:

```
securitydomain SECDOM1
description "Sample Security Domain1"
job       cpu =   $THISCPU, # The workstation where the user logs on
                  $MASTER,  # The master workstation
                  $AGENTS,  # Any fault tolerant agent
                  $REMOTES  # Any standard agent
                  cogs@     # Any workstation whose name starts with "cogs"
         + folder = /  # Jobs defined in any folder
         + cpufolder = /  # Workstations defined in any folder
        + name =  A@        # Any job whose name starts with "A"
        ˜  name =  A2@       # but doesn't start with A2
        + jcltype = SCRIPTNAME # Allow only SCRIPTNAME type of job definition
        + jcltype = DOCOMMAND  # Allow only DOCOMMAND type of job definition
        + logon =   $USER,  # Streamlogon is the conman/composer user
                    $OWNER, # Streamlogon is the job creator
                    $JCLOWNER, # Streamlogon is the OS owner of the file
                    $JCLGROUP  # Streamlogon is the OS group of the file
        ~  logon =   root, twsuser  # The job cannot logon as "root" or "twsuser"
        + jcl   =   "/usr/local/bin/@"  # The jobs whose executable file that is
   present in /usr/local/bin
        ~  jcl   =  "@rm@" # but whose JSDL definition does not contain the
   string "rm"
end

securitydomain SECDOM2
description "Sample Security Domain2"
    common      cpu=@+name=@
    userobj     cpu=@  + cpufolder = /
    job         cpu=@+ folder = / + cpufolder = /
    schedule    cpu=@+name=AP@+ folder = / + cpufolder = /
    resource    cpu=@ + folder = / +  + cpufolder = /
    prompt      folder = /
    file        name=@
    cpu         cpu=@  + folder = /
    parameter   cpu=@  + folder = /  + cpufolder = /
    calendar    folder = /
    report      name=@
    eventrule   name=@ + folder = /
    action      provider=@
    event       provider=@
    vartable    name=@  +  folder = /
    wkldapp     name=@  +  folder = /
    runcygrp    name=@  +  folder = /
    lob         name=@
    folder      name=/
end
```

Security role definition

In the role-based security model, a security role represents a certain level of authorization and includes the set of actions that users or groups can perform. You can include multiple security role definitions in the same text file, along with security domain definitions and access control list definitions.

Each security role definition has the following format and arguments:

Syntax

securityrole *security_role_name*
  [**description** "*description*"]
    object_type **access**[=*action*[,*action*]...]
    [*object_type* **access**[=*action*[,*action*]...]]...
  end

[**securitydomain** ...]

[**accesscontrollist** ...]

Arguments

securityrole*securityrolename*

Specifies the name of the security role. The name must start with a letter, and can contain alphanumeric characters, dashes, and underscores. It can contain up to 16 characters.

description "*description*"

Provides a description of the security role. The description can contain up to 120 alphanumeric characters. The text must be enclosed within double quotes.

***object_type* access[=*action*[,*action*]...]**

For each object type, specifies a list of actions that users or groups can perform on that specific object type.

Table 25: Security object types on page 194 shows the different object types and how they are referenced with **composer** and with the Dynamic Workload Console:

Table 25. Security object types

Object type - composer	Object type - Dynamic Workload Console	Description
action	Actions	Actions defined in scheduling event rules
calendar	Calendars	User calendars
cpu	Workstations	Workstations, domains, and workstation classes
event	Events	Event conditions in scheduling event rules

Table 25. Security object types (continued)

Object type - composer	Object type - Dynamic Workload Console	Description
eventrule	Event Rules	Scheduling event rule definitions
file	Files	HCL Workload Automation database files
folder	Folders	The folder within which jobs and job streams are defined.
job	Jobs	Scheduled jobs and job definitions
parameter	Parameters	Local parameters
prompt	Prompts	Global prompts
report	Reports	The following reports in Dynamic Workload Console: **RUNHIST** Job Run History **RUNSTATS** Job Run Statistics **WWS** Workstation Workload Summary **WWR** Workstation Workload Runtimes **SQL** Custom SQL **ACTPROD** Actual production details (for current and archived plans) **PLAPROD** Planned production details (for trial and forecast plans)
resource	Resources	Scheduling resources
runcygrp	Run Cycle Groups	Run cycle groups
schedule	Job Streams	Job streams
userobj	User Objects	User objects
vartable	Variable Tables	Variable tables

Table 25. Security object types (continued)

Object type - composer	Object type - Dynamic Workload Console	Description
wkldappl	Workload Application	Workload application

Table 26: Actions that users or groups can perform on the different objects on page 196 shows the actions that users or groups can perform on the different objects.

Table 26. Actions that users or groups can perform on the different objects

Actions that users or groups can perform on the different objects			
acl	deldep	modify	stop
add	delete	release	submit
adddep	display	reply	submitdb
altpass	fence	rerun	unlink
altpri	kill	resetfta	unlock
build	limit	resource	use
cancel	link	run	
confirm	list	shutdown	
console	manage	start	

For the actions that users or groups can perform on a specific object type, for each of the HCL Workload Automation task, see the section about managing security roles with the Dynamic Workload Console, in the *Dynamic Workload Console User's Guide*.

Example

Examples

The following example defines security role `SECROLE1` and security role `SECROLE2`:

```
SECURITYROLE SECROLE1
 DESCRIPTION "Sample Security Role"
 SCHEDULE    ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,DELDEP,DELETE,
                                    DISPLAY,LIMIT,MODIFY,
  RELEASE
       RESOURCE          ACCESS=ADD,DELETE,DISPLAY,MODIFY,RESOURCE,USE,LIST,UNLOCK
       PROMPT            ACCESS=ADD,DELETE,DISPLAY,MODIFY,REPLY,USE,LIST,UNLOCK
       FILE              ACCESS=BUILD,DELETE,DISPLAY,MODIFY,UNLOCK
       FOLDER            ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK,ACL
       CPU               ACCESS=LIMIT,LINK,MODIFY,SHUTDOWN,START,STOP,UNLINK,LIST,UNLOCK,RUN
       PARAMETER         ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
       CALENDAR          ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
       REPORT            ACCESS=DISPLAY
       EVENTRULE         ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
       ACTION            ACCESS=DISPLAY,SUBMIT,USE,LIST
       EVENT             ACCESS=USE
       VARTABLE          ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
       WKLDAPPL          ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
       RUNCYGRP          ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
       LOB               ACCESS=USE
END

SECURITYROLE SECROLE2
 DESCRIPTION "Sample Security Role"
 SCHEDULE          ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,DELDEP,DELETE,
                              DISPLAY,LIMIT,MODIFY,
  RELEASE
 RESOURCE          ACCESS=ADD,DELETE,DISPLAY,MODIFY,RESOURCE,USE,LIST,UNLOCK
 PROMPT            ACCESS=ADD,DELETE,DISPLAY,MODIFY,REPLY,USE,LIST,UNLOCK
END
```

The following example defines a new security role `APP_ADMIN`, for the user `APP1_ADMIN` and assigns administrator permissions on the folder hierarchy `/PRD/APP1/`, so that the `APP1_ADMIN` user can create access control lists to give other users access to the objects in this folder or its sub-folders:

Security role definition

```
SECURITYROLE APP_ADMIN
  DESCRIPTION "Security Role"
  JOB  ADD,MODIFY,SUBMITDB,USE,ADDDEP,RUN,RELEASE,REPLY,DELETE,DISPLAY,
       CANCEL,SUBMIT,CONFIRM,RERUN,LIST,DELDEP,KILL,UNLOCK,ALTPRI
  SCHEDULE ADD,ADDDEP,ALTPRI,CANCEL,DELDEP,DELETE,DISPLAY,LIMIT,MODIFY,RELEASE
  FOLDER   ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK,ACL
```

Security file

```
USER APP_ADMINofAPP1
  CPU=@+LOGON="APP_ADMIN"
BEGIN
  JOB    FOLDER="/PRD/APP1/","/PRD/APP1" + CPUFOLDER = / ACCESS=ADD,ADDDEP,
         ALTRPRI,CANCEL,SUBMIT,
         CONFIRM,RERUN,LIST,DELDEP,KILL,UNLOCK,ALTPRI
  SCHEDULE  FOLDER="/PRD/APP1/","/PRD/APP1" + CPUFOLDER = / ACCESS=ADD,ADDDEP,
         ALTPRI,CANCEL,DELDEP,
         DELETE,DISPLAY,LIMIT,MODIFY,RELEASE
  FOLDER NAME="/PRD/APP1/","PRD/APP1"   ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,
         LIST,UNLOCK,ACL
```

Actions on security objects

The following tables show the actions that users or groups can perform on the different object types, for each HCL Workload Automation task. See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with **composer** command line interface.

Table 27. Actions that users or groups can perform when designing and monitoring the workload

Design and Monitor Workload	
Actions that users or groups can perform	**Security object types**
List (list)	Jobs (job)
Display (display)	Job Streams (schedule)
Create (add)	User Objects (userobj)
Delete (delete)	Prompts (prompt)
Modify (modify)	Resources (resource)
Use (use)	Calendars (calendar)
Unlock (unlock)	Run Cycle Groups (runcygrp)
Actions on remote workstations while modeling jobs (cpu-run)	Variable Tables (vartable) Workload Application (wkldappl)
Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with the **composer** command-line interface.	Workflow Folders (folder) Parameters (parameter)

Table 28. Actions that users or groups can perform when modifying current plan

Modify current plan
Actions that users or groups can perform on the current plan
Add job stream dependency (schedule - adddep)
Add job dependency (job - adddep)
Remove job dependency (job - deldep)
Remove job stream dependency (schedule - deldep)
Change job priority (job - altpri)
Change job stream priority (schedule - altpri)
Cancel job (job - cancel)
Cancel job stream (schedule - cancel)
Rerun job (job - rerun)

Table 28. Actions that users or groups can perform when modifying current plan (continued)

Modify current plan Actions that users or groups can perform on the current plan
Confirm job (job - confirm)
Release job (job - release)
Release job stream (schedule - release)
Kill jobs (job - kill)
Reply to prompts (prompt - reply)
Reply to job prompts (job - reply)
Reply to job stream prompts (schedule - reply)
Alter user password (userobj - altpass)
Change jobs limit (schedule - limit)
Actions on job remote system (job - run)
Change resource quantity (resource - resource)

Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with the **composer** command line interface.

Table 29. Actions that users or groups can perform when submitting workload

Submit Workload Workload definitions that can be added to the current plan
Only existing job definitions (job - submitdb)
Existing jobs definitions and ad hoc jobs (job - submit)
Existing job stream definitions (schedule - submit)

Table 29. Actions that users or groups can perform when submitting workload (continued)

Submit Workload
Workload definitions that can be added to the current plan

Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with the **composer** command line interface.

Table 30. Actions that users or groups can perform when managing the workload environment

Manage Workload Environment
Actions that users or groups can perform on workstations, domains, and workstation classes
List workstations (cpu - list)
Display workstation details (cpu - display)
Create workstations (cpu - add)
Delete workstations (cpu - delete)
Modify workstations (cpu - modify)
Use workstations (cpu - use)
Unlock workstations (cpu - unlock)
Start a workstation (cpu - start)
Stop a workstation (cpu - stop)
Change limit (cpu - limit)
Change fence (cpu - fence)
Shutdown (cpu - shutdown)
Reset FTA (cpu - resetfta)
Link (cpu - link)
Unlink (cpu - unlink)
Use 'console' command from conman (cpu - console)
Upgrade workstation (cpu - manage)

Table 30. Actions that users or groups can perform when managing the workload environment (continued)

Manage Workload Environment
Actions that users or groups can perform on workstations, domains, and workstation classes
Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with the **composer** command line interface.

Table 31. Actions that users or groups can perform when managing event rules

Manage Event Rules
Actions that users or groups can perform on event rules
List event rules (eventrule - list)
Display event rules details (eventrule - display)
Create event rules (eventrule - add)
Delete event rules (eventrule - delete)
Modify event rules (eventrule - modify)
Use event rules (eventrule - use)
Unlock event rules (eventrule - unlock)
Display actions in the event rules (action - display)
Monitor triggered actions (action - list)
Use action types in the event rules (action - use)
Submit action (action - submit)
Use events in the event rules (event - use)
Use a File Monitor event on the workstation where the file resides. (event - display)

Table 31. Actions that users or groups can perform when managing event rules (continued)

Manage Event Rules
Actions that users or groups can perform on event rules

Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with the **composer** command line interface.

Table 32. Administrative tasks that users or groups can perform

Administrative Tasks
Administrative tasks that users or groups can perform
View configuration (dump security and global options) (file - display)
Change configuration (makesec, optman add) (file - modify)
Delete objects definitions (file - delete)
Unlock objects definitions (file - unlock)
Allow planman deploy, prodsked and stageman (file - build)
Delegate security on folders (folder - acl)

Note: See in parenthesis the corresponding *action* and *object* values that you must use when defining role-based security with the **composer** command-line interface.

Table 33. Actions that users or groups can perform on workload reports

Workload Reports	
Actions that users or groups can perform on workload reports	
Generate workload reports (display report)	**Reports in Dynamic Workload Console** **RUNHIST** Job Run History **RUNSTATS** Job Run Statistics **WWS** Workstation Workload Summary **WWR** Workstation Workload Runtimes

Table 33. Actions that users or groups can perform on workload reports (continued)

Workload Reports	Actions that users or groups can perform on workload reports
	SQL Custom SQL **ACTPROD** Actual production details (for current and archived plans) **PLAPROD** Planned production details (for trial and forecast plans)

Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with the **composer** command line interface.

Table 34. Actions that users or groups can perform on folders.

Folders	Actions that users or groups can perform on folders
Access folders	chfolder (display) listfolder (list or list and display) mkfolder (modify) rmfolder (delete) renamefolder (add)

Note: See in parenthesis the corresponding *actions* and *objects* values that you must use when defining role-based security with the **composer** command line interface.

Attributes for security object types

Table 35: Attributes for security object types on page 204 shows the attributes that you can specify for each security object type (see in parenthesis the corresponding object type and object attribute that you must use when defining security objects with the **composer** command line interface).

Table 35. Attributes for security object types

Attribute Security object type	Name (name)	Workstation (cpu)	Custom (custom)	JCL (jcl)	JCLtype (jcltype)	Logon (logon)	Provider (provider)	Type (type)	Host (host)	Port (port)	Folder (folder)	CPU Folder (cpufolder)
Actions (action)							✓	✓	✓	✓		
Calendars (calendar)	✓										✓	
Workstations (cpu)	✓							✓			✓	
Events (event)			✓				✓	✓				
Event rules (eventrule)	✓										✓	
Files (file)	✓											
Jobs (job)	✓	✓		✓	✓	✓					✓	✓
Parameters (parameter)	✓	✓									✓	✓
Prompts (prompt)	✓										✓	
Reports (report)	✓											
Resource (resource)	✓	✓									✓	
RunCycle groups (runcygrp)	✓										✓	
Job streams (schedule)	✓	✓									✓	✓
User objects (userobj)		✓				✓						✓
Variable tables (vartable)	✓										✓	
Workload applications (wkldappl)	✓										✓	
Folders (folder)	✓											

For the values that are allowed for each object attribute, see Specifying object attribute values on page 204.

Specifying object attribute values

The following values are allowed for each object attribute (see in parenthesis the corresponding object type and object attribute for the **composer** command line interface):

Name (name)

Specifies one or more names for the object type.

- For the Files (file) object type, the following values apply:

  globalopts

  Allows the user to set global options with the `optman` command. The following access types are allowed:

    - Display access for `optman ls` and `optman show`
    - Modify access for `optman chg`

  prodsked

  Allows the user to create, extend, or reset the production plan.

  security

  Allows the user to manage the security file.

  Symphony

  Allows the user to run **stageman** and **JnextPlan**.

  trialsked

  Allows the user to create trial and forecast plans or to extend trial plans.

  Note: Users who have restricted access to files should be given at least the following privilege to be able to display other object types that is, Calendars (calendar) and Workstations (cpu):

  ```
  file  name=globalopts  action=display
  ```

- For the **Variable Tables (vartable)** object type, you can use the $DEFAULT value for the **Name (name)** attribute to indicate the default variable table. This selects the table that is defined with the `isdefault` attribute.

Workstation (cpu)

Specifies one or more workstation, domain, or workstation class name. Workstations and workstation classes can optionally be defined in a folder. If this attribute is not specified, all defined workstations and domains can be accessed. Workstation variables can be used:

$MASTER

The HCL Workload Automation master domain manager.

$AGENTS

Any fault-tolerant agent.

$REMOTES

Any standard agent.

$THISCPU

The workstation on which the user is running the HCL Workload Automation command or program.

If you use the **composer** command line to define security domains, the following syntax applies:

```
cpu=[folder/]workstation]...
```

folder=*foldername*

Scheduling objects such as, jobs, job streams, and workstations, to name a few, can be defined in a folder. A folder can contain one or more scheduling objects, while each object can be associated to only one folder. The default folder is the root folder (/).

cpufolder=*foldername*

The folder within which the workstation or workstation class is defined.

Custom (custom)

Use this attribute to assign access rights to events defined in event plug-ins. The precise syntax of the value depends on the plug-in. For example:

- Specify different rights for different users based on SAP R/3 event names when defining event rules for SAP R/3 events.
- Define your own security attribute for your custom-made event providers.
- Specify the type of event that is to be monitored. Every event can refer to an event provider.

If you use **composer** command line to define security domains, the following syntax applies:

```
custom=value[,value]...
```

JCL (jcl)

Specifies the command or the path name of a job object's executable file. If omitted, all defined job files and commands qualify.

You can also specify a string that is contained in the task string of a JSDL definition to be used for pattern matching.

If you use **composer** command line to define security domains, the following syntax applies:

```
jcl="path" | "command" | "jsdl"
```

JCL Type (jcltype)

Specifies that the user is allowed to act on the definitions of jobs that run only scripts (if set to scriptname) or commands (if set to docommand). Use this optional attribute to restrict user authorization to actions on the definitions of jobs of one type only. Actions are granted for both scripts and commands when JCL Type (jcltype) is missing.

A user who is not granted authorization to work on job definitions that run either a command or a script is returned a security error message when attempting to run an action on them.

If you use **composer** command line to define security domains, the following syntax applies:

```
jcltype=[scriptname | docommand]
```

Logon (logon)

Specifies the user IDs. If omitted, all user IDs qualify.

You can use the following values for the **Logon (logon)** attribute to indicate default logon:

$USER

Streamlogon is the conman/composer user.

$OWNER

Streamlogon is the job creator.

$JCLOWNER

Streamlogon is the OS owner of the file.

$JCLGROUP

Streamlogon is the OS group of the file.

If you use **composer** command line to define security domains, the following syntax applies:

```
logon=username[,username]...
```

Provider (provider)

For **Actions (action)** object types, specifies the name of the action provider.

For **Events (event)** object types, specifies the name of the event provider.

If **Provider (provider)** is not specified, no defined objects can be accessed.

If you use **composer** command line to define security domains, the following syntax applies:

```
provider=provider_name[,provider_name]...
```

Type (type)

For **Actions (action)** object types, is the `actionType`.

For **Events (event)** object types, is the `eventType`.

For **Workstations (cpu)** object types, the permitted values are those used in composer or the Dynamic Workload Console when defining workstations, such as `manager`, `broker`, `fta`, `agent`, `s-agent`, `x-agent`, `rem-eng`, `pool`, `d-pool`, `cpuclass`, and `domain`.

Note: The value `master`, used in conman is mapped against the `manager` security attributes.

If **Type (type)** is not specified, all defined objects are accessed for the specified providers (this is always the case after installation or upgrade, as the type attribute is not supplied by default).

If you use **composer** command line to define security domains, the following syntax applies:

```
type=type[,type]...
```

Host (host)

For **Actions (action)** object types, specifies the TEC or SNMP host name (used for some types of actions, such as sending TEC events, or sending SNMP). If it does not apply, this field must be empty.

If you use **composer** command line to define security domains, the following syntax applies:

```
host=host_name
```

Port (port)

For **Actions (action)** object types, specifies the TEC or SNMP port number (used for some types of actions, such as sending TEC events, or sending SNMP). If it does not apply, this field must be empty.

If you use **composer** command line to define security domains, the following syntax applies:

```
port=port_number
```

Classic security model

A template file named `TWA_home/TWS/config/Security.conf` is provided with the product. During installation, a copy of the template file is installed as `TWA_home/TWS/Security.conf`, and a compiled, operational copy is installed as `TWA_home/TWS/Security`.

This version of the file contains a full access definition for the user who installed the product, *TWS_user*, and the system administrator (root on UNIX™ or Administrator on Windows™), who are the only users defined and allowed to connect to the user interfaces and to perform all operations on all scheduling resources.

Within the HCL Workload Automation network, using the security file you can make a distinction between local **root** users and the **root** user on the master domain manager by allowing local **root** users to perform operations affecting only their login workstations and providing the master domain manager **root** user the authorizations to perform operations affecting any workstation across the network.

As you continue to work with the product you might want to add more users with different roles and authorization to perform specific operations on a defined set of objects.

Do not edit the original `TWA_home/TWS/config/Security.conf` template, but follow the steps described in Updating the security file on page 209 to make your modifications on the operational copy of the file.

Security management overview

The way HCL Workload Automation manages security is controlled by a configuration file named ***security file***. This file controls activities such as:

- Linking workstations.
- Accessing command-line interface programs and the Dynamic Workload Console.
- Performing operations on scheduling objects in the database or in the plan.

In the file you specify for each user what scheduling objects the user is allowed to access, and what actions the user is allowed to perform on those objects. You can determine access by object type (for example, workstations or resources) and, within an object type, by selected attributes, such as the object's name or the workstation in the object's definition. You can use wildcards to select related sets of objects. Access rights can be granted on an "included" or an "excluded" basis, or a combination of both.

Whenever you need to change access permissions you modify the configuration file and convert it into an encrypted format (for performance and security), replacing the previous file. The system uses this encrypted *security file* from that point onwards.

Each time a user runs HCL Workload Automation programs, commands, and user interfaces, the product compares the name of the user with the user definitions in the *security file* to determine if the user has permission to perform those activities on the specified scheduling objects.

By default, the security on scheduling objects is managed locally on each workstation. This means that the system administrator or the *TWS_user* who installed the product on that system can decide which HCL Workload Automation users defined on that system can access which scheduling resources in the HCL Workload Automation network and what actions they can perform.

Alternatively, you can centralize control of how objects are managed on each workstation. This can be configured by setting the **enCentSec** global option. In this scenario, you configure all user permissions in the *security file* on the master domain manager. The encrypted version of the file is distributed automatically every time you run **JnextPlan**, so that all workstations have the file locally to determine the permissions of the users on that workstation.

For more information about centralized security, see Centralized security management on page 213. For more information about global options, see Global options - detailed description on page 27.

Updating the security file

About this task

By default, every workstation in an HCL Workload Automation network (domain managers, fault-tolerant agents, and standard agents) has its own security file. You can maintain that file on each workstation, or, if you enable centralized security management, you can create a security file on the master domain manager and copy it to each domain manager and agent, ensuring that all HCL Workload Automation users are assigned the required authorization in the file (see Centralized security management on page 213). Whether working on an agent workstation for an individual security file, or on the master domain manager to modify a centralized file, the steps are just the same; all that changes are the number of users you are defining - just those on the local system or all in the HCL Workload Automation network.

If you are updating or upgrading your fault-tolerant agents to version 9.5 Fix Pack 2 or later, you must manually update the security file on the fault-tolerant agents in your environment to add access to folders for all of the scheduling objects that can be defined or moved into folders. These updates are especially important if you plan to use the command line on the fault-tolerant agents to perform operations on the objects in folders. More specifically, if centralized security is enabled (enCentSec / ts = YES), then you must first update or upgrade all of the fault-tolerant agents in your environment to version 9.5 Fix Pack 2 or later before you begin using the folder feature. If centralized security management is not enabled (enCentSec / ts = NO), all stanzas that reference CPU are automatically updated to include folder access, for example, `CPU CPU=@+FOLDER="/"`, unless you use wildcard characters (@) as matching criteria in your stanzas, for example, `CPU CPU=@HR`. In this case, if you want to be able to move these CPUs into folders, then you need to manually update those stanzas to include access to all folders, `CPU CPU=@HR+FOLDER="/"`.

Neither the HCL Workload Automation processes nor the WebSphere Application Server Liberty infrastructure needs to be stopped or restarted to update the security file. You just need to close any open conman user interfaces before running makesec.

To modify the security file, perform the following steps:

1. Configure the environment, running one of the following scripts:

   In UNIX®:

   - `. ./TWA_home/TWS/tws_env.sh` for Bourne and Korn shells
   - `. ./TWA_home/TWS/tws_env.csh` for C shells

   In Windows®:

   - `TWA_home\TWS\tws_env.cmd`
2. Navigate to the following directory from where you can submit the dumpsec and makesec commands:

   ```
   TWA_home/TWS
   ```

   ```
   TWA_home\TWS
   ```
3. Run the dumpsec command to decrypt the current security file into an editable configuration file. See dumpsec on page 211.
4. Modify the contents of the editable security configuration file using the syntax described in Configuring the security file on page 214.
5. Close any open conman user interfaces using the **exit** command.
6. Stop any connectors on systems running Windows™ operating systems.
7. Run the makesec command to encrypt the security file and apply the modifications. See makesec on page 212.
8. If you are using local security, the file will be immediately available on the workstation where it has been updated.

   If you are using centralized security (see Centralized security management on page 213), perform the following steps:

   a. If you are using a backup master domain manager, copy the file to it.
   b. Distribute the centralized file manually to all fault-tolerant agents in the network (not standard, extended, or broker agents), and store it in the following directory:

      For a fresh installation of version 9.5.*x* or later

      ```
      TWA_DATA_DIR
      ```

      ```
      TWA_home\TWS
      ```

Upgraded environment originating from a version earlier than 9.5:

```
TWA_home/TWS
```

```
TWA_home\TWS
```

c. Run JnextPlan to distribute the Symphony file that corresponds to the new Security file.

See dumpsec on page 211 and makesec on page 212 for a full description of the commands.

dumpsec

Writes in an editable format the information contained in the compiled and encrypted security file. The output file can be edited and then used as input for the **makesec** command which compiles and activates the modified security settings.

Authorization

You must have ***display*** access to the security file and write permission in the `TWA_home/TWS` directory from where the command *must* be run.

Syntax

dumpsec –v | –u

dumpsec *security_file* [**>** *output_file*]

Comments

If no arguments are specified, the operational security file is sent to stdout. To create an editable copy of the security file, redirect the output of the command to an output file, using the redirect symbol.

Arguments

–v

Displays command version information only.

–u

Displays command usage information only.

security_file

Specifies the name of the security file to dump.

[> *output_file*]

Specifies the name of the output file, If omitted, the security file is output to the stdout.

Example

Examples

The following command dumps the operational security file (`TWA_home/TWS/Security`) to a file named **mysec**:

```
dumpsec > mysec
```

The following command dumps a security file named **sectemp** to **stdout**:

```
dumpsec sectemp
```

makesec

Compiles security definitions and installs the security file. Changes to the security file are recognized as soon as makesec has completed, or, in the case of centralized security, after **JnextPlan** has distributed it.

Note: Before running the **makesec** command, stop conman, and, on systems running Windows® operating systems, any connectors.

Authorization

You must have ***modify*** access to the security file and read permission in the *TWA_home*`/TWS` directory from where the command *must* be run.

Syntax

makesec –v | –u

makesec [**–verify**] *in_file*

Comments

The **makesec** command compiles the specified file and installs it as the operational security file (`../`*TWA_home*`/TWS/Security`). If the **–verify** argument is specified, the file is checked for correct syntax, but it is not compiled and installed.

Arguments

–v

Displays command version information only.

–u

Displays command usage information only.

–verify

Checks the syntax of the user definitions in *in_file*. The file is not compiled and installed as the security file.

in_file

Specifies the name of a file or set of files containing user definitions. Syntax checking is performed automatically when the security file is installed.

Example

Examples

Example 1: Modifying the security file definitions - full scenario

The following example shows how to modify the security file definitions:

1. An editable copy of the operational security file is created in a file named `tempsec` with the dumpsec command:

   ```
   dumpsec > tempsec
   ```
2. The user definitions are modified with a text editor:

   ```
   edit tempsec
   ```
3. The file is then compiled and installed with the **makesec** command:

   ```
   makesec tempsec
   ```

Example 2: Compiling user definitions from multiple files

The following command compiles user definitions from the fileset `userdef*` and replaces the operational security file:

```
makesec userdef*
```

Centralized security management

An HCL Workload Automation environment where centralized security management is enabled is an environment where all workstations share the same security file information contained in the security file stored on the master domain manager and the HCL Workload Automation administrator on the master domain manager is the only one who can add, modify, and delete entries in the security file valid for the entire HCL Workload Automation environment.

This is configured with the ***enCentSec*** global option. By default the value assigned to the ***enCentSec*** option is **no**.

To set central security management, the HCL Workload Automation administrator must run the following steps on the master domain manager:

1. Use the **optman** command line program, to set the value assigned to the *enCentSec* global property to **yes**. For information on how to manage the global properties using optman, see Setting global options on page 12.
2. Save the information in the security file into an editable configuration file using the **dumpsec on page 211** command.
3. Set the required authorizations for all HCL Workload Automation users, as described in Configuring the security file on page 214
4. Close any open conman user interfaces using the **exit** command.
5. Stop any connectors on systems running Windows® operating systems.
6. Compile the security file using the **makesec on page 212** command.
7. If you are using a backup master domain manager, copy the compiled security file to it as soon as possible.
8. Distribute the compiled security file to all the workstations in the environment and store it in their `*TWA_home*/TWS` directories.
9. Run **JnextPlan** to update the security information distributed with the `Symphony` file.

The value of the checksum of the newly compiled security file is encrypted and loaded into the `Symphony` file and distributed to all the workstations in the HCL Workload Automation network.

On each workstation, when a link is established or when a user connects to a user interface or attempts to issue commands on the plan, either with **conman** or the Dynamic Workload Console, HCL Workload Automation compares the value of the checksum in the security file delivered with the `Symphony` file with the value of the checksum of the security file stored on the workstation. If the values are equal, the operation is allowed. If the values are different, the operation fails and a security violation message is issued.

Centralized security usage notes

The following are some considerations to be aware of if centralized security management is enabled in your HCL Workload Automation environment.

When centralized security is enabled (enCentSec / ts = YES), and you plan to start using the folder feature to define or move scheduling objects into dedicated folders, then you must first update or upgrade all of the fault-tolerant agents in your environment to version 9.5 Fix Pack 2 or later. If centralized security management is not enabled (enCentSec / ts = NO), all stanzas that reference CPU are automatically updated to include folder access, for example, `CPU CPU=@+FOLDER="/"`, unless you use wildcard characters (@) as matching criteria in your stanzas, for example, `CPU CPU=@HR`. In this case, if you want to be able to move these CPUs into folders, then you need to manually update those stanzas to include access to all folders, `CPU CPU=@HR+FOLDER="/"`.

In a network with centralized security management, two workstations are unable to establish a connection if one of them has ***enCentSec*** turned off in its `Symphony` file or if their security file information does not match.

The only exception to the security file matching criteria introduced by the centralized security management mechanism is that a workstation must always accept incoming connections from its domain manager, regardless of the result of the security file matching process.

Centralized security does not apply to HCL Workload Automation operations for which the `Symphony` file is not required. Commands that do not require the `Symphony` file to run use the local security file. For example, the parms command, used to modify or display the local parameters database, continues to work according to the local security file, even if centralized security is active and the local security file differs from the centralized security rules.

If a workstation's security file is deleted and re-created, the checksum of the new security file will not match the value in the `Symphony` file. In addition, a run-number mechanism associated with the creation process of the `Symphony` file ensures prevention from tampering with the file.

Configuring the security file

In the security file you can specify which scheduling objects a user can manage and how. You define these settings by writing user definitions. A user definition is an association between a name and a set of users, the objects they can access, and the actions they can perform on the specified objects.

When defining user authorization consider that:

- When commands are issued from the **composer** command line program, the user authorizations are checked in the security file of the master domain manager since the methods used to manage the entries in the database are invoked on the master domain manager. Therefore the user must be defined:
    - As system user on the system where the master domain manager is installed.
    - In the security file on the master domain manager with the authorizations needed to run the allowed commands on the specific objects.
- When commands are issued from the **conman** command line program, the user must be authorized to run the specific commands in the security file both on the connecting workstation and on the master domain manager where the command actually runs.

The security file is parsed one line at a time, thus any given line in the security file has been assigned a maximum length of 32768 characters. Since during the encryption process (makesec), one extra character is added to any string value in order to store its length, the number of "visible" characters could actually be more or less than 32768. As an example, consider the following line:

```
CPU=@+LOGON=test1, test2
```

The actual number of characters written into the encrypted Security file is determined according to this formula:

```

"CPU=" : 2 chars (token)
"@" : 2 chars (1 + 1 for the length)
"LOGON=" : 2 chars (token)
"test1," : 7 chars (6 + 1 for the length) (string)
"test2" : 6 chars (5 + 1 for the length) (string)
--------------------------------------------------
total : 19 chars
```

However, if counting the actual number of visible characters, there are 23 characters including the single space between test1 and test2 and the comma separating them.

Note: The "CPU" and "LOGON" each have a real length of two characters even though they actually have three and five characters respectively. This is because certain keywords are "tokenized." This can actually help reduce the apparent character count in this case.

The configuration of the security file is described in these sections:

- Security file syntax on page 215
- Specifying user attributes on page 217
- Specifying object types on page 224
- Specifying object attributes on page 226
- Specifying access on page 231
- The TWS_user - special security file considerations on page 254

Security file syntax

The syntax of the security file is as follows:

Syntax

[# *comment*]

user *definition_name user_attributes*

begin [* *comment*]

object_type [*object_attributes*]. **access**[**=***keyword*[**,***keyword*]...]

[*object_type* [*object_attributes*]. **access**[**=***keyword*[**,***keyword*]...]]...

end | **continue**

Arguments

[# | *] *comment*

All text following a pound sign or an asterisk and at least one space is treated as a comment. Comments are not copied into the operational security file installed by the **makesec** command.

user *definition_name*

Specifies the name of the user definition. The name can contain up to 36 alphanumeric characters and must start with an alphabetic character.

user_attributes

Contains one or more attributes that identify the user or users to whom the definition applies. For details of how to define user attributes, see Specifying user attributes on page 217.

begin

Begins the part containing object statements and accesses within the user definition.

object_type

Identifies the type of object (for example: workstation, resource, or prompt) to which access is to be given for the specified user or users. All object types that the specified user or users needs to access must be explicitly defined. If they are not, no access will be given. For details of how to define object types, see Specifying object types on page 224.

object_attributes

Contains one or more attributes that identify the specific objects of the defined object type to which the same access is to be given. If no object attributes are defined, access is given to all objects of the defined object type. For details of how to define object attributes, see Specifying object attributes on page 226.

access[=*keyword*[,*keyword*]...]

Describes the access to the specified objects given to the selected users. If none is specified (by specifying just the keyword "access") no access is given to the associated objects. If **access=@** then all access rights are assigned to the specified users. For details of how to define access, see Specifying access on page 231.

continue

Allows a user to inherit authorization from multiple *stanzas*. Add the `Continue` keyword before the `Begin` keyword of each subsequent *stanza* to request that HCL Workload Automation must not stop at the first *stanza*, but must continue including also the following *stanzas* that match the user definition. The user gets the accesses for the first matching entry of each *stanza*. For an example of the use of the `Continue` keyword, see Users logged into multiple groups [continue keyword] on page 260.

end

Terminates the user definition. The users defined in the user definition that terminates with an `end` statement do not match any subsequent user definition.

Wildcards

The following wildcard characters are permitted in user definition syntax:

?

Replaces one alphanumeric character.

@

Replaces zero or more alphanumeric characters.

For information about variables supplied with the product that can be used in object attributes, refer to Using variables in object attribute definitions on page 231. Refer to Sample security file on page 255 for an example on how to use variables.

Specifying user attributes

The user attributes define *who* has the access that is going to be subsequently defined. They can identify one user, a selection of users, a group of users, a selection of groups of users, or all users. You can also exclude one or more specific users or groups from a selection. As well as being identified by logon ID and group name, users can also be described by the workstation from which they log on. And finally, you can mix selection criteria, for example selecting all users in a named group that can access from a set of workstations identified by a wildcard, but excluding a specific set of users identified by their logon IDs.

A user must be uniquely identified. If different users have the same identifier, an error is issued when makesec command is run. You must edit the security file by using dumpsec command, assign a unique identifier to users, and rerun the makesec command.

The general syntax

You make this selection by specifying one or more user attributes. Each user attribute is specified as follows:

*user_attribute_type***=***value*

user_attribute_type

Can be *cpu* (workstation), *group*, or *logon*

value

Identifies an individual *cpu* (workstation), *group*, or *logon*, or, by using wildcards, can identify a set of any of these.

Including or excluding

Each attribute can be *included* or *excluded* from the selection.

Thus, for each *attribute type*, your options are one of the following:

Include all

This is the default. Thus, for example, if you want to include all *groups*, you need add no user attribute with respect to any group.

Include a selection

This can be defined in one of these ways:

- By specifically including users you want to select (individuals or one or more sets)
- By specifically excluding (from the *include all* default) all users you do *not* want to select
- By specifically including a set of users and then excluding some of those contained in the set

Which of these options you choose is determined by which is easier to specify.

Using the include or exclude symbols

Include

Precede the user attribute expression by a plus (+) sign. All users identified by the expression will be selected, unless they are also selected by an *exclude* expression. If the first attribute in your definition is an *include*, it does not need to have a (+) sign defined, because the sign is implicit.

The default (if you specify no user attributes) is to include all users, on all workstations, in all groups, so if you want to define, for example, all users except one named user, you would just supply the *exclude* definition for the one user.

Exclude

Precede the user attribute expression by a tilde (~) sign. All users identified by the expression will *never* be selected, regardless of if they are identified by any *include* expressions.

Selection expressions

You can use the following different types of selection expression:

Basis selection expressions

Include only one attribute

user_attribute_type=*value*

For example, to include one named user logon ID, and exclude all other users:

```
logon=jsmith1
```

Exclude one attribute

~*user_attribute_type*=*value*

For example, to exclude one set of logon IDs identified by a wildcard (those that start with the letter "j"), but include all others:

```
~logon=j@
```

Include only several attributes of the same type

user_attribute_type=*value*[**,***value*]...

For example, to include three specific users and exclude all others:

```
logon=jsmith1,jbrown1,jjones1
```

Exclude several attributes of the same type

~*user_attribute_type*=*value*[**,***value*]...

For example, to exclude three specific users and include all others:

```
~logon=jsmith1,jbrown1,jjones1
```

Complex selection expressions

Include users identified by different selection expressions

basic_selection_expression[**+***basic_selection_expression*]...

The selection expressions can be of the same or a different attribute type:

Same attribute type

An example of the same attribute type is the following, which selects all the groups beginning with the letter "j", as well as those with the letter "z":

```
group=j@+group=z@
```

If the first selection identifies 200 users, and the second 300, the total users selected is 500.

Different attribute type

An example of selection expressions of a different attribute type is the following, which selects all the groups beginning with the letter "j", as well as all users with IDs beginning with a "6":

```
group=j@+logon=6@
```

If the first selection identifies 200 users, and the second 20, of whom 5 are also in the first group, the total users selected is 5.

Exclude users identified in one selection expressions from those identified in another

basic_selection_expression[**~***basic_selection_expression*]...

Same attribute type

The selection expressions can be of the same attribute type, provided that the second is a subset of the first. An example of the same attribute type is the following, which selects all the workstations beginning with the letter "j", but excludes those with a "z" as a second letter:

```
group=j@~group=jz@
```

If the first selection identifies 200 users, and the second 20, the total users selected is 180. Note that if the second expression had not been a subset of the first, the second expression would have been ignored.

Different attribute type

Selection expressions of a different attribute type do not have to have a subset relationship, an example being the following, which selects the group "mygroup", but excludes from the selection all users in the group with IDs beginning with a "6":

```
group=mygroup~logon=6@
```

If the first selection identifies 200 users, and the second 20, of whom 5 are also in the first group, the total users selected is 195.

Multiple includes and excludes

You can link together as many include and exclude expressions as you need to identify the precise subset of users who require the same access. The overall syntax is thus:

[**~**]*user_attribute_type***=***value*[**,***value*]... [{**+**|**~**}*user_attribute_type***=***value*[**,***value*]...

Note: Making your *first* user attribute an *exclude* means that *all* user attributes of that type are selected *except* the indicated *value*. Thus, **~***user_attribute_type*=*value* equates to the following:

*user_attribute_type***=@~***same_user_attribute_type*=*value*

However, if you use this syntax, you cannot, and do not need to, specifically add "**+***user_attribute_type***=**@", after the negated item, so you do not define:

*~user_attribute_type=value***+***same_user_attribute_type=@*

Order of user definition

You must order user definitions from most specific to least specific. HCL Workload Automation scans the security file from top-down, with each user ID being tested against each definition in turn. If the user ID is satisfied by the definition, it is selected, and the matching stops.

For example:

Incorrect:

```
#First User Definition in the Security File
USER TwsUser
CPU=@+LOGON=TWS_user
Begin
job name=@ access=modify
End

#Second User Definition in the Security File
USER Twsdomain:TwsUser
CPU=@+LOGON=TWSDomain\\TWS_user
Begin
job name=@ access=display
End
```

The definitions are intended to determine the following:

1. Users on all workstations with a logon of "TWS_user" will be given "modify" access to all jobs
2. Users on all workstations with a logon of "TWSDomain\TWS_user" will be given "display" access to all jobs

However, all users with a logon of "TWS_user" will satisfy the first rule, regardless of their domain, and will be given "modify" access to all jobs. This is because defining a user without its domain is a shorthand way of defining that user ID in *any* domain; it is the equivalent of "@\TWS_User". So the second rule will never be satisfied, for any user, because the matching for the "TWS_user" stops after a successful match is made.

Correct

```
#First User Definition in the Security File
USER Twsdomain:Tws_User
CPU=@+LOGON="TWSDomain\\TWS_user"
Begin
job name=@ access=display
End

#Second User Definition in the Security File
USER Tws_User
CPU=@+LOGON=TWS_user
Begin
job name=@ access=modify
End
```

By putting the more specific definition first, both object access definitions are applied correctly.

See Sample security file on page 255 for a practical example.

User attribute types - detailed description

The *user_attribute_types* and their associated *values* can be any of the following:

cpu={[*folder/*]*workstation*|/@/[*folder/*]*workstation*|/@/@}

where:

workstation

Specifies the workstation on which the user is logged in. Wildcard characters are permitted. The following HCL Workload Automation variables can be used:

$master

Means that the user is logged in on the HCL Workload Automation master domain manager.

$manager

Means that the user is logged in on the HCL Workload Automation domain manager.

$thiscpu

Means that the user is logged in on the HCL Workload Automation workstation on which the security check is running.

@

Specifies that the user is accessing HCL Workload Automation with the Dynamic Workload Console, or is logged in on any HCL Workload Automation workstation.

group=*groupname*

Specifies the name of the group of which the user is a member. Available for UNIX™ users. Wildcard characters are permitted.

logon={*user name*|@}

where:

user name

Specifies the user ID with which the user is logged in on a HCL Workload Automation workstation. Wildcard characters are permitted. The **cpu=** attribute must be set to a specific workstation name (no wildcards) or **@**.

The `user name` value can have one of the following formats:

user name

The Windows user. For example if you use the `user1` value in the logon field, in the `Security` file you have the following line:

```
..............
logon=user1
...............
```

domain\user name

The user belongs to a Windows domain. Insert the escape character '\' before the '\' character in the `domain\user name` value. For example if you use the `MYDOMAIN\user1` value in the logon field, in the `Security` file you have the following line:

```
..............
logon=MYDOMAIN\\user1
...............
```

user name@internet_domain

The user belongs to an internet domain. The user name is in User Principal Name (UPN) format. UPN format is the name of a system user in an email address format. The user name is followed by the "at sign" followed by the name of the Internet domain with which the user is associated.

Insert the escape character '\' before the '@' character in the `user name@internet_domain` value. For example if you use the `administrator@bvt.com` value in the logon field, in the `Security` file you have the following line:

```
..............
logon=administrator\@bvt_env.com
...............
```

For more information about the use of the wildcard with the `domain\user name` and `user name@internet_domain` format in the `Security` file, see Sample security file on page 255.

Note:

1. If the WebSphere Application Server Liberty security configuration option **useDomainQualifiedUserNames** is set to *true*, each user ID defined in the security file must have the format `domain\username` to use the product from one of the following:
    - **composer**
    - **Dynamic Workload Console**
    - **logman**
    - **optman**
    - **planman**

   For more information on WebSphere Application Server Liberty security configuration, see Changing the security settings on page 418.

2. If the user is defined on a Windows™ 2003 system, or when upgrading the Windows™ operating system from an older version to one of those mentioned above, make sure you add the **Impersonate a client after authentication** right to the user settings.

@

Specifies any user logged in with any name or being a member of any HCL administrators group.

Specifying object types

Specify one or more object types that the user or users in the associated user definition is authorized to access. If you specify the object type but no attributes, the authorized actions defined for the user with the **access** keyword apply to all the objects of that type defined in the HCL Workload Automation domain. If an object type from the following list is omitted for a user or users, no objects of that type can be accessed.

The object types are the following:

action

Actions defined in scheduling event rules

calendars

User calendars

cpu

Workstations, domains and workstation classes

event

Event conditions in scheduling event rules

eventrule

Scheduling event rule definitions

file

HCL Workload Automation database file

folder

The folder within which scheduling objects such as, jobs, job streams, and workstations, to name a few, are defined.

job

Scheduled jobs and job definitions

parameter

Local parameters.

prompt

Global prompts

report

The reports on the Dynamic Workload Console that have the following *names*:

RUNHIST

Job Run History

RUNSTATS

Job Run Statistics

WWS

Workstation Workload Summary

WWR

Workstation Workload Runtimes

SQL

Custom SQL

ACTPROD

Actual production details (for current and archived plans)

PLAPROD

Planned production details (for trial and forecast plans)

Permission to use these reports is granted by default to the *TWS_user* on fresh installations.

resource

Scheduling resources

runcygrp

Run cycle groups

schedule

Job streams

userobj

User objects

vartable

Variable tables. This includes authorization to the variable definitions in the tables.

wkldappl

Workload applications

Specifying object attributes

Specify one or more attributes that identify a set of objects that the user of the user definition is authorized to access. If you specify the object type but no object sets, the authorized actions defined for the user with the **access** keyword apply to all the objects of that type defined in the HCL Workload Automation domain.

The general syntax

Each object attribute is specified as follows:

object_attribute=*value*

object_attribute

Object attributes differ according to the object. All objects can be selected by *name*, but some, *jobs*, for example, can be selected by the *workstation* on which they run. See Object attribute on page 226 for full details of which attributes are available for each object type.

value

Identifies an individual object, or, by using wildcards, a set of objects. See Specifying object attributes on page 226 for full details of which attributes are available for each object type.

Object attribute

Specifying object attributes on page 226 lists object attributes that are used to identify a specific set of objects from all objects of the same type. For example, access can be restricted to a set of resource objects having the same name or being defined on the same workstation, or both.

Table 36. Object attribute types for each object type

Attribute Object	name	cpu	folder	cpufolder	custom	jcl	jcltype	logon	provider	type	host	port
action									✓	✓	✓	✓
calendar	✓		✓									
cpu (workstation)		✓		✓						✓		
event					✓				✓	✓		
eventrule	✓		✓									
file	✓											
folder	✓											
job	✓	✓	✓	✓		✓	✓	✓				
lob	✓											
parameter	✓	✓	✓	✓								

Table 36. Object attribute types for each object type (continued)

Attribute Object	name	cpu	folder	cpufolder	custom	jcl	jcltype	logon	provider	type	host	port
prompt	✓		✓									
report	✓											
resource	✓	✓	✓	✓								
runcygrp	✓		✓									
schedule (job stream)	✓	✓	✓	✓								
userobj		✓		✓				✓				
vartable	✓		✓									
wkldappl	✓		✓									

Note:

- Granting access to a workstation class or a domain means to give access just to the object itself, and grant no access to the workstations in the object.
- When you specify access rights on a folder, the access rights apply also to all sub-folders.

Including or excluding

Each attribute can be *included* or *excluded* from the selection using the plus (+) and tilde (~) symbols, in the same way as for the user attributes.

Selection expressions

The detailed syntax and use of the selection expressions for objects is the same as that used to select users:

[**~**]*object_attribute*=*value*[**,***value*]...[{**+**|**~**}*object_attribute*=*value*[**,***value*]...

Order of object definition

You must order object definitions from most specific to least specific, in the same way as for user attributes. For example,

Incorrect

```
job name=@ access=display
job name=ar@ access=@
```

In this case, a job with the name beginning with "ar" would satisfy the first definition, and so would be given the display access, not all access.

Correct

```
job name=ar@ access=@
job name=@ access=display
```

Ensure that you order object definitions from most specific to least specific also when you use the `Continue` keyword.

The `Continue` keyword allows a user to inherit authorization from multiple *stanzas*. The user receives accesses as defined in the first matching entry of each *stanza* that matches the user definition. For an example of a security file with the `Continue` keyword, see Users logged into multiple groups [continue keyword] on page 260

Specifying object attribute values

The following describes the values allowed for each object attribute type:

name=*name*[,*name*]...

Specifies one or more names for the object type. Wildcard characters are permitted. Multiple names must be separated by commas.

- The following values apply to the file object type:

  globalopts

  Allows the user to set global options with the `optman` command. Gives the following access types:

  - Display access for `optman ls` and `optman show`
  - Modify access for `optman chg`

  prodsked

  Allows the user to create, extend, or reset the production plan.

  security

  Allows the user to manage the security file.

  Symphony

  Allows the user to run **stageman** and **JnextPlan**.

  trialsked

  Allows the user to create trial and forecast plans or to extend trial plans.

  Note: Users who have restricted access to files should be given at least the following privilege to be able to display other objects (ie. calendars and cpus):

  ```
  file  name=globalopts  access=display
  ```

- For the event object type use one or more of the event type names listed in the *`TWSObjectsMonitor` events* table or the *`FileMonitor` events* table in the *HCL Workload Automation: User's Guide and Reference*.

- For the action object type use one or more of the action type names listed in the table *Action types by action provider* in the *HCL Workload Automation: User's Guide and Reference*.
- For the **vartable** object type, you can use the $DEFAULT value for the **name** attribute to indicate the default variable table. This selects the table defined with the `isdefault` attribute.

cpu=*workstation* + folder=*foldername*

Specifies one or more workstation, domain, or workstation class names. Workstations and workstation classes can optionally be defined in a folder; if defined, the folder can be specified in the folder attribute.Wildcard characters are permitted. Multiple names must be separated by commas. If this attribute is not specified, all defined workstations and domains can be accessed. Workstation variables can be used - see Using variables in object attribute definitions on page 231.

folder=*foldername*

Scheduling objects such as, jobs, job streams, and workstations, to name a few, can be defined in a folder. A folder can contain one or more scheduling objects, while each object can be associated to only one folder. The default folder is the root folder (/).

cpufolder=*foldername*

The folder within which the workstation or workstation class is defined.

custom=value[,*value*]...

Use this attribute to assign access rights to events defined in event plug-ins. The precise syntax of the value will depend on the plug-in. For example:

- Specify different rights for different users based on SAP R/3 event names when defining event rules for SAP R/3 events.
- Define your own security attribute for your custom-made event providers.
- Specify the type of event that is to be monitored. Every event can be referred to an event provider.

jcl="*path*" | "*command*" | "*jsdl*"

Specifies the command or the path name of a job object's executable file. The command or path must be enclosed in double quotation marks (" "). Wildcard characters are permitted. If omitted, all defined job files and commands qualify.

You can also specify a string contained in the task string of a JSDL definition to be used for pattern matching. Ensure that the string begins and ends with the @ wildcard character and that it is entirely enclosed in double quotation marks as follows: `"@my_string>@"`.

jcltype=[scriptname | docommand]

Specifies that the user is allowed to act on the definitions of jobs that run only scripts (if set to scriptname) or commands (if set to docommand). Use this optional attribute to restrict user authorization to actions on the definitions of jobs of one type or the other only. Actions are granted for both scripts and commands when jcltype is missing.

A user who is not granted authorization to work on job definitions that run either a command or a script is returned a security error message when attempting to run an action on them.

logon=*username*[,...]

Specifies the user IDs. Wildcard characters are permitted. Multiple names must be separated by commas. If omitted, all user IDs qualify.

The user ID can be a Windows domain user or an internet domain user and must be defined in one of the following formats:

domain\user name

The user belongs to a Windows domain. Insert the escape character '\' before the '\' character in the `domain\user name` value. For example if you use the `MYDOMAIN\user1` value in the logon field, in the `Security` file you have the following line:

```
..............
logon=MYDOMAIN\\user1
...............
```

user name@internet_domain

The user belongs to an internet domain. The user name is in User Principal Name (UPN) format. UPN format is the name of a system user in an email address format. The user name is followed by the "at sign" followed by the name of the Internet domain with which the user is associated.

Insert the escape character '\' before the '@' character in the `user name@internet_domain` value. For example if you use the `administrator@bvt.com` value in the logon field, in the `Security` file you have the following line:

```
..............
logon=administrator\@bvt_env.com
...............
```

provider=*provider_name*[,...]

For **action** object types, specifies the name of the action provider.

For **event** object types, specifies the name of the event provider.

Wildcard characters are permitted. Multiple names must be separated by commas. If `provider` is not specified, no defined objects can be accessed.

type=*type*[,...]

For **action** object types, is the `actionType`.

For **event** object types, is the `eventType`.

For **cpu** object types, the permitted values are those used in composer or the Dynamic Workload Console when defining workstations, such as `manager`, `broker`, `fta`, `agent`, `s-agent`, `x-agent`, `rem-eng`, `pool`, `d-pool`, `cpuclass`, and `domain`.

Note: The value `master`, used in conman is mapped against the `manager` security attributes.

Wildcard characters are permitted. Multiple names must be separated by commas. If **type** is not specified, all defined objects are accessed for the specified providers (this is always the case after installation or upgrade, as the type attribute is not supplied by default).

host=*host_name*

For **action** object types, specifies the TEC or SNMP host name (used for some types of actions, such as sending TEC events, or sending SNMP). If it does not apply, this field must be empty.

port=*port_number*

For **action** object types, specifies the TEC or SNMP port number (used for some types of actions, such as sending TEC events, or sending SNMP). If it does not apply, this field must be empty.

Using variables in object attribute definitions

The following variables supplied with the product can be used in object attributes:

Workstation identifiers

$master

The HCL Workload Automation master domain manager.

$manager

The HCL Workload Automation domain manager.

$thiscpu

The workstation on which the user is running the HCL Workload Automation command or program.

Variable table identifiers

$default

The name of the current default variable table.

Specifying access

About this task

Specify the type of access the selected users are allowed to have to the specified objects as follows:

access[=*keyword*[,*keyword*]...]

- To specify that no actions are permitted, use **access=**
- To specify that all actions are permitted, use **access=@**
- To specify any other access, consult the access tables, by object type, below.

How the access tables are organized

The access tables for object types are as follows:

Object types - calendar, cpu, eventrule, folder, job, prompt, resource, run cycle group, schedule, userobj, vartable - using in composer on page 233

Most of the **composer** and GUI database maintenance actions are common to most objects, so they are listed in a table of common object access keywords.

Object type - action on page 236

This gives the access rights for action objects, which are not included in the common table.

Object type - calendar on page 237

This gives the access rights for calendars, which are different or additional to those in the common table.

Object type - cpu on page 238

This gives the access rights for workstations (cpus), which are different or additional to those in the common table.

Object type - event on page 240

This gives the access rights for events, which are different or additional to those in the common table.

Object type - file on page 240

This gives the access rights for files, which are different or additional to those in the common table.

Object type - folder on page 241

This gives the access rights for folders, which are different or in addition to those in the common table.

Object type - job on page 244

This gives the access rights for jobs, which are different or additional to those in the common table.

Object type - parameter on page 248

This gives the access rights for local parameters, which are not included in the common table.

Object type - prompt on page 248

This gives the access rights for prompts, which are different or additional to those in the common table.

Object type - report on page 249

This gives the access rights for reports, which are different or additional to those in the common table.

Object type - resource on page 249

This gives the access rights for resources, which are different or additional to those in the common table.

Object type - run cycle group on page 250

This gives the access rights for run cycle groups, which are different or additional to those in the common table.

Object type - schedule on page 251

This gives the access rights for job streams (schedules), which are different or additional to those in the common table.

Object type - userobj on page 252

This gives the access rights for userobj, which are different or additional to those in the common table.

Object type - vartable on page 253

This gives the access rights for variable tables, which are not included in the common table.

Object type - workload application on page 254

This gives the access rights for workload applications, which are not included in the common table.

Object types - calendar, cpu, eventrule, folder, job, prompt, resource, run cycle group, schedule, userobj, vartable - using in composer

The following table gives the access keywords required to use composer to work with objects of the following types:

- calendar
- cpu
- eventrule
- folder
- job
- prompt
- resource
- run cycle group
- schedule
- userobj
- vartable

Note:

- The `parameter` keyword is reserved for parameters created and managed in a local parameter database with the `parms` utility command.

  For more information about `parms`, see the related section in the *User's Guide and Reference*.

- If you plan to upgrade your environment from a previous version of HCL Workload Automation and use event-driven workload automation, you need to manually add the `display` access keyword to all workstations on which you plan to define File Monitor events.

  For more information about event-driven workload automation, see the related section in the *User's Guide and Reference*.

Table 37. Access keywords for composer actions

		Activity	Access keywords required
Composer	add	Add new object definitions in the database from a file of object definitions. Unlock access is needed to use the `;unlock` attribute. For variable tables, to *add* individual variable entries within a table, the table must have *modify* access. To add a CPU object as a member of a workstation class, you must add *use* access to the CPU object.	add, modify, unlock
	add event rule	Add an event rule of type File Monitor.	display
	create	Create a text file of object definitions in the database. Modify access is need to use the `;lock` attribute. For variable tables, create individual variable entries within the table.	display, modify
	delete	Delete object definitions from the database. For variable tables, to *delete* individual variable entries within a table, the table must have *modify* access.	delete
	display	Display object definitions in the database.	display
	extract	Extract a text file of object definitions from the database.	display
	list	List object definitions in the database.	If the **enListSecChk** global option is set to `yes` on the master domain manager then, either list, or list and display are required.
	lock	Lock object definitions in the database.	modify

Table 37. Access keywords for composer actions (continued)

	Activity		Access keywords required
	modify	Modify object definitions in the database. Definitions are extracted into a file. After you have edited the file the definitions are used to replace the existing ones. For variable tables, to *modify* individual variable entries within a table, the table must have *modify* access.	add, modify
	new	Create object definitions in the database from a template.	add, modify
	print	Print object definitions in the database.	display
	rename	Rename object definitions in the database. You need add access to the new object and delete and display access to the old object.	add, delete, display
	replace	Replace object definitions in the database. Unlock access is needed to use the `;unlock` attribute.	add, modify, unlock
	unlock	Unlock object definitions in the database. For variable tables, unlocking a table unlocks all the variables contained therein. Unlocking a variable unlocks the entire table where it is defined.	unlock
Dynamic Workload Console	Add event rule	Add an event rule of type File Monitor.	display
	Create object in database	Add new object definitions in the database.	add
	Delete object in database	Delete object definitions from the database. Unlock access is needed to use the `;unlock` option.	delete
	Display object in database	Display object definitions in the database.	display
	List object in database	List object definitions in the database.	display
	Modify object in database	Modify object definitions in the database. Unlock access is needed to use the `;unlock` option.	modify
	Unlock object in database	Unlock object definitions in the database locked by another user.	unlock
	Perform operations for job types with advanced options, both those supplied	Perform operations for job types with advanced options in the database.	run

Table 37. Access keywords for composer actions (continued)

	Activity		Access keywords required
	with the product and the additional types implemented through the custom plug-ins. You can define and perform operations on job types with advanced options with the Workload Designer.		
Using the workload service assurance feature	All activities	For any user to perform any workload service assurance activities, the *TWS_user* must have the following access keywords for all *cpu*, *job*, and *schedule* objects:	display, modify, list

Example

To allow a user to use the composer list, display, and modify actions on event rules, specify:

```
eventrule          access=add,display,modify
```

Object type - action

The following table gives the access keywords required for actions:

Table 38. Actions - access keywords

	Activity	Access keywords required
Dynamic Workload Console	Display action instances	display
	List action instances.	list
Dynamic Workload Console	Use these specific action types in event rule definitions.	use
conman		

Table 38. Actions - access keywords (continued)

	Activity	Access keywords required
	• For actions with provider `TWSAction` and types `sbj`, `sbd`, or `sbs`, you must set this keyword in combination with the `submit` access keyword for the specific jobs and job streams specified in the action. • For actions with provider `TWSAction` and type `reply`, you must set this keyword in combination with the `reply` access keyword set for the specific prompts specified in the action. The *TWS_user* of the workstation running the event processing server must have these `submit` and `reply` authorizations, otherwise the event processing server will not be able to run this type of actions.	

Example

To allow a user to use the Dynamic Workload Console to list action instances, specify:

```
action          access=list
```

Object type - calendar

The following table gives the additional access keywords required to work with calendars, other than those described in Table 37: Access keywords for composer actions on page 234:

Table 39. Calendar - additional access keywords

	Activity	Access keywords required
Composer Dynamic Workload Console	Use calendars in: • job streams • run cycles • run cycle groups	use

Example 1

To allow a user to only use calendars when working with job streams in any of the interfaces, specify:

```
calendar          access=use
```

Example 2

To allow a user to display, list, and print calendars, and use them when working with job streams in any of the interfaces, specify:

```
calendar          access=display,use,list
```

Object type - cpu

The following table gives the additional access keywords required to work with cpus (includes workstations, domains, and workstation classes), other than those described in Table 37: Access keywords for composer actions on page 234:

Table 40. Cpus - additional access keywords

		Activity	Access keywords required
Conman Dynamic Workload Console	console	View and send messages to the HCL Workload Automation **conman** console.	console
	deployconf	Force update the monitoring configuration file for the event monitoring engine.	start
	fence	Alter workstation job fences in the production plan.	fence
	limit cpu	Alter workstation job limits in the production plan.	limit
	link	Open workstation links.	link
	resetfta	Generates an updated Sinfonia file and sends it to a fault-tolerant agent on which the Symphony file has corrupted.	resetfta
	showcpus	Display workstations, domains and links in the plan.	list
	shutdown	Shut down HCL Workload Automation processing.	shutdown
	start	Start HCL Workload Automation processing.	start
	startappserver	Start the application server.	start
	starteventprocessor	Start the event processor server.	start
	startmon	Start the event monitoring engine.	start
	stop	Stop HCL Workload Automation processing.	stop
	stop;progressive	Stop HCL Workload Automation processing progressively.	stop
	stopappserver	Stop the application server.	stop
	stopeventprocessor	Stop the event processor server.	stop

Table 40. Cpus - additional access keywords (continued)

Activity			Access keywords required
	stopmon	Stop the event monitoring engine.	stop
	switcheventprocessor	Switch the event processor server from the master domain manager to the backup master domain manager or vice versa.	start, stop
	switchmgr	Switch the domain manager functionality to a workstation.	start, stop
	unlink	Close workstation links.	unlink
	upgrade	Install a fix pack or upgrade to a later version fault-tolerant agents and dynamic agents.	manage
Startup	Start HCL Workload Automation processing.		start
Using the workload service assurance feature	All activities	For any user to perform any workload service assurance activities, the *TWS_user* must have the following access keywords:	display, modify, list
Submit a job	When submitting a job defined in a folder, `use` access is required on the workstation (cpu) where the job is defined, in addition to access to the folder and the objects it contains.		use
Submit a job stream	When submitting a job stream defined in a folder, `use` access is required on the workstation (cpu) where the job is defined, in addition to access to the folder and the objects it contains.		
Composer Dynamic Workload Console	Use a File Monitor event on the workstation where the file resides.		display

Note: If you plan to upgrade your environment from a previous version of HCL Workload Automation and use event-driven workload automation, you need to manually add the `display` access keyword to all workstations on which you plan to define File Monitor events.

For more information about event-driven workload automation, see the related section in the *User's Guide and Reference*.

Example

To allow a user to display, list, and print workstation, workstation class, and domain definitions, link and unlink workstations, and access all workstations defined in the root (/) folder, specify:

```
cpu   name = @ + folder = /    access=display,link,unlink
```

Object type - event

The following table gives the access keywords required to work with events:

Table 41. Events - access keywords

	Activity	Access keywords required
Composer Dynamic Workload Console	Use an event in an event rule definition.	use

Note: If you plan to upgrade your environment from a previous version of HCL Workload Automation and use event-driven workload automation, you need to manually add the `display` access keyword to all workstations on which you plan to define File Monitor events.

For more information about event-driven workload automation, see the related section in the *User's Guide and Reference*.

Example

To allow a user to use an event in an event rule definition, specify:

```
event          access=use
```

Object type - file

The following table gives the access keywords required to work with files (valid only for the command line).

You must specify the file names on page 228 to which the type of access applies.

Table 42. Files - access keywords

	Activity		Access keywords required
dumpsec	Create a text file of the settings contained in the compiled security file.		display
JnextPlan	Generate the production plan.		build
makesec	Compile the security file from a text file of the settings.		modify
optman	ls	List all global options.	display

Table 42. Files - access keywords (continued)

Activity			Access keywords required
	show	Show the details of a global option.	display
	change	Change the details of a global option.	modify
planman	deploy	Manually deploy event rules.	build
prodsked	Work with the production plan.		build
stageman	Carry forward incomplete job streams, archive the old production plan, and install the new production plan.		build
agent_certificate	Download certificates and JWT when installing dynamic agents		display
CREATEPERSONALAPIKEY	Create personal API Key. This allows the CLI or API user access CLIs and APIs.		display
CREATESERVICEAPIKEY	Create service API keys. Only administrators should have this permission.		display
LISTALLAPIKEYS	Allows listing all API Keys in the database. This allows administrators list and then revoke existing API Keys. Without this permission, single users will be able to list ONLY API Keys with same UPN (User Personal Name) as their username (personally-owned API Keys) and service API Keys they have created (personally-owned Service API Keys).		display
DELETEALLAPIKEYS	Allows deleting all API Keys in the database. This allows administrators revoke existing API Keys.		display

Example 1

To allow a user to manage the globalopts file, specify:

```
file   name=globalopts     access=display,modify
```

Example 2

To allow a user to run **JnextPlan**, specify:

```
file          access=build
```

Note: The user will also be able to run **planman deploy**, **prodsked**, and **stageman**.

Object type - folder

The following table gives the additional access keywords required to work with folders, in addition to those common to most objects described in Table 37: Access keywords for composer actions on page 234:

Table 43. folders - access keywords

	Activity		Access keywords required
Composer	chfolder	Change the current folder or working directory.	display
	listfolder	Lists folders defined in the database.	list, or list and display
	mkfolder	Creates a new folder definition in the database.	add
	rmfolder	Deletes folders defined in the database.	delete
	renamefolder	Renames a folder definition in the database.	delete access to the folder with the old name, and add access to the folder with the new name
Conman	Chfolder	Changes the working directory or current directory.	display
	Listfolder	Lists folders defined in the plan.	list, or list and display

See Example on page 242 for detailed examples about how to restrict access to folders.

For more information about designing smart workflow folders, see the related section in *Dynamic Workload Console User's Guide*.

Example

The following examples demonstrate how to restrict access to specific folders. Even with access to a folder, a user still needs additional rights to work with the objects defined in it. When submitting a job or job stream defined in a folder, `use` access is required on the workstation (cpu) where the job is defined, in addition to access to the folder and the objects it contains.

HCL Workload Automation administrator can grant administrator permissions to a user on a folder, `ACL`, so that the user can freely assign access control lists to other users on the same folder or any sub-folders. Users can then access the objects in the folder or sub-folders. For more information about delegating administrator access to users and groups on a folder, see the related topic in the *Administration Guide*.

Examples

Tim the HCL Workload Automation administrator, delegates Linda, the `app1_admin` user, permissions on the folder `/PRD/APP1` and any sub-folders, by assigning her the `ACL` access on the folder. With this access, Linda can create access control lists to grant access to the folder or sub-folders to other users with a predefined role. The following is the security file for Linda, the `app1_admin` user:

```
###########################################################
#     Sample Security File
###########################################################
USER APPADMINofPRDAPP1  cpu=JUPITER+LOGON=app1_admin
begin
#  OBJECT     ATTRIBUTES                       ACCESS CAPABILITIES
# ----------  ------------                     -------------------
job           cpu=JUPITER  + folder = "/PRD/APP1","/PRD/APP1/"
                  access=add,delete,display,modify,use,list,unlock
schedule      cpu=JUPITER  + folder = "/PRD/APP1","/PRD/APP1/"
                  access=add,delete,display,modify,use,list,unlock
folder        name="/PRD/APP1","/PRD/APP1/"
                  access=add,delete,display,modify,use,list,unlock,acl
```

User `jsmith` is granted unrestricted access to jobs and job streams defined in the folder named APPS and on the workstation named `JUPITER`, specify:

```
###########################################################
#     Sample Security File
###########################################################
user  jsmith  cpu=JUPITER
begin
#  OBJECT     ATTRIBUTES                    ACCESS CAPABILITIES
# ----------  ------------                  -------------------
job           cpu=JUPITER  + folder = /APPS/  access=@
schedule      cpu=JUPITER  + folder = /APPS/  access=@
cpu           cpu=JUPITER+LOGON=jsmith        access=use
folder        name=/APPS/                     access=add,delete,display,
                                              modify,use,list,unlock,acl
```

To allow a user to have the specified rights on any folder, the root folder and any sub-folders, specify:

```
folder   name=/      access=add,delete,display,modify,use,list,unlock
```

To grant a user access only to the root folder (/), you can omit specifying the folder object in the security file. This is the same behavior as in security files for releases prior to Version 9.5. After upgrading to Version 9.5, all of the objects are moved to the root folder, so if you continue to use your old security file which does not include the v95fp1 attribute or object (for example, for jobs, `JOB CPU=@ ACCESS=ADD,ADDDEP,…,RERUN,SUBMIT,USE,LIST,UNLOCK`, then users have access to only the root (/) folder by default.

To allow a user to have the specified rights only on the "APPS" folder, specify:

```
folder   name=/APPS/      access=add,delete,display,modify,use,list,unlock
```

To allow a user to have the specified rights on the folder "APPS" and its sub-folders, specify:

```
folder   name=/APPS        access=add,delete,display,modify,use,list,unlock
```

To allow a user to have the specified rights only on folder "APP1" and its sub-folders, specify:

```
folder   name=/APPS/APP1       access=add,delete,display,modify,use,list,unlock
```

To allow a user to have all rights on the folder "APPS" and on the folder "APP2" and its sub-folder, but no rights on APP1, specify:

```
folder   name=/APPS/              access=@
folder   name=/APPS/APP1/APP2     access=@
```

Object type - job

The following table gives the additional access keywords required to work with jobs, other than those described in Table 37: Access keywords for composer actions on page 234:

Table 44. Jobs - additional access keywords

	Activity		Access keywords required
Composer	Use jobs in job streams.		use
Dynamic Workload Console	Also, if a job is used as a recovery job in a job definition, the user must have "use" access to the definition of the job identified as the recovery job.		
Conman Dynamic Workload Console	adddep	Add dependencies to jobs in the production plan. Not valid for workstations in end-to-end environment.	adddep
	altpri	Alter the priority of jobs in the production plan. Not valid for workstations in end-to-end environment.	altpri
	cancel job	Cancel jobs in the production plan. Not valid for workstations in end-to-end environment.	cancel
	confirm	Confirm completion of jobs in the production plan. Not valid for workstations in end-to-end environment.	confirm
	deldep job	Delete dependencies from jobs in the production plan. Not valid for workstations in end-to-end environment.	deldep
	display	Display jobs in the plan.	display
	Hold	Hold a job to prevent it from running	adddep
	kill	Kill running jobs.	kill
	release job	Release jobs from dependencies in the production plan. Not valid for workstations in end-to-end environment.	release
	reply	Reply to job prompts in the production plan.	reply
	rerun	Rerun jobs in the production plan. Not valid for workstations in end-to-end environment.	rerun; submitdb

Table 44. Jobs - additional access keywords (continued)

		Activity	Access keywords required
		To use the **from** argument, you must have ***submitdb*** access to the job.	
	showjobs	Display information about jobs in the production plan.	list
Conman Dynamic Workload Console	submit docommand	Submit commands as jobs or recovery jobs into the production plan. If the submit also identifies a second job with the "ALIAS" or "RECOVERYJOB" arguments, the user must have "submit" access to that other job, as well Not valid for workstations in end-to-end environment.	submit
	submit file	Submit files as jobs or recovery jobs into the production plan. If the submit also identifies a second job with the "ALIAS" or "RECOVERYJOB" arguments, the user must have "submit" access to that other job as well. Not valid for workstations in an end-to-end environment.	submit
	submit job	Submit jobs or recovery jobs into the production plan. If the submit also identifies a second job with the "ALIAS" or "RECOVERYJOB" arguments, the user must have "submit" access to that other job as well. If the job is defined in a folder, then `use` access is required on the workstation (cpu) where the job is defined, in addition to access to the folder itself and the objects it contains. Not valid for workstations in an end-to-end environment.	submit
		Restricts the submission action to jobs defined in the database. With this authorization level a user cannot submit ad hoc jobs. Use this keyword to allow a user to submit only jobs defined in the database. Use the submit keyword to allow a user to submit both defined and ad hoc jobs. Users granted only submitdb rights:	submitdb

Table 44. Jobs - additional access keywords (continued)

Activity			Access keywords required
		- Cannot run submit docommand and submit file successfully - Are displayed tasks related to ad hoc job submission on the graphical user interfaces, but if they run them, are returned error messages for lacking the submit access right.	
	submit sched	Submit job streams into the production plan. Not valid for workstations in end-to-end environment.	submit
	Hold	Hold a job to prevent it from running	adddep
Dynamic Workload Console	For critical jobs on which you run any of the following actions: - Display hot list - Display critical path - Display incompleted predecessors - Display completed predecessors	The predecessors are listed regardless of the fact that this authorization might not be extended to them. However, if you want to run any further action on any of the listed predecessors, this will require that you have the proper authorization.	list
Using the workload service assurance feature	All activities	For any user to perform any workload service assurance activities, the *TWS_user* must have the following access keywords:	display, modify, list

Example 1

To allow a user to manage only job dependencies for jobs defined in the root (/) folder, specify:

```
job       access=adddep,deldep
```

Example 2

To allow a user to only manage critical jobs defined in the root (/) folder, specify:

```
job          access=list,altpri
```

Example 3

User `administrator` is granted add, modify, and display rights for all job definitions defined in the folder named "APPS" and any sub-folders, on workstations defined in the root (/) folder, and is therefore permitted to create and modify job definitions that run scripts or commands as needed, with no restriction:

```
USER TWSADMIN
CPU=@+LOGON=administrator
BEGIN
JOB CPU=@  + FOLDER = /APPS + CPUFOLDER = / ACCESS=ADD,MODIFY,DISPLAY,…
[…]
END
```

User `sconnor` is granted the same rights for jobs that match the condition jcltype=scriptname, which means that he can create or modify only job definitions that run scripts and cannot change any of them into a job that runs a command. He can also access all workstation defined in the root (/) folder:

```
USER RESTRICTED
CPU=@+LOGON=sconnor
BEGIN
JOB CPU=@+JCLTYPE=SCRIPTNAME  + FOLDER = /APPS + CPUFOLDER = / ACCESS=ADD,MODIFY,DISPLAY,…
[…]
END
```

Example 4

User `administrator` is granted submit permission for all jobs defined in all folders ("/"), and is therefore permitted to submit jobs defined in the database and ad hoc, with no restriction:

```
USER TWSADMIN
CPU=@+LOGON=administrator
BEGIN
JOB CPU=@  + FOLDER = / + CPUFOLDER = / ACCESS=ADD,ADDDEP,…,RERUN,SUBMIT,USE,LIST,UNLOCK
[…]
END
```

User `jsmith` is granted submitdb permission for all jobs defined in all folders, allowing her to submit all jobs defined in the database, but she is not permitted to run ad hoc job submissions. She also has access to workstations in the /MYCPUS folder:

```
USER RESTRICTED
CPU=@+LOGON=jsmith
BEGIN
JOB CPU=@  + FOLDER = / + CPUFOLDER = /MYCPUS ACCESS=ADD,ADDDEP,…,RERUN,SUBMITDB,USE,LIST,UNLOCK
[…]
END
```

Object type - parameter

The following table gives the access keywords required to work with parameters:

Note: Starting from version 8.5, the `parameter` keyword is reserved for parameters created and managed in a local parameter database with the `parms` utility command. See the *HCL Workload Automation: User's Guide and Reference* for details on `parms`.

Table 45. Parameters - additional access keywords

	Activity	Access keywords required
parms	Manage local parameter definitions.	display

Example

To allow a user to perform all activities on parameters and with access to all workstations defined in the root (/) folder, specify:

```
parameter  + folder = /  + cpufolder = /   access=@
```

Object type - prompt

The following table gives the additional access keywords required to work with prompts, other than those described in Table 37: Access keywords for composer actions on page 234:

Table 46. Prompts - additional access keywords

	Activity		Access keywords required
Composer Dynamic Workload Console	Use prompts when defining or submitting jobs and job streams		use
Conman Dynamic Workload Console	adddep	Use prompts when adding dependencies to jobs in the production plan. Not valid for workstations in end-to-end environment.	use
	recall	Display prompts waiting for a response.	display
	reply	Reply to a job or Job Scheduler prompt.	reply
	showprompts	Display information about prompts.	list

Table 46. Prompts - additional access keywords (continued)

		Activity	Access keywords required
	submit docommand	Use prompts when submitting commands as jobs into the production plan. Not valid for workstations in end-to-end environment.	use
	submit file	Use prompts when submitting files as jobs into the production plan. Not valid for workstations in end-to-end environment.	use
	submit job	Use prompts when submitting jobs into the production plan. Not valid for workstations in end-to-end environment.	use
	submit sched	Use prompts when submitting job streams into the production plan. Not valid for workstations in end-to-end environment.	use

Example

To allow a user to perform all activities on prompts except reply to them, specify:

```
prompt          access=use,display,list
```

Object type - report

The following table gives the access keywords required to work with reports.

Table 47. Files- access keywords

	Activity	Access keywords required
Dynamic Workload Console	Display reports on page 225 on Dynamic Workload Console.	display

Example

To allow a user to display reports on the Dynamic Workload Console, specify:

```
report          access=display
```

Object type - resource

The following table gives the additional access keywords required to work with resources, other than those described in Table 37: Access keywords for composer actions on page 234:

Table 48. Resources - additional access keywords

	Activity		Access keywords required
Composer Dynamic Workload Console	Use resources when defining or submitting jobs and job streams		use
Conman Dynamic Workload Console	adddep	Use resources when adding dependencies to jobs in the production plan. Not valid for workstations in end-to-end environment.	use
	resource	Change the number of units of a resource on a workstation.	resource
	showresources	Display information about resources.	list
	submit docommand	Use resources when submitting commands as jobs into the production plan. Not valid for workstations in end-to-end environment.	use
	submit file	Use resources when submitting files as jobs into the production plan. Not valid for workstations in end-to-end environment.	use
	submit job	Use resources when submitting jobs into the production plan. Not valid for workstations in end-to-end environment.	use
	submit sched	Use resources when submitting job streams into the production plan. Not valid for workstations in end-to-end environment.	use

Example

To allow a user to display information about resources defined in the root folder, and change the units of a resource on a workstation defined in the root folder (/), but not to use them in any other scheduling objects or actions, specify:

```
resource   + folder = /   + cpufolder = /    access=list,resource
```

Object type - run cycle group

The following table gives the access keywords required to work with run cycle groups:

Table 49. Run cycle groups- access keywords

	Activity	Access keywords required
Composer	Use run cycle groups in job streams.	use

Table 49. Run cycle groups- access keywords (continued)

Activity			Access keywords required
Dynamic Workload Console			

Example

To allow a user to create and delete a run cycle group, specify:

```
runcygrp          access=add,delete
```

Object type - schedule

The following table gives the additional access keywords required to work with job streams, other than those described in Table 37: Access keywords for composer actions on page 234:

Table 50. Job streams - additional access keywords

Activity			Access keywords required
Conman Dynamic Workload Console	adddep	Add dependencies to job streams in the production plan. Not valid for workstations in end-to-end environment.	adddep
	altpri	Alter the priority of job streams in the production plan. Not valid for workstations in end-to-end environment.	altpri
	cancel sched	Cancel job streams in the production plan. Not valid for workstations in end-to-end environment.	cancel
	deldep sched	Delete dependencies from job streams in the production plan. Not valid for workstations in end-to-end environment.	deldep
	display	Display job streams in the plan. .	display
	limit sched	Modify the limit for jobs concurrently running within a Job Scheduler.	limit
	release sched	Release job streams from dependencies in the production plan. Not valid for workstations in an end-to-end environment.	release
	reply	Reply to job stream prompts in the production plan.	reply
	showschedules	Display information about job streams in the production plan.	list
	submit sched	Submit job streams into the production plan.	submit

Table 50. Job streams - additional access keywords (continued)

Activity			Access keywords required
		If the submit also identifies a second job stream with the "ALIAS" argument, the user must have "submit" access to that other job stream as well. If the job stream is defined in a folder, then `use` access is required on the workstation (cpu) where the job stream is defined, in addition to access to the folder itself and the objects it contains. Not valid for workstations in an end-to-end environment.	
Using the workload service assurance feature	All activities	For any user to perform any workload service assurance activities, the *TWS_user* must have the following access keywords:	display, modify, list

Example

To allow a user to perform all actions on job streams defined in the `"test"` folder and its sub-folders, except submit and release, and access to all workstations defined in the root (/) folder, specify:

```
schedule folder = /test  + CPUFOLDER = / access=adddep,altpri,cancel,deldep,display,
    limit,reply,list
```

Object type - userobj

The following table gives the additional access keywords required to work with users, other than those described in Table 37: Access keywords for composer actions on page 234:

Table 51. Users - additional access keywords

Activity			Access keywords required
Composer Dynamic Workload Console	Modeling of job types with advanced options	When defining job types with advanced options allows the modeler to specify in the credentials section of the job that the `user name` and `password` values required to submit the job are resolved at run time with values extracted from the database and defined with the User definition composer commands (`username` and `password`) or Dynamic Workload Console panel.	use

Table 51. Users - additional access keywords (continued)

	Activity		Access keywords required
		Note that on dynamic agents User definitions can be used regardless of the operating system.	
Conman	altpass	Alter user passwords in the plan.	altpass
Dynamic Workload Console			

Example

The following access definition allows a user to:

- List and modify user information, including passwords in the database (`display`, `modify`, and `altpass`).
- When defining job types with advanced options on dynamic agents, to specify in the credentials section of the job that the `user name` and `password` values required to submit the job are resolved at run time with values extracted from the database and defined with the User definition (`use`).

```
userobj          access=display,modify,altpass,use,list
```

Object type - vartable

The following table gives the access keywords for using variable tables and the variables they contain (this includes the global variables)

Table 52. Variable tables - access keywords

	Activity	Access keywords required
Composer	Use variable tables in run cycles, run cycle groups, job streams, and workstations	use
Dynamic Workload Console		

Example

To allow a user only to use variable tables when defining other scheduling objects, specify:

```
vartable          access=use
```

Object type - workload application

The following table gives the access keywords required to work with workload applications:

Table 53. Workload applications - access keywords

	Activity		Access keywords required
Dynamic Workload Console	add	Add new workload applications templates to the database. Unlock access is needed to use the `;unlock` attribute.	add, unlock
	create	Create a workload application template in the database. Modify access is needed to use the `;lock` attribute.	display, modify
	delete	Delete a workload application template from the database.	delete
	display	Display a workload application template.	display
	list	List workload application templates in the database.	list
	lock	Lock workload application templates in the database.	modify
	modify	Modify a workload application template in the database.	add, modify
	new	Create a workload application template in the database.	add, modify
	rename	Rename workload application templates in the database. The user needs add access to the new object and delete and display access to the old object.	add, delete, display
	replace	Replace workload application templates in the database. Unlock access is needed to use the `;unlock` attribute.	add, modify, unlock
	unlock	Unlock workload application templates in the database.	unlock

Example

To allow a user to create and delete a workload application, specify:

```
wkldappl          access=add,delete
```

The *TWS_user* - special security file considerations

The TWS_user is a special user, and requires special consideration for the security file.

Required access for the *TWS_user* for workload service assurance

For any user to perform Workload service Assurance activities, the *TWS_user* must have *display*, *modify* and *list* access keywords assigned for all *job*, *schedule* and *cpu* objects.

New *TWS_user* in migrated Security file

If you change the *TWS_user* of your environment, for example, as you might do when performing a parallel upgrade, and then you migrate the Security file (to preserve your settings) you must set up the new *TWS_user* in the Security file in advance, with all its required access rights, before attempting to start HCL Workload Automation.

Update definitions for Windows domain *TWS_user* in the Security file after upgrade to version 10.2.5

Due to new support of the UPN Windows user, if you have Windows domain users that are defined in the logon fields as `domain\username`, after performing an upgrade to version 10.2.5, update the `Security` file before starting the HCL Workload Automation instance. Insert the escape character '\' before the '\' character in the `domain\username` value.

For example, if you use the `MYDOMAIN\user1` value in the logon field, after the upgrade, in the `Security` file you must update the line in following way:

```
..............
logon=MYDOMAIN\\user1
...............
```

Sample security file

This section contains a sample security file divided into sections for each different class of user.

Note that the order of definitions is from most to least-specific. Because of the order, *TWS_users* and **root** users are matched first, followed by users in the **sys** group, and then users in the **mis** group. All other users are matched with the last definition, which is the least specific.

TWS_users and root users logged in on the master domain manager

user mastersm cpu=$master + logon=*TWS_user*,root

```
###########################################################
#     Sample Security File
###########################################################
# APPLIES TO TWS_users AND ROOT USERS LOGGED IN ON THE
# MASTER DOMAIN MANAGER.
user mastersm  cpu=$master + logon=TWS_user,root
begin
#  OBJECT     ATTRIBUTES        ACCESS CAPABILITIES
# ----------  ------------      ----------------------
job           cpu=@ + folder = / + cpufolder = /  access=@
schedule      cpu=@ + folder = / + cpufolder = /  access=@
resource      + folder = / + cpufolder = /      access=@
prompt        + folder = /                      access=@
file               access=@
calendar      + folder = /    access=@
cpu           cpu=@  + folder = /                 access=@
parameter     name=@ ~ name=r@  + folder = /  + cpufolder = /  access=@
userobj       cpu=@ + logon=@  + cpufolder = /    access=@
eventrule     name=@    + folder = /    access=add,delete,display,modify,list,unlock
action        provider=@        access=display,submit,use,list
```

```
event         provider=@        access=use
report        name=@            access=display
runcygrp      name=@   + folder = /      access=add,delete,display,modify,use,list,unlock
vartable      name=a@,$default  + folder = /  access=add,delete,display,modify,use,list,unlock
wkldappl      name=@    + folder = /     access=add,delete,display,modify,list,unlock
lob           name=@            access=use
folder        name=/            access=@
end
```

This user definition applies to GUI and CLI access for *TWS_users* and **root** users logged into a master domain manager. They are given unrestricted access to all objects, except parameters that have names beginning with **r**. Access to the **r** parameters is given only to users in the **mis** group. They are the only ones who can generate all kinds of plans and who can create, update, and delete event rule definitions.

All users have access to all variable tables beginning with "a" and to the default table, irrespective of the default variable table name.

TWS_users and root users logged in on any domain manager (other than the master)

user testerlondon cpu=$manager + logon=*TWS_user*,root

```
###########################################################
#     Sample Security File
###########################################################
# APPLIES TO TWS_users AND ROOT USERS LOGGED IN ON ANY
# DOMAIN MANAGER.
user testerlondon  cpu=$manager + logon=TWS_user,root
begin
#  OBJECT     ATTRIBUTES      ACCESS CAPABILITIES
# ----------  ------------    ----------------------
job           cpu=@  + folder = /  + cpufolder = / access=add,delete,display
schedule      cpu=@  + folder = /  + cpufolder = / access=add,delete,display
resource      + folder = / + cpufolder = /        access=@
prompt        + folder = /              access=@
file          name=prodsked   access=build, display
file          name=trialsked  access=build, display
calendar      + folder = /              access=@
cpu           cpu=@   + folder = /     access=@
parameter  name=@ ~ name=v@  + folder = / + cpufolder = / access=@
userobj    cpu=@ + logon=@  + cpufolder = /  access=@
eventrule       name=@   + folder = /     access=add,delete,display,modify,list,unlock
action     provider=@         access=display,submit,use,list
event      provider=@         access=use
report     name=@             access=display
runcygrp      name=@   + folder = /       access=add,delete,display,modify,use,list,unlock
vartable   name=a@,$default + folder = /  access=add,delete,display,modify,use,list,unlock
wkldappl      name=@     + folder = /     access=add,delete,display,modify,list,unlock
lob           name=@          access=use
folder        name=/          access=@
end
```

This user definition applies to GUI and CLI access for *TWS_users* and **root** users logged into any domain manager other than the master. They are given unrestricted access to all objects, except parameters that have names beginning with **v**, and jobs and jobs streams to which they have limited access. They can access all workstations defined in the root folder (/). They can generate all types of plans and can create, update, and delete event rule definitions defined in the root folder.

All users have access to all variable tables beginning with "a" and to the default table, irrespective of the default variable table name.

TWS_users and root users logged in on any workstation other than any domain manager

user sm ~CPU=$MANAGER logon=*TWS_user*,root

```
###########################################################
# APPLIES TO TWS_users AND ROOT USERS LOGGED IN ON ANY
# WORKSTATION OTHER THAN THE MASTER DOMAIN MANAGER.
user sm  logon=TWS_user,root
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=$thiscpu + folder = /   + cpufolder = / access=@
schedule      cpu=$thiscpu + folder = /   + cpufolder = / access=@
resource      cpu=$thiscpu  + folder = / + cpufolder = /  access=@
prompt        + folder = /       access=@
calendar      + folder = /                 access=@
cpu           cpu=$thiscpu + folder = /    access=@
parameter     cpu=$thiscpu ~ name=r@ + folder = /  + cpufolder = /   access=@
action        provider=@       access=display,submit,use,list
event         provider=@       access=use
report        name=RUNHIST,RUNSTATS  access=display
runcygrp      name=@    + folder = /   access=add,delete,display,modify,use,list,unlock
file          name=globalopts   access=display
lob           name=@            access=use
folder        name=/myfolder    access=@
end
```

This user definition applies to *TWS_users* and **root** users to whom definition (1) does not apply, which are those who are logged in on any workstation other than the master domain manager or any other domain manager. They are given unrestricted access to all objects on their login workstation. Note that prompts, files, and calendars are global in nature and are not associated with a workstation.

They can use event rules, but are not allowed to create, update, or delete event rule definitions.

Users logged into the *sys* group on the master domain manager

user masterop cpu=$master + group=sys

```
###########################################################
# APPLIES TO USERS LOGGED INTO THE SYS GROUP ON THE
# MASTER DOMAIN MANAGER.
user masterop  cpu=$master + group=sys
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=@ + logon="TWS_domain\TWS_user"
                   + folder = /  access=@
job           cpu=@ + logon=root  + folder = /
```

```
                                  + cpufolder = /
                                 access=adddep,altpri,cancel,confirm,
                                 deldep,release,reply,rerun,submit,use
job           cpu=@ + logon=@ ~ logon=root  + folder = /
                                  + cpufolder = /
                                 access=add,adddep,altpri,cancel,confirm,deldep,
                                 release,reply,rerun,submit,use
schedule      cpu=$thiscpu   + folder = /  + cpufolder = /  access=@
schedule      cpu=@   + folder = /  + cpufolder = /
                                     access=adddep,altpri,cancel,
                                     deldep,limit,release,submit
resource      + folder = /      access=add,display,resource,use
file          name=globalopts   access=display
file          name=prodsked     access=display
file          name=symphony     access=display
file          name=trialsked    access=build, display
calendar      + folder = /                  access=display,use
cpu           cpu=@     + folder = /         access=@
parameter     name=@ ~ name=r@  + folder = /  access=@
report        name=RUNHIST,RUNSTATS  access=display
wkldappl      name=@   + folder = /         access=add,delete,display,modify,list,unlock
lob           name=@            access=use
folder        name=/            access=@
end
```

This user definition applies to users logged into the **sys** group on the master domain manager. They are given a unique set of access capabilities. Multiple object statements are used to give these users specific types of access to different sets of objects. For example, there are three job statements:

- The first job statement permits unrestricted access to jobs that run on any workstation (@) under the user's name (*TWS_domain**TWS_user*).
- The second job statement permits specific types of access to jobs that run on any workstation and that run as **root**.
- The third job statement permits specific types of access to jobs that run on any workstation. Jobs that run as root are excluded.

They are the only users defined on the master domain manager, different from maestro or root, who can generate trial and forecast plans.

Users logged into the *sys* group on any workstation other than the master domain manager

user op ~cpu=$master group=sys

```
###########################################################
# APPLIES TO USERS LOGGED INTO THE SYS GROUP ON ANY
# WORKSTATION OTHER THAN THE MASTER DOMAIN MANAGER
user op  group=sys
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=$thiscpu + logon=@  + folder = /
                                + cpufolder = / access=@
job           cpu=$thiscpu + logon=root  + folder = /
```

```
                                + cpufolder = /
                               access=adddep,altpri,cancel,confirm,deldep,
                               release,reply,rerun,submit,use
job           cpu=$thiscpu ~ logon=root  + folder = /
                                + cpufolder = /
                               access=adddep,altpri,cancel,confirm,deldep,
                               release,reply,rerun,submit,use
schedule      cpu=$thiscpu  + folder = /  + cpufolder = /  access=@
resource      + folder = /           access=add,display,resource,use
runcygrp      name=@    + folder = /   access=add,delete,display,modify,use,list,unlock
prompt        + folder = /             access=add,display,reply,use
calendar      + folder = /             access=use
cpu           cpu=$thiscpu  + folder = /  access=console,fence,limit,
                               link,start,stop,unlink
parameter     name=@ ~ name=r@ + folder = /  access=@
wkldappl      name=@     + folder = /   access=add,delete,display,modify,list,unlock
lob           name=@           access=use
folder        name=/           access=@
end

###########################################################
```

This user definition applies to **sys** group users to whom definition (3) does not apply, which are those who are logged in on any workstation other than the master domain manager. They are given a set of access capabilities similar to those in definition (3). The exception is that access is restricted to objects on the user's login workstation (**$thiscpu**).

Users logged into the *mis* group on any workstation

user misusers group=mis

```
###########################################################
# APPLIES TO USERS LOGGED INTO THE MIS GROUP ON
# ANY WORKSTATION.
user misusers  cpu=@           group=mis
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=$thiscpu  + folder = /
              + logon=@  + cpufolder = /      access=@
job           cpu=$thiscpu  + folder = /
              + logon=@
              ~ logon=root + cpufolder = /    access=submit,use
schedule      cpu=$thiscpu   + folder = /
               + cpufolder = /  access=add,submit,modify,display
cpu           cpu=@ + type=agent,s-agent,fta + folder = /
               access=console,fence,limit,link,start,stop,unlink
parameter     name=r@   + folder = / + cpufolder = /    access=@
parameter     name=@    + folder = / + cpufolder = /    access=display
runcygrp    name=@   + folder = /  access=add,delete,display,modify,use,list,unlock
folder        name=/           access=@
end
###########################################################
```

This user definition applies to users logged into the **mis** group on workstations defined in the root folder. They are given a limited set of access capabilities to fault-tolerant, standard, and dynamic agents. Resources, prompts, files, calendars, and workstations are omitted, which prevents access to these objects. These users are given unrestricted access to parameters with names that begin with **r**, and that are defined in the root folder, but can only display other parameters.

Users logged into multiple groups [continue keyword]

This is an example of a security file where the `continue` keyword is used. This kind of security file allows a user to inherit authorization from multiple *stanzas*. The user gets the accesses for the first matching entry of each *stanza* that matches the user definition.

user misusers cpu@ group=mis

```
###########################################################
# User misusers USER DEFINITION APPLIES TO USERS LOGGED IN TO
# THE MIS GROUP ON ANY WORKSTATION.
#
# User dbusers USER DEFINITION APPLIES TO USERS LOGGED IN TO
# THE DB GROUP ON ANY WORKSTATION.
#
# User default USER DEFINITION APPLIES TO ALL USERS.
#

user misusers  cpu=@           group=mis
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=@  + name=mis@   + folder = /
                    + cpufolder = / access=@
schedule      name=mis@  + folder = /  + cpufolder = /  access=@
parameter     name=mis@ + folder = /  + cpufolder = /   access=@
continue
folder        name=/            access=@

user dbusers  cpu=@            group=db
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=@ +  name=db_@  + folder = /
                        + cpufolder = / access=@
schedule      name=db_@   + folder = /
                        + cpufolder = /  access=@
parameter     name=db_@ + folder = /  + cpufolder = /   access=@
continue
folder        name=/           access=@

user default cpu=@ + logon=@
begin
#  OBJECT     ATTRIBUTES                ACCESS CAPABILITIES
# ----------  ------------              ----------------------
parameter     name=@ + folder = /  + cpufolder = /   access=display
folder        name=/                    access=@
end


###########################################################
```

Users that belong only to the *mis* group get access to all objects that have a name starting with the *mis* prefix, as specified in the `user misusers` user definition. In addition, the `user default` user definition gives them display access to all parameters.

Users that belong only to the *db* group get access to all objects that have a name starting with the *db_* prefix, as specified in the `user dbusers` user definition. In addition, the `user default` user definition gives them display access to all parameters.

Users that belong to both the *mis* and the *db* groups get access to the objects that have a name starting with the *mis* prefix and to the objects that have a name starting with the *db_* prefix, as specified in the `user misusers` and in the `user dbusers` user definitions. In addition, the `user default` user definition gives them display access to all parameters. Access to jobs, job streams, workstations, and parameters is limited to only those defined in the root (/) folder.

You must order definitions from most specific to least specific. The `user default` user definition gives generic accesses, and must be therefore specified at the end of the file.

All other users logged in on any workstation

user default cpu=@ + logon=@

```
###########################################################
# APPLIES TO ALL OTHER USERS LOGGED IN ON ANY
# WORKSTATION.
user default  cpu=@ + logon=@
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=@  + folder = /  + cpufolder = / access=@
schedule      cpu=@  + folder = /  + cpufolder = / access=@
resource      + folder = /  + cpufolder = /                  access=@
prompt        + folder = /                  access=@
file                            access=@
calendar      + folder = /                  access=@
cpu           cpu=@   + folder = /  access=@
parameter     name=@ ~ name=r@  + folder = /  + cpufolder = / access=@
userobj       cpu=@ + logon=@   + cpufolder = / access=@
eventrule     name=@  + folder = / access=add,delete,display,modify,list,unlock
action        provider=@        access=display,submit,use,list
event         provider=@        access=use
report        name=@            access=display
runcygrp      name=@            access=add,delete,display,modify,use,list,unlock
vartable      name=a@,$default  + folder = / access=add,delete,display,modify,use,list,unlock
wkldappl   name=@   + folder = /   access=add,delete,display,modify,list,unlock
lob           name=@            access=use
folder        name=/            access=@
end
###########################################################
```

They are given unrestricted access to all objects, except parameters that have names beginning with **r**. They are the only ones who can generate all kinds of plans and who can create, update, and delete event rule definitions. All users have access to all variable tables beginning with "a" and to the default table, irrespective of the default variable table name. Access to most scheduling objects is limited to those defined in the root (/) folder.

All domain1.com windows users logged in on any workstation

user cpu=@ + logon =@\@domain1.com

```
###########################################################
# APPLIES TO ALL OTHER USERS IN THE 'domain1.com' INTERNET DOMAIN LOGGED IN ON ANY
# WORKSTATION.
user default  cpu=@ + logon=@\@domain1.com
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=@ + logon =a@\@domain1.com  + folder = /
                       + cpufolder = /  access=display
job           cpu=@    + folder = /
                       + cpufolder = /  access=@
schedule      + folder = / + cpufolder = /     access=@
resource    + folder = /   + cpufolder = /   access=@
prompt      + folder = /                     access=@
file                            access=@
calendar    + folder = /                     access=@
cpu          cpu=@      + folder = /           access=@
parameter    name=@ ~ name=r@  + folder = /  + cpufolder = /  access=@
userobj      cpu=@ + logon=@  + cpufolder = /  access=@
eventrule    name=@      + folder = /        access=add,delete,display,modify,list,unlock
action       provider=@        access=display,submit,use,list
event        provider=@        access=use
report       name=@            access=display
runcygrp     name=@    + folder = /         access=add,delete,display,modify,use,list,unlock
vartable     name=g@,$default  + folder = / access=add,delete,display,modify,use,list,unlock
wkldappl     name=@       + folder = /      access=add,delete,display,modify,list,unlock
lob          name=@            access=use
folder       name=/            access=@
end
###########################################################
```

Windows Users in `domain1.com` whose name begins with 'a' can display only jobs and can manage parameters which name does not begin with **r**. All other `domain1.com` Windows users that are logged in on any workstation are given access to all objects defined in the root (/) folder, and to parameters that have names beginning with **r**. They are the only ones who can generate all kinds of plans and who can create, update, and delete event rule definitions. All users have access to all variable tables beginning with "g" and to the default table, irrespective of the default variable table name.

All MYWINDOM windows users logged in on any workstation

user default cpu=@ + logon=MYWINDOM\\@

```
###########################################################
# APPLIES TO ALL "MYWINDOM" WINDOWS USERS LOGGED IN ON ANY
# WORKSTATION.
user default  cpu=@ +  logon=MYWINDOM\\@
begin
#  OBJECT     ATTRIBUTES       ACCESS CAPABILITIES
# ----------  ------------     ----------------------
job           cpu=@    + folder = /
                       + cpufolder = /   access=@
schedule      cpu=@    + folder = /
                       + cpufolder = /   access=@
```

```
resource       + folder = /  + cpufolder = /   access=@
prompt         + folder = /                    access=@
file                                     access=@
calendar       + folder = /              access=@
cpu           cpu=@     + folder = /     access=@
parameter     name=@    + folder = /  + cpufolder = / access=@
userobj       cpu=@ + logon =MYWINDOW\\r@   + cpufolder = / access=display
userobj       cpu=@ + logon=@   + cpufolder = / access=@
eventrule     name=@     + folder = /    access=add,delete,display,modify,list,unlock
action        provider=@        access=display,submit,use,list
event         provider=@        access=use
report        name=@            access=display
runcygrp      name=@   + folder = /      access=add,delete,display,modify,use,list,unlock
vartable      name=g@,$default  + folder = /  access=add,delete,display,modify,use,list,unlock
wkldappl      name=@     + folder = /     access=add,delete,display,modify,list,unlock
lob           name=@            access=use
folder         name=/           access=@
end
############################################################
```

Windows Users in MYWINDOM whose name begins with 'r' can display only userjobs. All others MYWINDOM Windows user that are logged in on any workstation are given unrestricted access to all objects. Access to workstations is limited to workstations defined in the root (/) folder. Access to scheduling objects that can be defined in folders is limited to the root (/) folder, as specified. For example, access to prompts is limited to prompts defined in the root folder `prompt  + folder = / access=@`. They are the only ones who can generate all kinds of plans and who can create, update, and delete event rule definitions. All users have access to all variable tables beginning with "g" and to the default table, irrespective of the default variable table name.

Note: Starting with version 9.2, due to support of the Windows users in User Principal Name (UPN) format, you have to specify the windows domain users in a different way in the Security file. In the same example for the previous version you have the following syntax:

```
user default  cpu=@ +  logon=MYWINDOM\@
..........................................
userjob       cpu=@ + logon =MYWINDOM\r@  access=display
```

Security file on the master domain manager to install fix packs or upgrade fault-tolerant agents and dynamic agents

`user MAESTRO CPU=@+LOGON=tws94user,Administrator`

```
###########################################################
# APPLIES TO tws94user and Administrator LOGGED IN ON ANY WORKSTATIONS.
############################################################
USER MAESTRO
  CPU=@+LOGON=tws94user,Administrator
BEGIN
  USEROBJ  CPU=@  + cpufolder = / ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,ALTPASS,LIST,UNLOCK
  JOB      CPU=@  + folder = /  + cpufolder = /
                  ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,CONFIRM,DELDEP,DELETE,DISPLAY,
                    KILL,MODIFY,RELEASE,REPLY,RERUN,SUBMIT,USE,LIST,UNLOCK,SUBMITDB,RUN
  SCHEDULE CPU=@  + folder = /  + cpufolder = /
                  ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,DELDEP,DELETE,DISPLAY,LIMIT,
                    MODIFY,RELEASE,REPLY,SUBMIT,LIST,UNLOCK
  RESOURCE CPU=@ + folder = / + cpufolder = / ACCESS=ADD,DELETE,DISPLAY,MODIFY,
                  RESOURCE,USE,LIST,UNLOCK
  PROMPT          ACCESS=ADD,DELETE,DISPLAY,MODIFY,REPLY,USE,LIST,UNLOCK
  FILE  NAME=@    ACCESS=BUILD,DELETE,DISPLAY,MODIFY,UNLOCK
  CPU      CPU=@   + folder = /
                  ACCESS=ADD,CONSOLE,DELETE,DISPLAY,FENCE,LIMIT,LINK,
                     MODIFY,SHUTDOWN,START,STOP,UNLINK,LIST,UNLOCK,RUN,RESETFTA,MANAGE
  PARAMETER CPU=@  + cpufolder = /  ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
  CALENDAR        ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
  REPORT    NAME=@ ACCESS=DISPLAY
  EVENTRULE NAME=@ ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
  ACTION    PROVIDER=@  ACCESS=DISPLAY,SUBMIT,USE,LIST
  EVENT     PROVIDER=@  ACCESS=USE
  VARTABLE  NAME=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
  WKLDAPPL  NAME=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
  RUNCYGRP  NAME=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
  LOB       NAME=@  ACCESS=USE
  folder    NAME=/  ACCESS=@
END
########################################################################
```

The default MAESTRO definition applies to Dynamic Workload Console and CLI access for `tws94user` and `Administrator` users logged into any workstation in the network. They can install a fix pack or upgrade to a later version fault-tolerant agents and dynamic agents in the network simultaneously.

For more information about this feature, see the section about centralized agent update in *Planning and Installation Guide*.

Chapter 4. Configuring authentication

This section describes how to configure authentication using, amongst other methods, the popular LDAP (Lightweight Directory Access Protocol). It is divided into these main topics:

Where to configure authentication

Authentication must be configured for each WebSphere Application Server Liberty profile, following these rules:

To authenticate command-line users

For users of the command-line, the command-line client, and the command-line as clients connected to the master domain manager using HTTP or HTTPS, the same authentication method must be configured for the following components:

- Master domain manager
- Backup master domain manager

To authenticate Z connector users

The Z connector is always installed on the same instance as the Dynamic Workload Console. You do not need to separately configure authentication for it.

To authenticate dynamic domain manager users

The same authentication method must be configured for each dynamic domain manager and its corresponding backup dynamic domain manager. This authentication method does not need to be the same as that used for the master domain manager.

Available configurations

On installation, all HCL Workload Automation components that use WebSphere Application Server Liberty are configured by default to use a local file-based user repository. For information about supported authentication mechanisms in WebSphere Application Server Liberty see the section about authenticating users in WebSphere Application Server Liberty.

You can implement an LDAP-based user repository by configuring the sample authentication templates provided in XML format.

If you choose to enable an LDAP-based user repository, for your convenience, a set of sample configuration templates are provided in XML format. See Configuring HCL Workload Automation using templates on page 422 for a list of the templates. You can further customize the templates by adding additional elements to the XML files. For a full list of the elements that you can configure to complement or modify the configuration, see the section about LDAP user registry in WebSphere Application Server Liberty documentation.

Rules for using a Federated User Registry with HCL Workload Automation

This section describes the simple rules you must follow when configuring HCL Workload Automation to use a Federated User Registry:

No duplicate User IDs

You can define any number of user registries in a Federated User Registry. However, no user ID must be present in more than one registry and no user ID must be present twice in the same registry. Thus, if you configure multiple user registries it is because you have users in different non-inclusive groups that use different user registries and which need to access HCL Workload Automation.

Completing the LDAP configuration

About this task

After you have configured the WebSphere Application Server Liberty to use a new authentication configuration, whichever configuration method you used, you must also update the security file, and propagate the changes in your environment.

Updating the security file

About this task

If you use the classic security model, you need to update the HCL Workload Automation security file to allow users to access HCL Workload Automation objects. For more information, see the section about updating the security file in *Administration Guide*. The following example shows an updated security file, where the user `TEST_LDAP` has been added to the `USER MAESTRO` section:

```
USER MAESTRO
 CPU=@+LOGON=tws83,Administrator,administrator,TEST_LDAP
BEGIN
 USEROBJ CPU=@  ACCESS=ADD,DELETE,DISPLAY,MODIFY,ALTPASS,UNLOCK,LIST
 JOB     CPU=@  + FOLDER = /   ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,CONFIRM,DELDEP,DELETE,DISPLAY,KILL,
                      MODIFY,RELEASE,REPLY,RERUN,SUBMIT,USE,LIST,UNLOCK
 SCHEDULE     CPU=@   + FOLDER = /  ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,DELDEP,DELETE,
      DISPLAY,LIMIT,MODIFY,RELEASE,REPLY,SUBMIT,LIST,UNLOCK
 RESOURCE     CPU=@   + FOLDER = /  + CPUFOLDER = / ACCESS=ADD,DELETE,DISPLAY,MODIFY,RESOURCE,
                                                          USE,LIST,UNLOCK
 PROMPT        + FOLDER = / ACCESS=ADD,DELETE,DISPLAY,MODIFY,REPLY,USE,LIST,UNLOCK
 FILE   NAME=@ ACCESS=CLEAN,DELETE,DISPLAY,MODIFY,UNLOCK
 CPU    CPU=@  + FOLDER = /   ACCESS=ADD,CONSOLE,DELETE,DISPLAY,FENCE,LIMIT,LINK,MODIFY,
      SHUTDOWN,START,STOP,UNLINK,LIST,UNLOCK
 PARAMETER   CPU=@  + FOLDER = /  + CPUFOLDER = /  ACCESS=ADD,DELETE,DISPLAY,MODIFY,UNLOCK,LIST
 CALENDAR     + FOLDER = /  ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,UNLOCK,LIST
```

```
        FOLDER      NAME=/          ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK, ACL
END
```

In this example, the useDomainQualifiedUserNames security property is set to false therefore the user name has been specified without the domain.

Propagating the changes

About this task

Propagate the changes you have made as follows:

1. If your changes involved changing the primary WebSphere Application Server Liberty administrator, then update the wa_user.xml file with the credentials. The `wauser_variables.xml` file can be found in the path:

   Dynamic Workload Console

   ```
   DWC_DATA_dir/usr/servers/dwcServer/configDropins/overrides
   ```

   master domain manager

   ```
   TWA_DATA_DIR/usr/servers/engineServer/configDropins/overrides
   ```

   Dynamic Workload Console

   ```
   DWC_home\usr\servers\dwcServer\configDropins\overrides
   ```

   master domain manager

   ```
   TWA_home\usr\servers\engineServer\configDropins\overrides
   ```

   a. Copy the `wauser_variables.xml` file for both the Dynamic Workload Console and the master domain manager to a temporary directory.
   b. Create a copy of the original `wauser_variables.xml` file for both the Dynamic Workload Console and the master domain manager in another directory for backup purposes.
   c. Edit the files in the temporary directory with the updated information about the primary WebSphere Application Server Liberty administrator.
   d. Copy the updated `wauser_variables.xml` files to the `overrides` directory on both the Dynamic Workload Console and the master domain manager.
2. Update the USERNAME and PASSWORD fields in the `useropts` file on every command-line client that points to your workstation.
3. Update the USERNAME and PASSWORD fields in the `useropts` file on every fault-tolerant agent in your environment that has an HTTP/HTTPS connection defined in localopts that points to your workstation. The HTTP/HTTPS connection is used to submit a predefined job or job stream.
4. Update the USERNAME and PASSWORD fields in the engine connection parameters on every connected Dynamic Workload Console.

Example

Note: To change the `useropts` file, change the USERNAME and type the new PASSWORD in plain text between double quotation marks. The password will be encrypted the first time you log in.

Configuring an IBM Tivoli Directory Server

Enable web single sign-on and use IBM Tivoli Directory Server as an identity provider.

About this task

Client applications, for example, the Dynamic Workload Console, can verify the identity of a user by relying on authentication from IBM Tivoli Directory Server. You can configure the WebSphere Application Server Liberty server to function as IBM Tivoli Directory Server to take advantage of web single sign-on and to use IBM Tivoli Directory Server as an identity provider.

To simply the configuration of the WebSphere Application Server Liberty server, a sample configuration file in XML format is provided named `auth_IDS_config.xml`.

Update the configuration file with the details about your identity provider.

a. Copy the template file to a working directory. The template is located in the following path:

   UNIX

   ```
   DWC_DATA_dir/usr/servers/dwcServer/configDropins/templates/authentication
   ```

   Windows

   ```
   DWC_home\usr\servers\dwcServer\configDropins\templates\authentication
   ```

b. Edit the template file in the working directory with the desired configuration.

c. Optionally, create a backup copy of the configuration file `authentication_config.xml` present in the `overrides` directory in a different directory.
   Ensure you do not copy the backup file in the path where the template files are located.

d. The `overrides` directory is located in the following path:

   UNIX

   ```
   DWC_DATA_dir/usr/servers/dwcServer/configDropins/overrides
   ```

   Windows

   ```
   DWC_home\usr\servers\dwcServer\configDropins\overrides
   ```

e. Copy the updated template file to the `overrides` directory, renaming it to `authentication_config.xml` to override the original `authentication_config.xml` file.

Alternatively, if you prefer maintaining the original name of the template, ensure you delete `authentication_config.xml` after you have copied the updated template file to the `overrides` directory to avoid conflicts.

f. Stop and restart WebSphere Application Server Liberty using the stopappserver and startappserver commands located in `TWA_home/appservertools`.

What to do next

For more detailed information about the IBM Tivoli Directory Server parameters and values to configure in the `auth_IDS_config.xml` file, see the related WebSphere Application Server Liberty documentation at Configuring LDAP user registries in Liberty.

Example configurations of LDAP servers for IDS

Refer to this template if you are using an IBM Tivoli Directory Server (IDS). This file describes a default configuration. For more advanced and specific configurations, refer to the relevant WebSphere Application Server Liberty documentation, for example at Configuring LDAP user registries in Liberty or to your LDAP administrator.

IBM Directory Server

```
<server description="federated_basicLDAP">


 <variable name="admin.group.name" value="Admins"/>


 <variable name="ldap.base.DN" value=""/>


 <variable name="ldap.port" value=""/>


 <variable name="ldap.host" value=""/>


 <variable name="ldap.adminDN" value=""/>


 <variable name="ldap.password" value=""/>

 <jndiEntry value="${admin.group.name}" jndiName="admin.group.name" />


 <administrator-role>
   <group>${admin.group.name}</group>
 </administrator-role>


 <federatedRepository searchTimeout="20m">
   <primaryRealm name="TWSRealm" allowOpIfRepoDown="true">
             <participatingBaseEntry name="o=BasicRealm"/>
```

```
            <participatingBaseEntry name="${ldap.base.DN}"/>
            <uniqueGroupIdMapping inputProperty="uniqueName" outputProperty="uniqueName"/>
   <groupSecurityNameMapping inputProperty="cn" outputProperty="cn"/>
   <groupDisplayNameMapping inputProperty="cn" outputProperty="cn"/>
   <userDisplayNameMapping inputProperty="principalName" outputProperty="principalName"/>
   <userSecurityNameMapping inputProperty="principalName" outputProperty="principalName"/>
   <uniqueUserIdMapping inputProperty="uniqueName" outputProperty="uniqueName"/>
        </primaryRealm>
 </federatedRepository>


 <ldapRegistry
  baseDN="${ldap.base.DN}"
  ldapType="IBM Tivoli Directory Server"
 port="${ldap.port}"
 host="${ldap.host}"
 id="ldap"
 bindDN="${ldap.adminDN}"
 bindPassword="${ldap.password}"
 searchTimeout="20"
 sslEnabled="false"
 sslRef="twaSSLSettings"
    userFilter="(&amp;(uid=%v)(objectclass=ePerson))"
 groupFilter="(&amp;(cn=%v)(|(objectclass=groupOfNames)
  (objectclass=groupOfUniqueNames)(objectclass=groupOfURLs)))"
 userIdMap="*:uid"
 groupIdMap="*:cn"
 groupMemberIdMap="mycompany-allGroups:member;
  mycompany-allGroups:uniqueMember;
  groupOfNames:member;
  groupOfUniqueNames:uniqueMember">
  <ldapEntityType name="Group">
    <objectClass>groupOfNames</objectClass>
   </ldapEntityType>
   <ldapEntityType name="PersonAccount">
    <objectClass>inetOrgPerson</objectClass>
   </ldapEntityType>
   <ldapEntityType name="OrgContainer">
    <objectClass>organization</objectClass>
    <objectClass>organizationalUnit</objectClass>
    <objectClass>domain</objectClass>
    <objectClass>container</objectClass>
   </ldapEntityType>



</ldapRegistry>
basicRegistry id="basic" realm="BasicRealm">

       user name="${user.twsuser.id}" password="${user.twsuser.password}"/>

       group name="${admin.group.name}">
            member name="${user.twsuser.id}"/>
           </group>
  </basicRegistry>

</server>
```

Configuring Microsoft Active Directory

Enable web single sign-on and use Microsoft Active Directory as an identity provider.

About this task

Client applications, for example, the Dynamic Workload Console, can verify the identity of a user by relying on authentication from Microsoft Active Directory. You can configure the WebSphere Application Server Liberty server to function as Microsoft Active Directory to take advantage of web single sign-on and to use Microsoft Active Directory as an identity provider.

To simply the configuration of the WebSphere Application Server Liberty server, a sample configuration file in XML format is provided named `auth_AD_config.xml`.

Update the configuration file with the details about your identity provider.

a. Copy the template file to a working directory. The template is located in the following path:

   UNIX

   ```
   DWC_DATA_dir/usr/servers/dwcServer/configDropins/templates/authentication
   ```

   Windows

   ```
   DWC_home\usr\servers\dwcServer\configDropins\templates\authentication
   ```

b. Edit the template file in the working directory with the desired configuration.

c. Optionally, create a backup copy of the configuration file `authentication_config.xml` present in the `overrides` directory in a different directory.
   Ensure you do not copy the backup file in the path where the template files are located.

d. The `overrides` directory is located in the following path:

   UNIX

   ```
   DWC_DATA_dir/usr/servers/dwcServer/configDropins/overrides
   ```

   Windows

   ```
   DWC_home\usr\servers\dwcServer\configDropins\overrides
   ```

e. Copy the updated template file to the `overrides` directory, renaming it to `authentication_config.xml` to override the original `authentication_config.xml` file.

Alternatively, if you prefer maintaining the original name of the template, ensure you delete `authentication_config.xml` after you have copied the updated template file to the `overrides` directory to avoid conflicts.

f. Stop and restart WebSphere Application Server Liberty using the stopappserver and startappserver commands located in `TWA_home/appservertools`.

What to do next

For more detailed information about Microsoft Active Directory parameters and values to configure in the `auth_AD_config.xml` file, see the related WebSphere Application Server Liberty documentation, for example Configuring LDAP user registries in Liberty.

Example configurations of LDAP servers for Microsoft Active Directory

Refer to this template if you are using Microsoft Active Directory. This file describes a default configuration. For more advanced and specific configurations, refer to the relevant WebSphere Application Server Liberty documentation at Configuring LDAP user registries in Liberty or to your LDAP administrator.

```
<server description="federated_basicLDAP">

 <!--
 This variable specifies the group name containing the primary DWC's Administrator users.
 It can be a group defined in file based userRegisty (into <basicRegistry> section) or in your LDAP-based
       directory services authentication.
 -->
 <variable name="admin.group.name" value="Admins"/>

 <!--
 The value of your Base distinguished name (DN) of the directory service, which indicates the starting point
 for LDAP searches in the directory service.
 Sample: <variable name="ldap.base.DN" value="o=domain,c=us"/>
  -->
 <variable name="ldap.base.DN" value="DC=TWS,DC=COM"/>

 <!--
 The Port number of the LDAP server.
 Sample: <variable name="ldap.port" value="389"/>
  -->
 <variable name="ldap.port" value="389"/>

 <!--
 The Address of the LDAP server in the form of an IP address or a domain name service (DNS) name.
 Sample: <variable name="ldap.host" value="host.domain.com"/>
  -->
 <variable name="ldap.host" value="<your_host_name>"/>

 <!--
 The Distinguished name (DN) for the application server, which is used to bind to the directory service.
        Specify a user defined in Microsoft Active Directory Server with look-up rights.
        Sample: <variable name="ldap.adminDN" value="cn=testuser,o=domain,c=us"/>
  -->
 <variable name="ldap.adminDN" value="CN=Operators,DC=TWS,DC=COM"/>
 <!--
 The Distinguished name (DN) for the application server, which is used to bind to the directory service.
 You can use the liberty provided tool <wlp_dir>/bin/securityUtility to know the encrypted value
      of your password.
  1. run: <wlp_dir>/bin/securityUtility encode mypassword
  2. output: {xor}MiYvPiwsKDAtOw==
  3. fill the value field with the printed output value
 Sample: <variable name="ldap.password" value="{xor}MiYvPiwsKDAtOw=="/>
  -->
 <variable name="ldap.password" value=""/>

 <jndiEntry value="${admin.group.name}" jndiName="admin.group.name" />

 <!-- Assign 'admin' to Administrator -->
    <administrator-role>
       <group>${admin.group.name}</group>
    </administrator-role>

 <!--
  Details about how to configure LDAP registry and federate it with basic registry, can be found following this
 link:

 https://www.ibm.com/support/knowledgecenter/en/SSAW57_liberty/com.ibm.websphere.wlp.nd.multiplatform.doc/ae/tw
lp_sec_ldap.html

 https://www.ibm.com/support/knowledgecenter/en/SSEQTP_liberty/com.ibm.websphere.wlp.doc/ae/cwlp_repository_fed
eration.html

  To troubleshoot any LDAP authentication issues, copy trace.xml in overrides with the following
 traceSpecification:
   traceSpecification="com.ibm.ws.security.wim.*=all:com.ibm.websphere.security.wim.*=all"
 -->
 <federatedRepository searchTimeout="20m">
   <primaryRealm name="TWSRealm" allowOpIfRepoDown="true">
             <participatingBaseEntry name="o=BasicRealm"/>
             <participatingBaseEntry name="${ldap.base.DN}"/>
             <uniqueGroupIdMapping inputProperty="uniqueName" outputProperty="uniqueName"/>
    <groupSecurityNameMapping inputProperty="cn" outputProperty="cn"/>
    <groupDisplayNameMapping inputProperty="cn" outputProperty="cn"/>
    <userDisplayNameMapping inputProperty="principalName" outputProperty="principalName"/>
    <userSecurityNameMapping inputProperty="principalName" outputProperty="principalName"/>
    <uniqueUserIdMapping inputProperty="uniqueName" outputProperty="uniqueName"/>
         </primaryRealm>
  </federatedRepository>

 <!--
  Note for LDAP directory service configured in SSL:
   1. the settings sslEnabled to "true"
   2. Import the LDAP certificate in trustStore used by the server,
      (it is defined in configDropins/defaults/ssl_comfig.xml file, the default one is
                       resources/security/TWSServerTrustFile).
      For importing the exported LDAP certificate your_ldap.cert run
    $JAVA_HOME/bin/keytool -import -file ./your_ldap.cert -alias ldapCA -keystore
                          resources/security/TWSServerTrustFile
  -->
  <ldapRegistry id="AD"
      host="${ldap.host}" port="${ldap.port}" ignoreCase="true"
      baseDN="${ldap.base.DN}"
      bindDN="${ldap.adminDN}"
      bindPassword="${ldap.password}"
      ldapType="Microsoft Active Directory"
      sslEnabled="false"
      sslRef="twaSSLSettings">
     <activedFilters
      userFilter="(&amp;(sAMAccountName=%v)(objectcategory=user))"
   groupFilter="(&amp;(cn=%v)(objectcategory=group))"
     userIdMap="*:sAMAccountName"
      groupIdMap="*:cn"
      groupMemberIdMap="memberOf:member" >
  </activedFilters>
    </ldapRegistry>

 <basicRegistry id="basic" realm="BasicRealm">
        <!--  DO NOT DELETE -->
        <user name="${user.twsuser.id}" password="${user.twsuser.password}"/>
        <!--  END DO NOT DELETE -->
        <group name="${admin.group.name}">
             <member name="${user.twsuser.id}"/>
            </group>

    <!-- Sample for adding other users or group in file based user registry. -->
    <!--
     <user name="nonadmin" password="nonadmin"/>
    <user name="analyst" password="analyst"/>
      <user name="developer" password="developer"/>
      <user name="configurator" password="configurator"/>
      <user name="operator" password="operator"/>
      <group name="Admins">
             <member name="${user.twsuser.id}"/>
          </group>
    -->
  </basicRegistry>

</server>
```

If you have nested groups in your Microsoft Active Directory, ensure you set the recursiveSearch property in the ldapRegistry id="AD" section to `true`, as follows:

```
......
<ldapRegistry id="AD"
      host="${ldap.host}" port="${ldap.port}" ignoreCase="true"
      baseDN="${ldap.base.DN}"
      bindDN="${ldap.adminDN}"
      bindPassword="${ldap.password}"
      ldapType="Microsoft Active Directory"
                 recursiveSearch="true"
      sslEnabled="false"
      sslRef="twaSSLSettings">
     .........
    </ldapRegistry>
```

Configuring an OpenID Connect Client

Enable web single sign-on and use the OpenID Connect Provider as an identity provider.

About this task

Client applications, for example, the Dynamic Workload Console, can verify the identity of a user by relying on authentication from an OpenID Connect Provider. You can configure the WebSphere Application Server Liberty server to function as an OpenID Connect Client to take advantage of web single sign-on and to use the OpenID Connect Provider as an identity provider.

To simply the configuration of the WebSphere Application Server Liberty server, a sample configuration file in XML format is provided named `openid_connect.xml`.

Update the configuration file with the details about your identity provider.

a. Copy the template file to a working directory. The template is located in the following path:

   UNIX

   ```
   DWC_DATA_dir/usr/servers/dwcServer/configDropins/templates/authentication
   ```

   Windows

   ```
   DWC_home\usr\servers\dwcServer\configDropins\templates\authentication
   ```

b. Edit the template file in the working directory with the desired configuration.

c. Optionally, create a backup copy of the configuration file `authentication_config.xml` present in the `overrides` directory in a different directory.
   Ensure you do not copy the backup file in the path where the template files are located.

d. The `overrides` directory is located in the following path:

   UNIX

   ```
   DWC_DATA_dir/usr/servers/dwcServer/configDropins/overrides
   ```

   Windows

   ```
   DWC_home\usr\servers\dwcServer\configDropins\overrides
   ```

e. Copy the updated template file to the `overrides` directory, renaming it to `authentication_config.xml` to override the original `authentication_config.xml` file.
   Alternatively, if you prefer maintaining the original name of the template, ensure you delete `authentication_config.xml` after you have copied the updated template file to the `overrides` directory to avoid conflicts.

f. Stop and restart WebSphere Application Server Liberty using the stopappserver and startappserver commands located in `TWA_home/appservertools`.

What to do next

For more detailed information about the OpenID parameters and values to configure in the `openid_connect.xml` file, see the related WebSphere Application Server Liberty documentation at Configuring an OpenID Connect Client in Liberty.

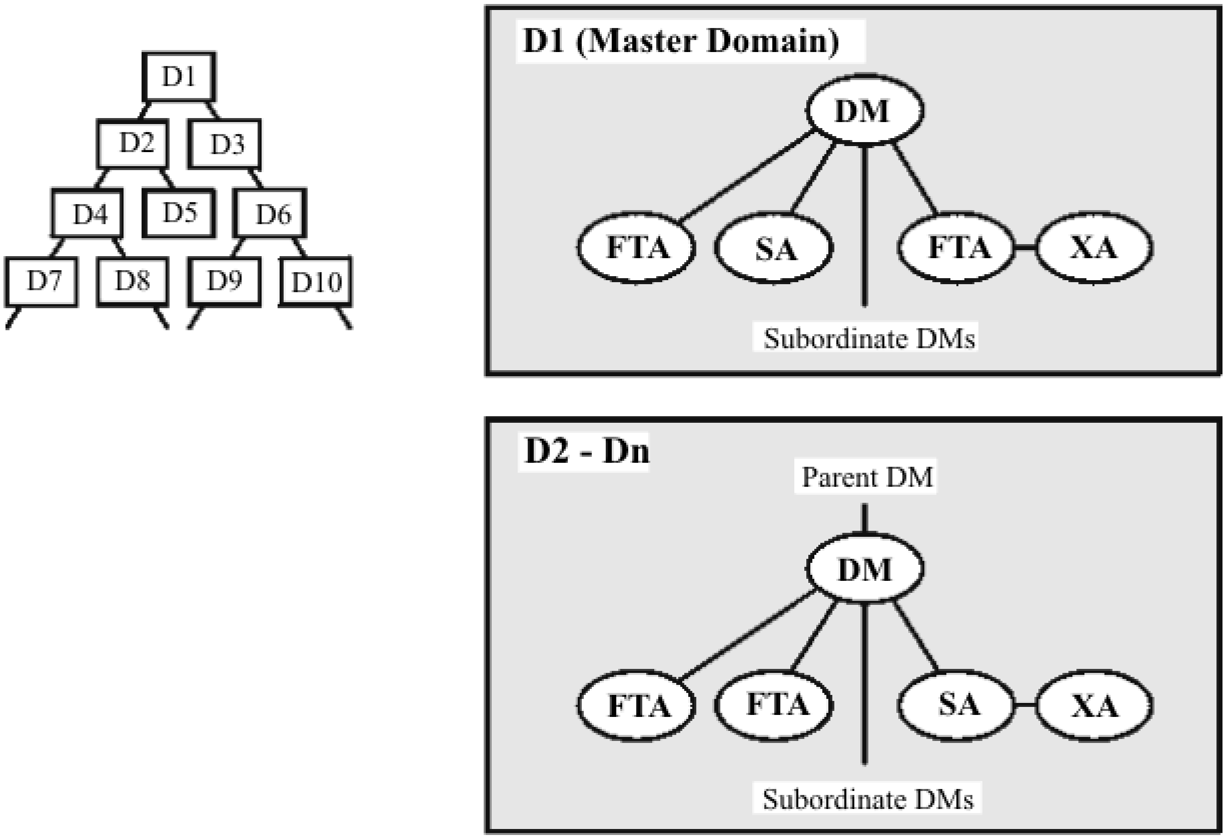
Chapter 5. Network administration

This chapter describes how to administer the HCL Workload Automation network. It has the following topics:

Network overview

A HCL Workload Automation network consists of one or more domains arranged hierarchically. A HCL Workload Automation domain is a logical grouping of workstations, consisting of a domain manager and a number of agents.

Figure 6. HCL Workload Automation network domain structure



Network definition

Domain

A named group of HCL Workload Automation workstations consisting of one or more agents and a domain manager. All domains have a parent, except the master domain.

Master domain

The topmost domain in an HCL Workload Automation network.

Master domain manager

The domain manager in the topmost domain of an HCL Workload Automation network. It contains the centralized master files used to document scheduling objects. It creates the Production Control file (Symphony) at the start of each production period and performs all logging and reporting for the network. See also Domain Manager.

Backup master domain manager

A fault-tolerant agent capable of assuming the responsibilities of the master domain manager.

Parent domain

The domain directly above the current domain. All domains, except the master domain, have a parent domain. All communications to/from a domain is rooted through the parent domain manager.

Domain Manager

The management hub in a domain. All communications in and from the agents in a domain is routed through the domain manager. See also Master Domain Manager.

Backup domain manager

A fault-tolerant agent capable of assuming the responsibilities of its domain manager.

Fault-tolerant agent

An agent workstation capable of resolving local dependencies and launching its jobs in the absence of a domain manager.

Standard agent

An agent workstation that launches jobs only under the direction of its domain manager.

Extended agent

An agent workstation that launches jobs only under the direction of its host. Extended agents can be used to interface HCL Workload Automation with non-HCL Workload Automation systems and applications

Dynamic agent

A workstation that manages a wide variety of job types, for example, specific database or FTP jobs, in addition to existing job types. This workstation is automatically created and registered when you install the dynamic agent. Because the installation and registration processes are performed automatically, when you view the agent in the Dynamic Workload Console, it results as updated by the Resource Advisor Agent. You can group agents in pools and dynamic pools.

In a simple configuration, dynamic agents connect directly to a master domain manager or to a dynamic domain manager. However, in more complex network topologies, if the network configuration prevents the master domain manager or the dynamic domain manager from directly communicating with the dynamic agent, then you can configure your dynamic agents to use a local or remote gateway.

Host

The scheduling function required by extended agents. It can be performed by any HCL Workload Automation workstation, except another extended agent.

Network communications

In a HCL Workload Automation network, agents communicate with their domain managers, and domain managers communicate with their parent domain managers. There are basically two types of communications that take place:

- Start-of-production period initialization (distribution of new Symphony file)
- Scheduling events in the form of change-of-state messages during the production period

Before the start of each new production period, the master domain manager creates a production control file called Symphony. Then, HCL Workload Automation is restarted in the network, and the master domain manager sends a copy of the new `Symphony` file to each of its automatically-linked agents and subordinate domain managers. The domain managers, in turn, send copies to their automatically-linked agents and subordinate domain managers. Agents and domain managers that are not set up to link automatically are initialized with a copy of `Symphony` as soon as a link operation is run in HCL Workload Automation.

Once the network is started, scheduling messages, like job starts and completions, are passed from the agents to their domain managers, through parent domain managers to the master domain manager. The master domain manager then broadcasts the messages throughout the hierarchical tree to update the `Symphony` files of all domain managers and the domain managers forward the messages to all fault-tolerant agents in their domain running in *FullStatus* mode.

Network links

Links provide communications between HCL Workload Automation workstations in a network. Links are controlled by the AUTO Link flag, and the Console Manager **link** and **unlink** commands. When a link is open, messages are passed between two workstations. When a link is closed, the sending workstation stores messages in a local `pobox` file and sends them to the destination workstation when the link is reopened.

This means that when links are closed, the message queues fill up with messages for the inaccessible workstations. To maximize the performance of HCL Workload Automation, monitor workstations for closed links and attempt to reopen them as soon as possible.

Note: Extended agents do not have links. They communicate with their domain managers through their hosts.

To have a workstation link opened automatically, turn on the AUTO Link flag in the workstation's definition. The link is first opened when HCL Workload Automation is started on the Master Domain workstation. If the subdomain manager and workstations are not initialized and their AUTO Link flag is on, the master domain manager attempts to link to its subordinates and begin the initialization processes. If the AUTO Link flag is turned off, the workstation is only initialized by running a **link** command from the master domain manager. After the workstation is initialized, it automatically starts and issues a link back to its domain manager.

If you stop a workstation, the links from it to other workstations are closed. However, the links from the other workstations to it remain open until either one of the following situations occurs:

- The stopped workstation is restarted and a **link** command is issued
- The other workstations' **mailman** processes time out, and perform an **unlink** for the workstation

When the **link** command is issued and the connection has been established, if the domain manager does not receive any reply within the timeout period, the `chkhltst` service is automatically invoked by **mailman**.

This service verifies that the workstation mailbox can be successfully read, and checks if there are errors in the mailbox header. Resulting information is logged in the `TWSMERGE.log` file of the domain manager as follows:

- If a file system error occurs while opening the mailbox, the following message is reported: `AWSBDY126E An error occurred opening the Mailbox.msg file in` *`CPU_NAME`*`.`
- If an error occurs while opening the mailbox because **mailman** is reading the mailbox, the following message is reported: `AWSBDY123I The Mailbox.msg file in` *`CPU_NAME`* `is correctly read by Mailman.`
- If the mailbox is correctly opened, but an error occurs while reading the header, the following message is reported: `AWSBDY125E An error occurred reading the header of the Mailbox.msg file in` *`CPU_NAME`*`.`
- If the mailbox is correctly opened and no error occurs while reading the header, the following message is reported: `AWSBDY124W The Mailbox.msg file in` *`CPU_NAME`* `is not read by Mailman.`

This service can also be launched manually by using the **conman** command. See the *HCL Workload Automation User's Guide and Reference* for more details.

To be certain that inter-workstation communication is correctly restored, you can issue a **link** command after restarting a workstation.

Working across firewalls

In the design phase of a HCL Workload Automation network, the administrator must know where the firewalls are positioned in the network, which fault-tolerant agents and which domain managers belong to a particular firewall, and which are the entry points into the firewalls. When this has been clearly understood, the administrator should define the **behindfirewall** attribute for some of the workstation definitions in the HCL Workload Automation database. In particular, if a workstation definition is set with the **behindfirewall** attribute to ON, this means that there is a firewall between that workstation and the HCL Workload Automation master domain manager. In this case, the workstation-domain manager link is the only link allowed between the workstation and its domain manager.

All HCL Workload Automation workstations should be defined with the **behindfirewall** attribute if the link with the corresponding domain manager, or with any domain manager in the HCL Workload Automation hierarchy right up to the master domain manager, is across a firewall.

When mapping an HCL Workload Automation network over an existing firewall structure, it does not matter which fault-tolerant agents and which domain managers are on the secure side of the firewall and which ones are on the non secure side. Firewall boundaries should be the only concern. For example, if the master domain manager is in a non secure zone and some of the domain managers are in secured zones, or vice versa, does not make any difference. The firewall structure must always be considered starting from the master domain manager and following the HCL Workload Automation hierarchy, marking all the workstations that have a firewall between them and their corresponding domain manager.

For all workstations with **behindfirewall** set to ON, the conman **start** and **stop** commands on the workstation, and the **showjobs** commands are sent following the domain hierarchy, instead of making the master domain manager or the domain manager open a direct connection to the workstation. This makes a significant improvement in security.

This attribute works for multiple nested firewalls as well. For extended agents, you can specify that an extended agent workstation is behind a firewall by setting the **behindfirewall** attribute to ON, on the host workstation. The attribute is read-only in the plan; to change it in the plan, the administrator must update it in the database and then re-create the plan.

See the *HCL Workload Automation: User's Guide and Reference* for details on how to set this attribute.

Configuring dynamic agent communications through a gateway

In some complex network topologies, the master domain manager or the dynamic domain manager are prevented from directly communicating with the dynamic agent.

Before you begin

In a simple configuration, dynamic agents connect directly to the master domain manager or to the dynamic domain manager. However, in more complex network topologies, if the network configuration prevents the master domain manager or the dynamic domain manager from directly communicating with the dynamic agent, for example, if the agents are behind a firewall and need to communicate through the internet, or if they need to communicate with a Network Address Translation (NAT) process, then you can configure your dynamic agents to use a local or remote gateway.

About this task

You can set up your dynamic agents to use a gateway for communication with the master domain manager or to the dynamic domain manager when you install a dynamic agent, or you can configure a gateway subsequent to the installation.

For information about the gateway parameters available with the installation of a dynamic agent, see the section about agent installation parameters in *Planning and Installation Guide*.

To configure an existing HCL Workload Automation version 9.2 or later dynamic agent to communicate to its master domain manager or dynamic domain manager through a local gateway, perform the following configuration steps:

1. Edit the `JobManager.ini` file on the dynamic agent workstation that you want to configure to communicate through a gateway. Edit the [ResourceAdvisorAgent] section so that the value of the **ResourceAdvisorURL** parameter is `https://$(tdwb_server):$(tdwb_port)/ita/JobManagerGW/JobManagerRESTWeb/JobScheduler/resource` , where, $(*tdwb_server*) and $(*tdwb_port*) correspond to the host name and port of the gateway that you want to use for communication with the master domain manager or the dynamic domain manager.
2. Stop and start the dynamic agent to implement the changes.

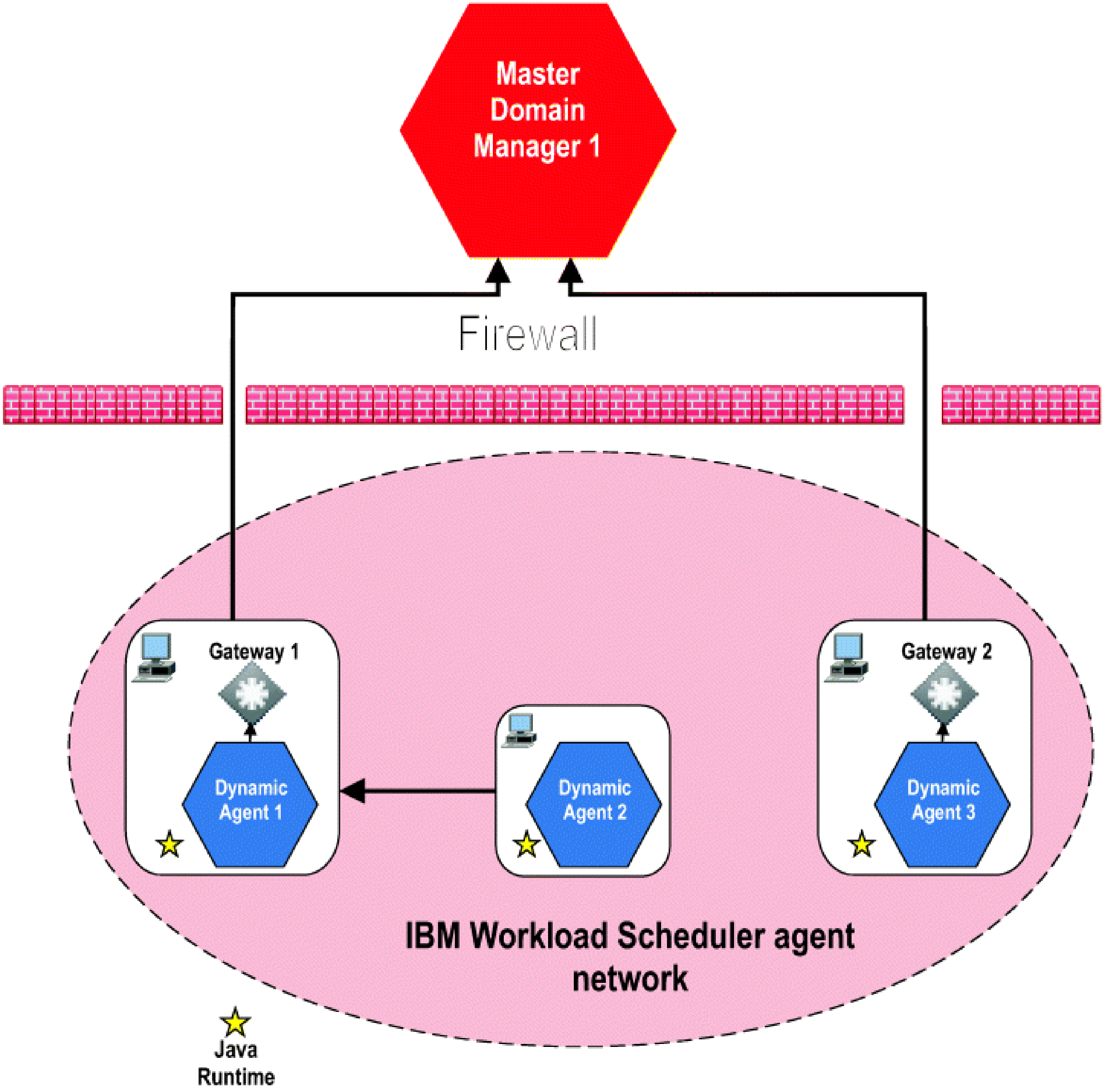
Results

The master domain manager or dynamic domain manager can now communicate with the dynamic agent workstation through the gateway.

Note: If you have more than 100 dynamic agents that communicate through one single gateway to the master domain manager or dynamic domain manager, in the `JobManagerGW.ini` file on the dynamic agent workstation where the gateway resides, set the `ActionPollers` parameter as described in Configuring general properties [ITA] on page 74.

Example

The following diagram depicts a network topology where the master domain manager communicates to the dynamic agents, located behind a firewall, through a gateway configured on one of the dynamic agents.



The following are the configuration settings used in the network topology depicted in the figure:

Table 54. Configuration settings

Dynamic Agent	Configuration File	Parameter	Value
Dynamic Agent 1 - Local gateway	JobManager.ini	Section `[ResourceAdvisorAgent]` **ResourceAdvisorUrl**	`https://$(`*tdwb_server*`):$(`*tdwb_port*`)/ita/JobManagerGW/`

Table 54. Configuration settings (continued)

Dynamic Agent	Configuration File	Parameter	Value

			`JobManagerRESTWeb/`
			`JobScheduler/resource`
			where,
			$(*tdwb_server*)
			The host name of the Dynamic Agent 1 workstation.
			$(*tdwb_port*)
			The port number of the Dynamic Agent 1 workstation.

Dynamic Agent 2 - Remote gateway	JobManager.ini	Section `[ResourceAdvisorAgent]` **ResourceAdvisorUrl**	`https://`
			`$(`*tdwb_server*`):`
			`$(`*tdwb_port*`)/ita/`
			`JobManagerGW/`
			`JobManagerRESTWeb/`
			`JobScheduler/resource`
			where,
			$(*tdwb_server*)
			The host name of the Dynamic Agent 1 workstation.
			$(*tdwb_port*)
			The port number of the Dynamic Agent 1 workstation.

Dynamic Agent 3 - Local gateway	JobManager.ini	Section `[ResourceAdvisorAgent]` **ResourceAdvisorUrl**	`https://`
			`$(`*tdwb_server*`):`
			`$(`*tdwb_port*`)/ita/`
			`JobManagerGW/`
			`JobManagerRESTWeb/`
			`JobScheduler/resource`
			where,
			$(*tdwb_server*)
			The host name of the Dynamic Agent 3 workstation.
			$(*tdwb_port*)
			The port number of the Dynamic Agent 3 workstation.

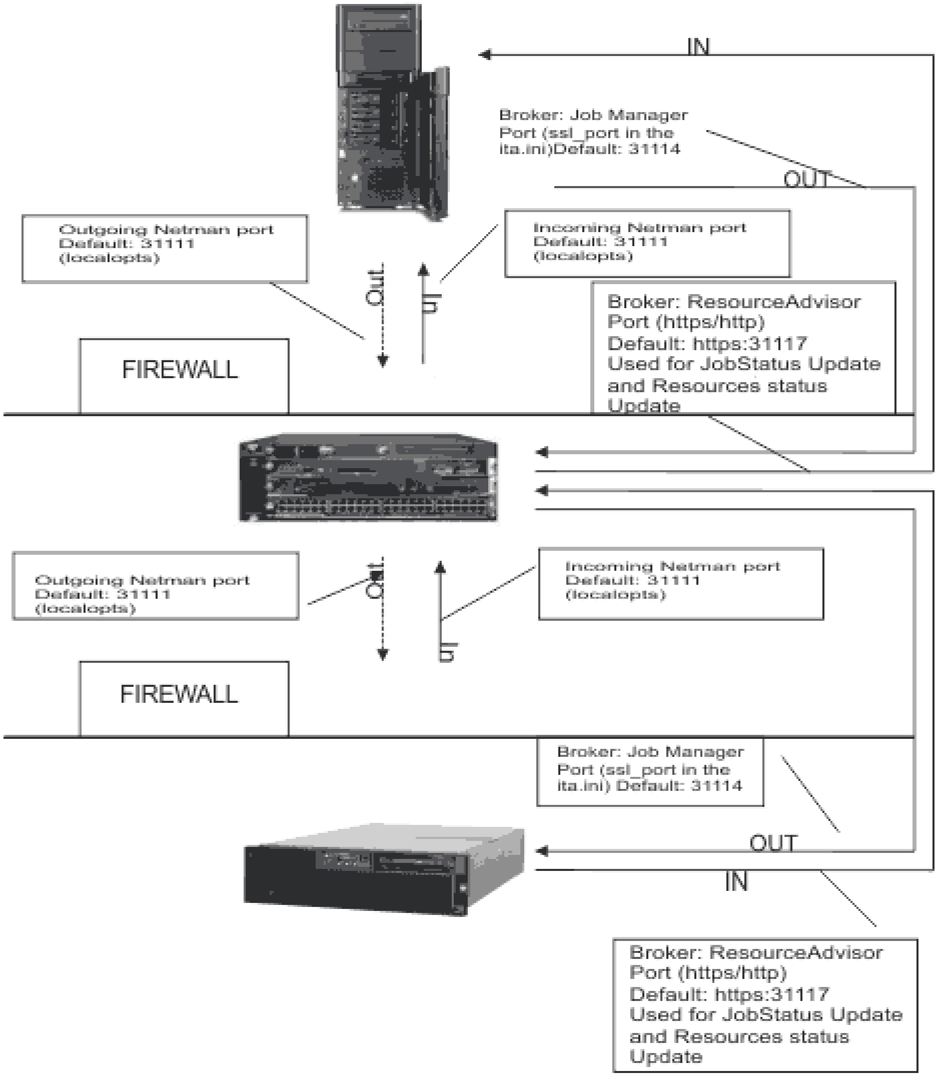
What to do next

For more information about the parameters in the `JobManager.ini` and `JobManagerGW.ini` files, see Configuring the agent on page 71.

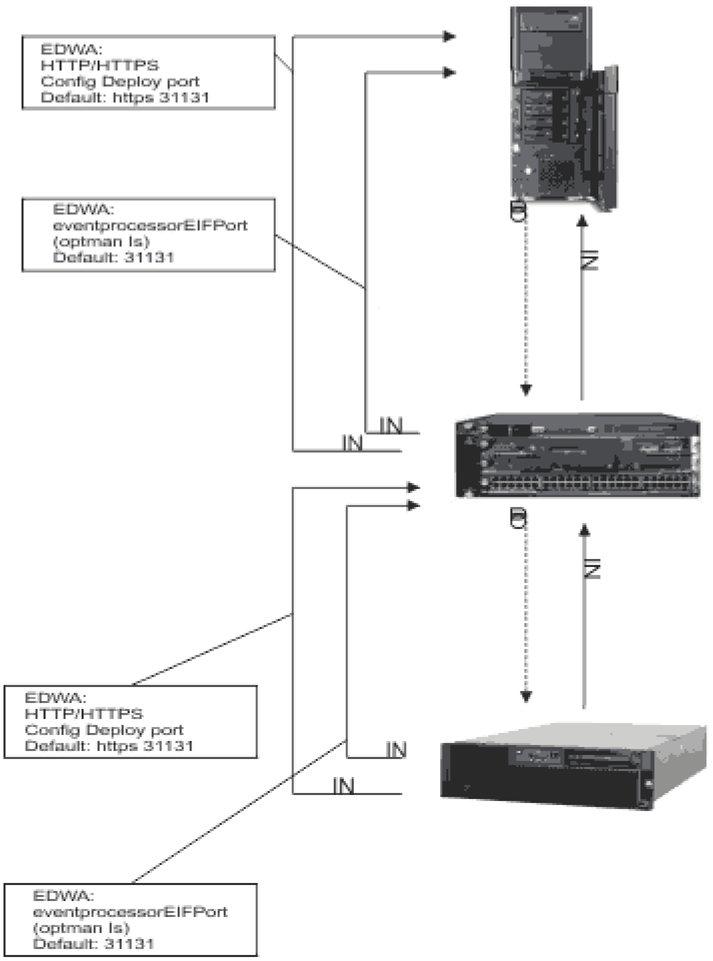
To see an example of the installation parameters that must be specified to configure a gateway when installing a dynamic agent, see the section containing example dynamic agent gateway installations in the *Planning and Installation Guide*.

Enabling Ports

When you install the master domain manager in a HCL Workload Automation network all the incoming and outgoing ports are shown in the figure below:



If you enable the event driven workload automation (EDWA) behind the firewall feature the figure below shows all the incoming and outgoing ports.



The following is the complete list of all ports:

Communication agent - master domain manager

- 31111 - incoming/outcoming Netman port (`localopts`)
- 31115 - **HTTP_PORT**
- 31116 - HTTPS Protocol port (`localopts`)
- 31113 - port used to listen for incoming SSL connections. This value must match the one defined in the nm SSL port local option of the workstation
- 31131 - **eventProcessorEIFPort** specifies the Job Manager Event Integration Facility (EIF) port number. (optman ls)
- 31132 - **-gweifport gateway_eif_port**. Specifies the Job Manager Event Integration Facility (EIF) port number. The default value is 31132. The valid range is 1 to 65535
- 5529 - EIF Probe server port. Used in event rule management

dynamic domain manager

- 31114 - dynamic domain manager **JobManager** port (**ssl_port** in the `ita.ini` file)
- 31117 - dynamic domain manager **ResourceAdvisor** port (used for **JobStatus Update** and **ResourcesStatus Update**)
- 41114 - **BROKER_NETMAN_PORT** – The TCP/IP port number used by the netman process to listen to communication from the dynamic domain manager
- 35116 - CLI connections (`localopts`)

Dynamic Workload Console

- 9444 - **HTTP_PORT**
- 9443 - **HTTPS_PORT**
- 12809 - **bootstrap** port
- 19402 - **bootstrap** security port, to be used for connecting to the Z connector

event-driven workload automation

- 31131 - HTTP/HTTPS **Config Deploy** port

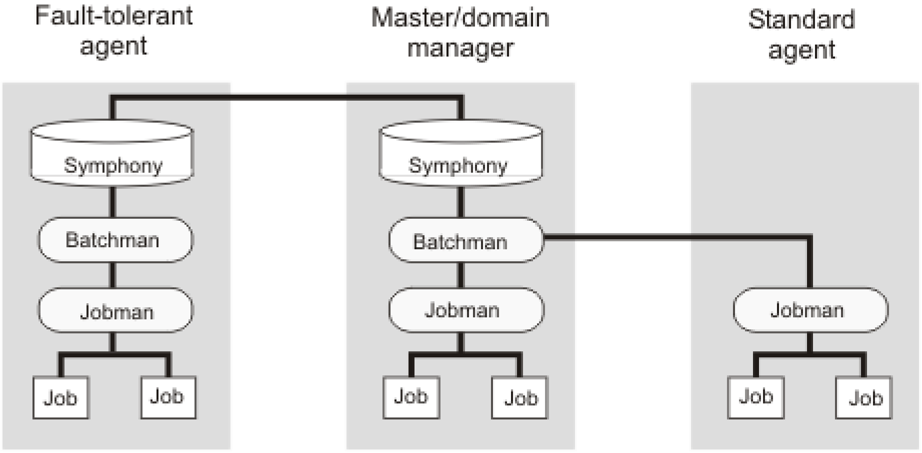
Network operation

The batchman process on each domain manager and fault-tolerant agent workstation operates autonomously, scanning its `Symphony` file to resolve dependencies and launch jobs. Batchman launches jobs via the jobman process. On a standard agent, the jobman process responds to launch requests from the domain manager's batchman.

The master domain manager is continuously informed of job launches and completions and is responsible for broadcasting the information to domain managers and fault-tolerant agents so they can resolve any inter-workstation dependencies.

The degree of synchronization among the `Symphony` files depends on the setting of the *FullStatus* mode in a workstation's definition. Assuming that these modes are turned on, a fault-tolerant agent's Symphony file contains the same information as the master domain manager's (see the section that explains how to manage workstations in the database in the *HCL Workload Automation: User's Guide and Reference*).

Figure 7. Symphony file synchronization



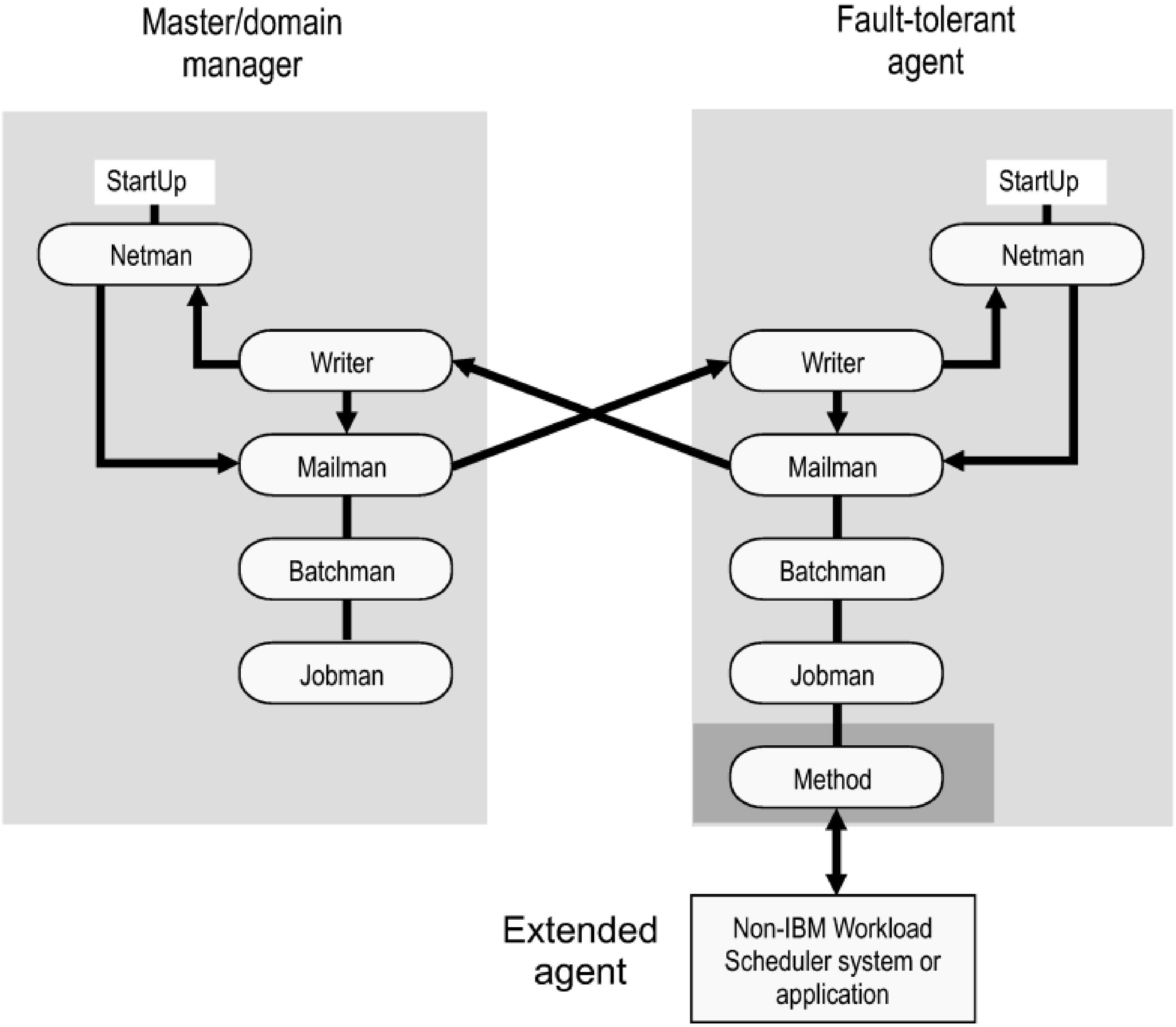
Network processes

Netman is started by the StartUp script (command). The order of process creation is netman, mailman, batchman, and jobman. On standard agent workstations, batchman does not run. All processes, except jobman, run as the **TWS** user. Jobman runs as **root**.

When network activity begins, netman receives requests from remote mailman processes. Upon receiving a request, netman creates a writer process and passes the connection off to it. Writer receives the message and passes it to the local mailman. The writer processes (there might be more than one on a domain manager) are started by link requests and are stopped by unlink requests (or when the communicating mailman terminates).

Domain managers, including the master domain manager, can communicate with a large number of agents and subordinate domain managers. For improved efficiency, you can define mailman servers on a domain manager to distribute the communications load (see the section that explains how to manage workstations in the database in the *HCL Workload Automation: User's Guide and Reference*).

Figure 8. Process creation on domain manager and fault-tolerant agent



The StartUp command is normally run automatically, but can also be run manually, as follows:

StartUp

Starts **netman**, the HCL Workload Automation network management process.

In Windows™, the **netman** service is started automatically when a computer is restarted. **StartUp** can be used to restart the service if it is stopped for any reason.

In UNIX™, the **StartUp** command can be run automatically by invoking it from the `/etc/inittab` file, so that WebSphere Application Server Liberty infrastructure and **netman** is started each time a computer is rebooted. **StartUp** can be used to restart **netman** if it is stopped for any reason.

The remainder of the process tree can be restarted with the

```
conman start
conman startmon
```

commands. See the documentation about conman in the *User's Guide and Reference* for more information.

Note: If you start the StartUp command using a remote shell, the netman process maintains the shell open without returning the prompt. To avoid this problem, modify the StartUp command so that the netman process is called in the background, as follows:

```
# Start netman
/usr/local/TWS95/mae95/TWS/bin/netman&
```

Authorization

You must have ***start*** access to the workstation.

Syntax

StartUp [**−v** | **−u**]

Arguments

−v

Displays the command version and exits.

−u

Displays command usage information and exits.

Example

Examples

To display the command name and version, run the following command:

```
StartUp -v
```

To start the **netman** process, run the following command:

```
StartUp
```

Monitoring the HCL Workload Automation processes

You can use event-driven workload automation (EDWA) to monitor the status of network processes and to start a predefined set of actions when one or more specific events take place. For more information about event-driven workload automation, refer to *User's Guide and Reference*.

You can monitor the following processes:

- agent
- appservman
- batchman
- jobman
- mailman
- monman
- netman

The .XML file contains the definition of a sample event rule to monitor the status of the specified processes on the specified workstation. This event rule calls the MessageLogger action provider to write a message in a log file in an internal auditing database. If the condition described in the rule is already existing when you deploy the rule, the related event is not generated. For more information about the MessageLogger action provider, refer to User's Guide and Reference:

```
<eventRule name="PROCESSES" ruleType="filter" isDraft="no">
 <eventCondition name="twsProcMonEvt1" eventProvider="TWSApplicationMonitor"
 eventType="TWSProcessMonitor">
   <scope>
    AGENT, BATCHMAN DOWN
   </scope>
   <filteringPredicate>
    <attributeFilter name="ProcessName" operator="eq">
     <value>process_name1</value>
    </attributeFilter>
    <attributeFilter name="TWSPath" operator="eq">
     <value>TWS_path</value>
    </attributeFilter>
    <attributeFilter name="Workstation" operator="eq">
     <value>workstation_name</value>
    </attributeFilter>
    <attributeFilter name="SampleInterval" operator="eq">
     <value>sample_interval</value>
    </attributeFilter>

   </filteringPredicate>
  </eventCondition>
  <action actionProvider="MessageLogger" actionType="MSGLOG" responseType="onDetection">
   <scope>
    OBJECT=AAAAAAA MESSAGE=TWS PROCESS DOWN: %{TWSPROCMONEVT1.PROCESSNAME}
ON %{TWSPROCMONEVT1.TWSPATH}
   </scope>
   <parameter name="ObjectKey">
    <value>object_key</value>
   </parameter>
   <parameter name="Severity">
    <value>message_severity</value>
   </parameter>
   <parameter name="Message">
    <value>log_message</value>
   </parameter>
  </action>
 </eventRule>
</eventRuleSet>
```

where:

process_name

Is the name of the process to be monitored. You can insert more that one process name, as follows:

```
<attributeFilter name="ProcessName" operator="eq">
     <value>agent</value>
     <value>batchman</value>
    </attributeFilter>
```

TWS_path

Is the directory containing the Symphony file and the bin directory.

workstation_name

Is the workstation on which the event is generated.

sample_interval

Is the interval, expressed in seconds, for monitoring the process status.

object_key

Is a key identifying the object to which the message pertains.

message_severity

Is the severity of the message.

log_message

Is the message to be logged.

Optimizing the network

The structure of a HCL Workload Automation network goes hand in hand with the structure of your enterprise's network. The structure of the domains must reflect the topology of the network in order to best use the available communication channels.

But when planning the HCL Workload Automation network, the following must be taken into consideration:

- Data volumes
- Connectivity

Data volumes

Network capacity must be planned to adapt to the amount of data that is circulating. Particularly high transmission volumes might be caused by the following:

- Transfer of large Symphony files.
- Message traffic between the master domain manager and a *FullStatus* agent.
- Message traffic from a domain manager when the domain has many agents.
- Heavy use of internetwork dependencies, which extends traffic to the entire network.

Connectivity

For the more critical agents in your network, you need to consider their position in the network. The reliability of workload execution on a particular agent depends on its capacity to receive a fresh Symphony file at the start of the production period. If the workload contains many dependencies, a reliable connection to the rest of the network is also required. These factors suggest that the best place for critical agents is in the master domain, or to be set up as domain managers immediately under the master domain manager, possibly receiving their Symphony files through a set of dedicated mailman servers. Further, it is important for critical agents that any domain manager above them in the tree structure must be hosted on powerful systems and must have an adequate backup system to ensure continuity of operation in the event of problems.

HCL Workload Automation provides two mechanisms to accommodate a particular network situation: the domain structure and mailman servers. Whereas domain structure establishes a hierarchy among HCL Workload Automation agents, mailman servers are used to tune the resources dedicated to the connection between two agents.

Domain

Use the HCL Workload Automation domain structure mechanism to create a tree-shaped structure for the network, where all communications between two points use the unique path defined by the tree (climb to the common ancestor and go down to the target, as opposed to direct TCP communication). As a consequence, the domain structure separates the network into more-manageable pieces. This is for easier filtering, overview, action, and monitoring. However, it does also introduce some delay in the workload processing. For instance when distributing the Symphony file, a fault-tolerant agent inside a domain needs to wait for two steps of Symphony distribution to be completed (from master domain manager to domain manager and from domain manager to fault-tolerant agent). The same is valid for every other type of communication that comes from the master domain manager.

This has the following implications:

- Critical business activities must be as close as possible to the master domain manager
- The domain manager must be installed on as powerful a workstation as possible
- A similarly powerful backup domain manager must be included in the network
- The network link between the domain manager and its backup must be as fast as possible to pass all the updates received from the subtree
- If intervention is needed directly on the domain, either give shell access to the operators to use the HCL Workload Automation command line, or install a connector so that the Dynamic Workload Console can be used.

Mailman servers

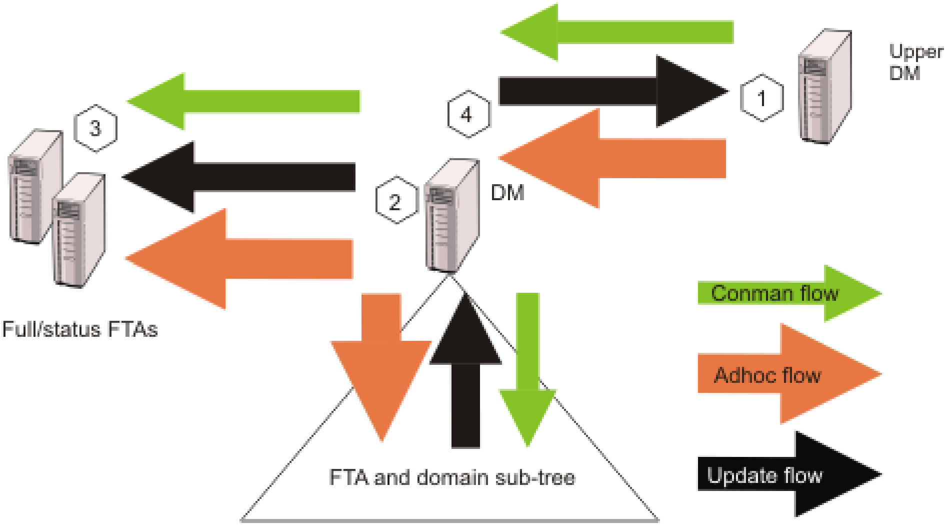
Mailman servers allocate separate processes dedicated to the communication with other workstations. The main mailman is dedicated to the transfer and network hub activities. The use of mailman servers on the domain manager must be carefully planned. The main parameter is the number of downstream connections at each level of the tree. This number describes the number of mailman servers that a main mailman is connected to, or the number of agents a mailman server is connected to. The maximum number of downstream connections is about 50 for Windows™ and about 100 for other UNIX™ workstations, depending on their power.

Typical downstream connections is about 15 for Windows™ and about 20 for other UNIX™ workstations. However, you must also take into consideration the link speed and the queue sizes, discussed below.

Planning space for queues

In order to plan space for event queues, and possible alert levels and reactions, it is necessary to model the flows passing through the agents, and the domain managers in particular.

Figure 9. Typical HCL Workload Automation network flows.



For a typical domain manager, the main flow comes from update activity reported by the sub tree, and from ad hoc submissions arriving from the master domain manager and propagating to the entire network. Under these conditions, the most critical errors are listed by order of importance in Table 55: Critical flow errors on page 294:

Table 55. Critical flow errors

Flow no.	Location	Queue	Risk	Impact
1	Upper domain manager	dm.msg	The queue fills up because of too many unlinked workstations in the domain or a downstream domain manager has failed.	The upper domain manager fails and propagates the error.
2	Domain manager	*FullStatus* fta.msg	The queue fills because of too many unlinked workstations in the domain or because the *FullStatus* fault-tolerant agent is not coping with the flow.	The domain manager fails and favors the occurrence of #1.
3	Domain manager and *FullStatus* fault-tolerant agent	Mailbox.msg or Intercom.msg	The queue fills because the *FullStatus* fault-tolerant agent cannot cope with flow.	The *FullStatus* fault-tolerant agent fails and favors the occurrence of #2.

Table 55. Critical flow errors (continued)

Flow no.	Location	Queue	Risk	Impact
4	Domain manager	tomaster.msg	The queue fills because of too many unlinked workstations in the domain.	The domain manager starts to unlink the subtree and accumulates messages in the structure.
5	Fault-tolerant agents - only when `enSwfaultTol` global option is set to *yes*	deadletter.msg	The queue fills because of too many unlinked workstations in the domain.	The agent stops.
6	Fault-tolerant agents - only when `enSwfaultTol` global option is set to *yes*	ftbox.msg	This queue is circular. The rate of messages entering the queue exceeds the rate of messages being processed, because of too many unlinked workstations in the domain.	Events are lost.

Note:

1. Flows are greater at the master domain manager and at any *FullStatus* fault-tolerant agents in the master domain than at subordinate domain managers or *FullStatus* fault-tolerant agents.
2. Use evtsize -show to monitor queue sizes.
3. The amount of update flow is related to the amount of workload running in a particular subtree and is unavoidable.
4. The amount of ad hoc flow is related to the amount of additional workload on any point of the network. It can be reduced by planning more workload even if it is inactive. Note that simple reruns (not `rerun from`) do not create an ad hoc flow.

The planning, alert, and recovery strategy must take into account the following points:

- Queue files are created with a fixed size and messages are added and removed in a cyclical fashion. A queue reaches capacity when the flow of incoming messages exceeds the outgoing flow for a sufficient length of time to use up the available space. For example, if messages are being added to a queue at a rate of 1MB per time unit and are being processed and removed at a rate of 0.5 MB per time unit, a queue sized at 10 MB (the default) is at capacity after 20 time units. But if the inward flow rate descends to be the same as the outward flow rate after 19 time units, the queue does not reach capacity.
- The risk of the domain manager failing can be mitigated by switching to the backup domain manager. In this case, the contents of the queues on the domain manager are unavailable until the domain manager backup is started. In

all cases, the size of the queue on the upper domain manager towards any other domain manager must respect the condition A, as indicated in the table Table 56: Queue sizing conditions. on page 296.

- The risk that fault-switching fault-tolerant agents might not be able to cope with the flow must be planned beforehand. The specifications for fault-switching fault-tolerant agents must be similar to those of the domain manager, to avoid that an agent receives a load that is not appropriate to its capacity. Check if a queue is forming at the *FullStatus* fault-tolerant agents, both in ordinary and peak operation situations.
- Once risk #2 has been dealt with, the possibility of a network link failure can be mitigated by sizing the queue from a domain manager to the *FullStatus* fault-tolerant agents appropriately as a function of the average network outage duration, and by increasing the size of the mailbox in case of unexpected long outage (see condition B of Table 56: Queue sizing conditions. on page 296).
- The same condition applies for avoiding an overflow of the domain manager's tomaster.msg queue with respect to network outages (see condition C) of Table 56: Queue sizing conditions. on page 296.

Table 56. Queue sizing conditions.

A	MaxAlertTime <= size(UpperDM#queueToDM) / averageAdhocFlow
B	MaxNetOutage <= size(DM#queueToFSFTA) / (averageAdhocFlow + averageUpdateFlow)
C	MaxNetOutage <= size(DM#queueToUpperDM) / (averageUpdateFlow)

Monitoring the HCL Workload Automation message queues

You can use event-driven workload automation (EDWA) to monitor the size of message queues and to start a predefined set of actions when one or more specific events take place. For more information about event-driven workload automation, refer to *HCL Workload Automation: User's Guide and Reference*.

You can monitor the following message queues:

- appserverbox
- mailbox
- clbox
- intercom
- courier
- monbox
- moncmd
- server
- tomaster
- pobox
- planbox

The following .XML file contains the definition of a sample event rule to monitor the mailbox queue on the specified workstation and send an email when the filling percentage is greater than the specified value. If the condition described in the rule is already existing when you deploy the rule, the related event is not generated. This event rule calls the MailSender action provider to send an email to the receivers you specify. For more information about the MailSender action provider, refer to *HCL Workload Automation: User's Guide and Reference*:

```
<?xml version="1.0"?>
<eventRuleSet  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules
   http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules/EventRules.xsd">
 <eventRule name="MONITORQUEUE" ruleType="filter" isDraft="no">
  <eventCondition name="twsMesQueEvt1" eventProvider="TWSApplicationMonitor" eventType="TWSMessageQueues">
   <scope>
    MAILBOX FILLED UP 80% ON FTA
   </scope>
   <filteringPredicate>
    <attributeFilter name="MailboxName" operator="eq">
     <value>mailbox_name</value>
    </attributeFilter>
    <attributeFilter name="FillingPercentage" operator="ge">
     <value>filling_percentage</value>
    </attributeFilter>
    <attributeFilter name="Workstation" operator="eq">
     <value>workstation_name</value>
    </attributeFilter>
    <attributeFilter name="SampleInterval" operator="eq">
     <value>sample_interval</value>
    </attributeFilter>
   </filteringPredicate>
  </eventCondition>
  <action actionProvider="MailSender" actionType="SendMail" responseType="onDetection">
   <scope>
    TWSUSER@TWS : THE MAILBOX ON workstation_name...
   </scope>
   <parameter name="To">
    <value>main_receiver_list</value>
   </parameter>
   <parameter name="Subject">
    <value>mail_subject</value>
   </parameter>
  </action>
 </eventRule>
</eventRuleSet>
```

where:

mailbox_name

Is the name of the mailbox to monitor.

filling_percentage

Is the filling percentage. Supported operators are as follows:

ge

causes the event generation when the mailbox filling percentage increases over the threshold value. The event is generated only the first time the specified mailbox filling percentage is reached. If you restart the SSM agent and the filling percentage is higher than the threshold value, the event is generated again. Table 57: Example for the ge operator on page 298 provides an example in which the **ge** operator is set to 70%.

Table 57. Example for the ge operator

Mailbox name	Filling percentage	Action
Sample (0)	>= 70%	event not generated
Sample (0)	< 70%	event not generated
Sample (n-1)	< 70%	event not generated
Sample (n)	>= 70%	event generated
Sample	>= 70%	event not

Table 57. Example for the ge operator (continued)

Mailbox name	Filling percentage	Action
(n +1)		generated

le

causes the event generation when the mailbox filling percentage decreases under the threshold value. The event is generated only the first time the specified mailbox filling percentage is reached. If you restart the SSM agent and the filling percentage is lower than the threshold value, the event is not generated until the filling percentage increases over the threshold value and then decreases under it again. Table 58: Example for the le operator on page 299 provides an example in which the **le** operator is set to 50%:

Table 58. Example for the le operator

Mailbox name	Filling percentage	Action
Sample (0)	<= 50%	event not generated
Sample (0)	> 50%	event not generated

Table 58. Example for the le operator (continued)

Mailbox name	Filling percentage	Action
Sample (n-1)	> 50%	event not generated
Sample (n)	<= 50%	event generated
Sample (n+1)	<= 50%	event not generated

workstation_name

Is the workstation on which the event is generated.

sample_interval

Is the interval, expressed in seconds, for monitoring the mailbox filling percentage.

main_receiver_list

Is the main receiver list.

mail_subject

Is the subject of the mail.

Changing a queue size

Use the evtsize command to resize a queue.

When you have used evtsize to resize a queue, the queue remain at that size until the next time you use evtsize. It only reverts to the default size of 60 MB if you delete it, at which point HCL Workload Automation re-creates it with the default size.

evtsize

Defines the size of the HCL Workload Automation message files. This command is used by the HCL Workload Automation administrator either to increase the size of a message file after receiving the message, “End of file on events file.", or to monitor the size of the queue of messages contained in the message file.

Authorization

You must be **maestro** or **root** in UNIX™, or **Administrator** in Windows™ to run **evtsize**. Stop the HCL Workload Automation engine before running this command.

Syntax

evtsize -V | -U

evtsize *file_name size*

evtsize -compact *file_name* [*size*]

evtsize -show *file_name*

Arguments

-V

Displays the command version and exits.

-U

Displays command usage information and exits.

-compact *file_name* [*size*]

Reduces the size of the specified message file to the size occupied by the messages present at the time you run the command. You can optionally use this keyword to also specify a new file size.

-show *file_name*

Displays the size of the queue of messages contained in the message file

file_name

The name of the event file. Specify one of the following:

`Courier.msg`

`Intercom.msg`

`Mailbox.msg`

`PlanBox.msg`

`Server.msg`

`pobox/`*`workstation`*`.msg`

size

The maximum size of the event file in bytes. When first built by HCL Workload Automation, the maximum size is set to 10 MB.

Note: The size of the message file is equal to or bigger than the real size of the queue of messages it contains and it progressively increases until the queue of messages becomes empty; as this occurs the message file is emptied.

Example

Examples

To set the maximum size of the `Intercom.msg` file to 20 MB, run the following command:

```
evtsize Intercom.msg 20000000
```

To set the maximum size of the `pobox` file for workstation `chicago` to 15 MB, run the following command:

```
evtsize pobox\chicago.msg 15000000
```

The following command:

```
evtsize -show Intercom.msg
```

returns the following output:

```
HCL Workload Scheduler (UNIX)/EVTSIZE 9.4 (1.2.2.4) Licensed Materials -
Licensed Materials - Property of IBM* and HCL**
5698-WSH
(C) Copyright IBM Corp. 1998, 2016 All rights reserved.
(C) Copyright HCL Technologies Ltd. 2016, 2024 All rights reserved
* Trademark of International Business Machines
** Trademark of HCL Technologies Limited
AWSDEK703I Queue size current 240, maximum 10000000 bytes (read 48, write 288)
```

where:

880

Is the size of the current queue of the `Intercom.msg` file

10000000

Is the maximum size of the `Intercom.msg` file

read 48

Is the pointer position to read records

write 928

Is the pointer position to write records

Tuning mailman servers

Once the distribution of agents to mailman servers has been established, all the groups of agents attached to the same server must respect the link condition.

The link condition relates the number of agents connected to a mailman process and the tuning parameters for unlink on the mailman and writer side.

No_agents(i)

The number of agents connected to a given mailman server *i*

Mm_unlink

A parameter set in the `localopts` of both domain manager and agent. Specifies the maximum number of seconds mailman waits before unlinking from a workstation that is not responding.

Wr_unlink

A parameter set in the `localopts` of both domain manager and agent. Specifies the number of seconds the writer process waits before exiting if no incoming messages are received.

Max_down_agents

The maximum probable number of agents that are unavailable without having the `ignore` flag set in the database and having the `autolink` flag on.

tcp timeout

A parameter set in the `localopts` of both domain manager and agent. Specify the maximum number of seconds that can be waited for the completion of a TCP/IP request on a connected workstation that is not responding.

The condition is:

```
Wr_unlink = Mm_unlink > 1.2 * Max_down_agents * tcp timeout
```

This condition expresses that if the time before unlink is smaller than the probable time of idle waiting of the mailman process (waiting connect timeout for each agent that is currently down) in its loop to reactivate the connections, the agents unlink constantly when some agents are down.

Netman configuration file

The netman configuration file exists on all HCL Workload Automation workstations to define the services provided by netman. It is called `<TWA_home>/TWS/network/Netconf`. The `NetConf` file includes comments describing each service. The services are:

2001

Start a writer process to handle incoming messages from a remote mailman.

2002

Start the mailman process. Mailman, in turn, starts the rest of the process tree (batchman, jobman).

2003

Stop the HCL Workload Automation process to handle incoming messages from a remote mailman.

2004

Find and return a stdlist file to the requesting Conman process.

2005

Switch the domain manager in a domain.

2006

Locally download scripts scheduled by an HCL Workload Automation for Z master domain manager.

2007

Required to bypass a firewall.

2008

Stop HCL Workload Automation workstations in a hierarchical fashion

2009

Runs the switchmgr script to stop and restart a manager in such a way that it does not open any links to other workstations until it receives the *switchmgr* event. Can only be used when the `enSwfaultTol` global option is set to *yes*.

2010

Starts mailman with the parameter *demgr*. It is used by the service *2009*. Can only be used when the `enSwfaultTol` global option is set to *yes*.

2011

Runs monman as a child process (`son bin/monman.exe`)

2012

Runs conman to stop the event monitoring engine (`command bin/conman.exe stopmon`).

2013

Runs conman to switch event processors (`command bin/conman.exe switchevtproc -this`)

2014

Runs conman to start event processing (`command bin/conman.exe startevtproc -this`)

2015

Runs conman to stop event processing (`command bin/conman.exe stopevtproc -this`)

2016

Runs conman to force the update of the monitoring configuration file for the event monitoring engine (`command bin/conman.exe deployconf`)

2017

Runs conman to stop event processing on a client (`client bin/conman.exe synchronizedcmd -stopevtproc`)

2018

Runs conman to check event processing on a client (`client bin/conman.exe synchronizedcmd -checkevtproc`)

2021

Runs conman to start appservman

2022

Runs conman to run the subcommand stopappserver that stops the application server

2023

Runs conman to run the subcommand startappserver that starts the application server

2501

Check the status of a remote job.

2502

Start the Console Manager – a service requested by the client side of the Remote Console. See the *IBM® Tivoli® Remote Control: User's Guide* for more information.

2503

Used by the connector to interact with r3batch extended agent.

Determining internal Symphony table size

The mailman service (2002) can optionally take a parameter that determines the initial size of the internal Symphony table. If you do not supply this parameter, mailman calculates the initial table size based on the number of records in the file.

Note: Mailman expands the table if it needs to, even if this parameter is not supplied.

In normal circumstances, leave mailman to take the default value in the `NetConf` file as supplied (32000). However, if you are experiencing problems with memory, you can allocate a table that is initially smaller. To do this you change the parameter to the service *2002* in the `NetConf` file. The syntax for the entry is:

```
2002     son     bin/mailman [ -parm number ]
```

where, *number* is used to calculate the initial Symphony table size based on the number of records in the Symphony file.

If *r* is the number of records in the Symphony file when batchman starts, Table 59: Calculation of internal Symphony table on page 306 shows how the size of the internal Symphony table is calculated, depending on the value of *number*:

Table 59. Calculation of internal Symphony table

Value of *number*	Table size
0	(4/3r) + 512

n	if n > r, n
	if n <= r, (4/3r) + 512

-1	65535

-n	if +n => r, n
	if +n < r, r + 512

If during the production period you add more jobs, the maximum internal Symphony table size is increased dynamically, up to the maximum number of records allowed in the Symphony file, which is 2,000,000,000.

Defining access methods for agents

Access methods are used to extend the job scheduling functions of HCL Workload Automation to other systems and applications. They run on:

Extended agents (applies only to HCL Workload Automation)

They are logical workstations related to an access method hosted by a physical HCL Workload Automation workstation (not another extended agent). More than one extended agent workstation can be hosted by the same HCL Workload Automation workstation and use the same access method. The extended agent runs on fault-tolerant agents defined using a standard HCL Workload Automation workstation definition, which gives the extended agent a name and identifies the access method. The access method is a program that is run by the hosting workstation whenever HCL Workload Automation submits a job to an external system.

Jobs are defined for an extended agent in the same manner as for other HCL Workload Automation workstations, except that job attributes are dictated by the external system or application.

Information about job running execution is sent to HCL Workload Automation from an extended agent using the job `stdlist` file. A method options file can specify alternate logins to launch jobs and check *opens* file dependencies. For more information, see the *User's Guide and Reference*.

A physical workstation can host a maximum of 255 extended agents.

dynamic agents and HCL Workload Automation agents (z-centric)

They communicate with external systems to start the job and return the status of the job. To run access methods on external applications using dynamic agents, you define a job of type **access method**.

Access methods are available on the following systems and applications.

- SAP R/3
- z/OS
- Custom methods
- unixssh
- unixrsh

The UNIX™ access methods included with HCL Workload Automation, are described in the related section in *Administration Guide*.

If you are working with dynamic agents, for information about defining HCL Workload Automation workstations, see the section that explains how to define workstations in the database in *User's Guide and Reference*. For information about writing access methods, see the section about the access method interface in *User's Guide and Reference*.

More information about access methods is found in *Scheduling Job Integrations with HCL Workload Automation*.

UNIX™ access methods

HCL Workload Automation includes two types of UNIX™ access methods, local UNIX access methods and remote UNIX access methods.

The Local UNIX™ access method runs on extended agents. Use the Local UNIX™ access method to enable a single UNIX™ workstation to operate as two HCL Workload Automation workstations, both of which you can run HCL Workload Automation scheduled jobs.

The Remote UNIX™ access method runs on extended agents and dynamic agents.

On extended agents

Use the Remote UNIX™ access method to designate a remote UNIX™ workstation to run HCL Workload Automation scheduled jobs without having HCL Workload Automation installed on it.

On dynamic agents

Define a job of type **xajob** that runs on dynamic agents. The dynamic agent communicates with the external system to start the job and return the status of the job.

Local UNIX™ access method running on fault-tolerant agents only

The Local UNIX™ method can be used to define multiple HCL Workload Automation workstations on one workstation: the host workstation and one or more extended agents. When HCL Workload Automation sends a job to a local UNIX™ extended agent, the access method, **unixlocl**, is invoked by the host to run the job. The method starts by running the standard configuration script on the host workstation (`<`*`TWA_home`*`>/TWS/jobmanrc`). If the logon user of the job is permitted to use a local configuration script and the script exists as `$HOME/TWS/.jobmanrc`, the local configuration script is also run. The job itself is then run either by the standard or the local configuration script. If neither configuration script exists, the method starts the job.

The launching of the configuration scripts, `jobmanrc` and `.jobmanrc` is configurable in the method script. The method runs the configuration scripts by default, if they exist. To disable this feature, you must comment out a set of lines in the method script. For more information, examine the script file `<`*`TWA_home`*`>/TWS/methods/unixlocl` on the extended agent's host.

Remote UNIX™ access method

The Remote UNIX™ access method can be used to designate a non-HCL Workload Automation workstation to run jobs scheduled by HCL Workload Automation. You can use `unixrsh` or `unixssh`:

The `unixrsh` access method

When HCL Workload Automation sends a job to a remote UNIX™ extended agent, the access method, `unixrsh`, creates a `/tmp/maestro` directory on the non-HCL Workload Automation workstation. It then transfers a wrapper script to the directory and runs it. The wrapper then runs the scheduled job. The wrapper is created only once, unless it is deleted, moved, or is outdated.

To run jobs using the `unixrsh` access method, the job logon users must be given appropriate access on the non-HCL Workload Automation UNIX™ workstation. To give appropriate access, a `.rhost`, `/etc/host.equiv`, or equivalent file must be set up on the workstation. On extended agents, if *opens* file dependencies are to be checked, *root* access must also be permitted. Contact your system administrator for help. For more information about the access method, examine the script file `TWA_home/TWS/methods/unixrsh` on an extended agent's host.

The `unixssh` access method

The `unixssh` access method works like `unixrsh` but uses a secure remote shell to connect to the remote host. The files used by this method are:

```
methods/unixssh
methods/unixssh.wrp
```

The `unixssh` method uses the *ssh* key. You can generate this keyword with any tools that are compatible with the secure remote shell.

Note: The passphrase must be blank.

The following scenario gives an example of how to set up the method:

You installed a HCL Workload Automation, fault-tolerant agent or dynamic agent with the *TWS_user*: `twsuser`. You want to run a remote shell in the remote host "REMOTE_HOST" with the user "guest". The procedure is as follows:

1. Create the public and private key for the user`twsuser`, The following is an example using rsa:
   a. Log on as `twsuser`
   b. Run

      ```
      ssh-keygen -t rsa
      ```

   c. When the tool asks for the passphrase, press Enter (leaving the passphrase blank.) The keys are saved as follows:

Key	Location	Comment
Public	*TWA_home*/TWS/.ssh/id_rsa.pub	
Private	*TWA_home*/TWS/.ssh/id_rsa	Do not send this file!

      Note: Different tools store the key in different places.

2. At the remote host, perform the following actions:
   a. Telnet to the remote host.
   b. Log on as "guest".
   c. Change to the `.ssh` directory in the user home directory, or create it if it does not exist (the directory permissions must be adequate: for example, 700 for the directory and 600 for its contents).
   d. Append the *public* key you created in step 1 to the `authorized_keys` file (create the file if it does not exist), using the command:

      ```
      cat id_rsa.pub >> authorized_keys
      ```

3. At the fault-tolerant agent or dynamic agent, make the remote host "known" before attempting to let HCL Workload Automation processes use the connection. This action can be achieved in one of two ways:
   - Log on as `twsuser` and connect to the host using the command:

     ```
     ssh -l guest remote_host_name ls
     ```

     A prompt is displayed saying that the host is not known, and asking permission to access it. Give permission, and the host is added to the list of known hosts.
   - Alternatively, use the ssh documentation to add the remote host to the file of known hosts.

Managing production for extended agents

In general, jobs that run on extended agents behave like other HCL Workload Automation jobs. HCL Workload Automation tracks a job's status and records output in the job's `stdlist` files. These files are stored on the extended agent's *host* workstation. For more information on managing jobs, see the section that describes HCL Workload Automation plan tasks in the *HCL Workload Automation: User's Guide and Reference*.

Failure launching jobs on extended agents and dynamic agents

If the access method is not in the proper directory on the extended agent's host, on the dynamic agent, or the method cannot be accessed by HCL Workload Automation, jobs fail to launch or a file dependency is not checked. For a job, the HCL Workload Automation jobs logon or the logon specified in the method options file must have read and execute permissions for the access method. When checking a file to satisfy an *opens* dependency, root is used as the login unless another login is specified in the method options file. For more information about method options, see the *HCL Workload Automation: User's Guide and Reference*.

IP address validation

When a TCP/IP connection is established, netman reads the requester's node name and IP address from the socket. The IP address and node name are used to search the `Symphony` file for a known HCL Workload Automation workstation with one of the following possible results:

- If an IP address match is found the validation is considered successful.
- If a node name match is found, the validation is considered successful.
- If no match is found in the `Symphony` file or the IP address returned does not match the one read from the socket, the validation is considered unsuccessful.

The local option, `nm ipvalidate`, determines the action to be taken if IP validation is unsuccessful. If the option is set to `full`, unsuccessful validation causes HCL Workload Automation to close the connection and generate an error message. If the option is set to `none` (default), HCL Workload Automation permits all connections, but generates a warning message for unsuccessful validation checks.

Support for Internet Protocol version 6

HCL Workload Automation supports Internet Protocol version 6 (IPv6) in addition to the legacy IPv4. To assist customers in staging the transition from an IPv4 environment to a complete IPv6 environment, HCL Workload Automation provides IP dual-stack support. In other terms, the product is designed to communicate using both IPv4 and IPv6 protocols simultaneously with other applications using IPv4 or IPv6.

To this end, the IPv4-specific `gethostbyname` and `gethostbyaddr` functions have been replaced by the new `getaddrinfo` API that makes the client-server mechanism entirely protocol independent.

The `getaddrinfo` function handles both name-to-address and service-to-port translation, and returns `sockaddr` structures instead of a list of addresses These `sockaddr` structures can then be used by the socket functions directly. In this way, `getaddrinfo` hides all the protocol dependencies in the library function, which is where they belong. The application deals only with the socket address structures that are filled in by `getaddrinfo`.

Operating system configuration (UNIX™ only)

IP validation depends on the system call `getaddrinfo()` to look up all the valid addresses for a host. The behavior of this routine varies, depending on the system configuration. When `getaddrinfo()` uses the file `/etc/hosts`, it returns the first matching entry. If the connection is initiated on an address which appears after the first matching entry, IP validation fails. To resolve the problem, place the entry used to initiate the connection before any other matching entries in the `/etc/hosts` file. If `getaddrinfo()` uses the "named" name server or the Network Information Service server and `getaddrinfo()` fails, contact your system administrator for assistance.

IP address validation messages

Following is a list of the messages for IP validation. If the Local Option `nm ipvalidate` is set to `none` (default), the errors appear as warnings.

See the end of the list of conditions for the key to the variables:

- HCL Workload Automation workstation name is not found in the Symphony file

```
Ip address validation failed for request:
 Service <num> for <program> on <workstation>(<operating_system_type>).
Connection received from IP address:
 <c_ipaddr>. MAESTRO CPU <workstation> not found in
Symphony file.
```

- Call to `getaddrinfo()` fails:

```
IP address validation failed for request:
 Service num for <program> on cpu(<operating_system_type>).
Connection received from IP address:
 <c_ipaddr>. getaddrinfo() failed, unable to
 retrieve IP address of connecting node: <node>.
```

- IP Addresses returned by `getaddrinfo()` do not match the IP address of connection workstation:

```
IP address validation failed for request:
 Service <num> for <program> on <workstation>(<operating_system_type>).
Connection received from IP address:
 <c_ipaddr>. System known IP addresses for node
 name node: <k_ipaddr>.
```

- The IP address specified in the workstation definition for the HCL Workload Automation workstation indicated in the service request packet does not match the IP address of connecting workstation:

```
IP address validation failed for request:
 Service <num> for <program> on <workstation>(<operating_system_type>).
Connection received from IP address:
 <c_ipaddr>. TWS known IP addresses for cpu
 <k_ipaddr>.
```

- Regardless of the state of `nm ipvalidate`, the following information message is displayed when IP validation cannot be performed because the `Symphony` file does not exist or an error occurs when reading it:

```
IP address validation not performed for
 request: Service <num> for <program> on
 <workstation>(<operating_system_type>). Connection received from IP
 address: <c_ipaddr>. Cannot open or read
Symphony file. Service request accepted.
```

Where:

<num>

Service number (2001-**writer**, 2002-**mailman**...)

<program>

Program requesting service

<workstation>

HCL Workload Automation workstation name of connecting workstation

<operating_system_type>

Operating system of connecting workstation

<node>

Node name or IP address of connecting workstation

<c_ipaddr>

IP address of connecting workstation

<k_ipaddr>

Known IP address for connecting workstation

IP validation is always successful in the absence of a `Symphony` file. In communications from a domain manager to an agent it is normally successful because a `Symphony` file does not yet exist. However, if the agent has a `Symphony` file from a previous run, the initial link request might fail if the `Symphony` file does not include the name of the domain manager.

Impact of network changes

Any changes that you make to your network might have an impact on HCL Workload Automation. Workstations can be identified within HCL Workload Automation by host name or IP address. Any changes to host names or IP addresses of specific workstations must obviously be also implemented in the HCL Workload Automation database. However, remember that if those workstations are involved in jobs that are currently scheduled in the Symphony file, those jobs are looking for the old workstation identity.

Changes to host names or IP addresses of specific workstations can be activated immediately by running JnextPlan -for 0000. A new production plan is created (containing the updated IP addresses and host names), but the plan time span is not extended.

Thus, plan any network changes with the job schedules in mind, and for major changes you are advised to suspend HCL Workload Automation activities until the changes complete in the network and also implemented in the HCL Workload Automation database.

Network changes also have a specific impact on the connection parameters used by the application server and the command-line client:

Application server

If you change the network you will need to change the communication parameters specified in the application server configuration files. For more information, see Configuring HCL Workload Automation using templates on page 422.

Command-line client

When you connect from the command-line client you supply a set of connection parameters. This is done in one of these ways:

From the localopts file

The default method is that the connection parameters in the `localopts` file are customized when the command line client is installed.

From the useropts file

A `useropts` file might have been created for the user in question, containing a version of the connection parameters personalized for the user.

In the command line, individually

When you invoke one of the command-line programs, you can optionally include the parameters as arguments to the command. These override the values in the `localopts` or `useropts` files.

In the command line, in a file

When you invoke one of the command-line programs, you can optionally include the parameters in a file, the name of which is identified as the -file argument to the command. These override the values in the `localopts` or `useropts` files.

Modify whichever method you are using to incorporate the new network connection details.

Chapter 6. Configuring secure communications

HCL Workload Automation ensures your business operations are always secure with robust, authenticated, and encrypted connections powered by the Secure Sockets Layer (SSL) protocol.

By leveraging certificates, only trusted devices, users, and systems can access the environment resources, providing the ultimate protection for your critical workloads.

In the following chapters, you can find all the information you need to easily manage certificates and maintain secure communication across your entire infrastructure:

- Managing certificates using Certman on page 314
- Certificate rotation on page 325
- Scenario: SSL Communication across the fault-tolerant agent network on page 327

Managing certificates using Certman

Using Certman tool, you can easily manage the custom certificates in both fresh installation or upgrade procedures.

Certman supports the following actions:

- Generate new certificates and a Certificate Authority (CA) on page 314
- Generate new certificates from an existing Certificate Authority (CA) on page 316
- Extract certificates from an existing keystore on a master domain manager on page 317
- Verify the validity of certificates on page 321
- Import certificates from a master domain manager into the Dynamic Workload Console on page 322
- Remove an alias from the keystore or truststore on page 323

You can find Certman at the following path: `TWS_INST_DIR/TWS/bin`, where `TWS_INST_DIR` is the HCL Workload Automation installation directory.

To verify the version of Certman you are using, you can run the following command: `certman version`.

Running any command of Certman, you can also see where the logs file is located.

Generate new certificates and a Certificate Authority (CA)

About this task

If you want to secure the communication based on the Secure Sockets Layer (SSL) protocol, but you do not have a corporate Certificate Authority (CA), you can use Certman to create one and generate the required certificates.

1. Browse to the following path: `<image_location>/TWS/<interp_name>/Tivoli_LWA_<interp_name>\TWS \bin`
2. Generate the CA and certificates by running the following command:

```
certman generate -keypasswd <pwd> -outpath <output path> [-capath <ca path>] [-wauser <user>] [-wagroup
 <group>]
```

Where:

keypasswd

Specify the password to encrypt the private key.

outpath

Specify the folder where generate the certificates.

capath

Leave empty to generate a `ca.crt` and `ca.key`.

wauser

Optionally, specify the TWS_user that must be set as owner of the output files.

wagroup

Optionally, specify the TWS_user that must be set as group of the output files..

Note: To specify an owner and group in **wauser** and **wagroup** parameters, the user who launches Certman must have the permissions to change the owner and group on output files.

Results

The following output files are the CA and certificates you can find in the specified output folder:

- **`ca.crt`**

  The file that contains the `Root ca`.

- **`ca.key`**

  The private key of the CA.

- **`tls.crt`**

  The certificate signed and validated by the CA.

- **`tls.key`**

  The private key of the tls certificate.

- **`tls.sth`**

  The stash file of the tls certificate that contains the password encoded in Base64 format.

Note: It is strongly suggested that you save the `ca.key` so that in future, if needed, you can generate or replace the certificates only.

After having generated the CA, add it to the OS and browser so that they can trust the new CA.

Generate new certificates from an existing Certificate Authority (CA)

About this task

If you need to generate new certificates from an existing Certificate Authority (CA), you can use Certman to generate the required certificates. In this case, the **capath** parameter is required and you need to provide the path of the existing `ca.crt` and `ca.key` files.

1. Browse to the following path: `<image_location>/TWS/<interp_name>/Tivoli_LWA_<interp_name>\TWS\bin`
2. Generate the new certificates by running the following command:

   ```
   certman generate -keypasswd <pwd> -outpath <output path> [-capath <ca path>] [-wauser <user>] [-wagroup
    <group>]
   ```

   Where:

   keypasswd

   Specify the password to encrypt the private key.

   outpath

   Specify the folder where generate the certificates.

   capath

   Specify the path where `ca.crt` and `ca.key` files are stored.

   wauser

   Optionally, specify the TWS_user that must be set as owner of the output files.

   wagroup

   Optionally, specify the TWS_user that must be set as group of the output files..

   Note: To specify an owner and group in **wauser** and **wagroup** parameters, the user who launches Certman must have the permissions to change the owner and group on output files.

Results

The following output files are the certificates you can find in the specified output folder:

- **`tls.crt`**

  The certificate signed and validated by the CA.

- **`tls.key`**

  The private key of the tls certificate.

- **`tls.sth`**

  The stash file of the tls certificate that contains the password encoded in Base64 format.

Extract certificates from the keystore and truststore

About this task

You can use Certman to extract certificates from the keystore and truststore on a master domain manager, an agent, or the Dynamic Workload Console to provide them to the backup master domain manager or Dynamic Workload Console.

Extract certificates from version 10.2.3 or later

About this task

You can extract certificates from the keystore and truststore on a master domain manager, an agent, or the Dynamic Workload Console V10.2.3 or later by completing the following steps:

1. Browse to one of the following installation bin paths, according to the component from which you want to extract the certificate:

   Master domain manager

   `<MDM_INST_PATH>/TWS/bin/certman`, where `<MDM_INST_PATH>` is the master domain manager installation directory.

   Dynamic Workload Console

   `<DWC_INST_PATH>/bin/certman`, where `<DWC_INST_PATH>` is the Dynamic Workload Console installation directory.

   Agent

   `<AGENT_INST_PATH>/TWS/bin/certman`, where `<AGENT_INST_PATH>` is the agent installation directory.

2. Extract the certificates by running the following command:

   ```
   certman extract -outpath <output path> [-storepasswd <pw>] [-agentscope] [-wauser <user>] [-wagroup
    <group>] [-workdir <working directory>] [-cachain-splitted]
   ```

   Where:

   outpath

   Specify the folder where to store the certificates.

   storepasswd

   Optionally, specify the password of the keystore on the master domain manager.

   Note: For version 9.4.x, this parameter is required.

   agentscope

   Optionally, specify that the action performed by the command applies to the keystore of an agent.

Note: To target the keystore of a master domain manager, omit the `agentscope` option and run the command separately.

wauser

Optionally, specify the TWS_user that must be set as owner of the output files.

wagroup

Optionally, specify the TWS_user that must be set as group of the output files..

Note: To specify an owner and group in **wauser** and **wagroup** parameters, the user who launches Certman must have the permissions to change the owner and group on output files.

workdir

Optionally, specify the working directory used by the command for storing data while running. When the command stops running, the working directory is deleted. Ensure you have write access to the specified directory and enough space is available.

cachain-splitted

Optionally, specify the CA chain to be splitted into multiple files. By default, it is false.

Results

The following output files are the certificates you can find in the specified output folder:

- **`ca.crt`**

  The file that contains the intermediate CA certificate and ends up with the `Root ca`.

  Note: If you enabled the **cachain-splitted** parameter, the `ca.crt` contains only the `Root ca`. The intermediate CA certificates are stored in the additionalCAs subfolder.

- **`tls.crt`**

  The certificate signed and validated by the CA.

- **`tls.key`**

  The private key of the tls certificate.

- **`tls.sth`**

  The stash file of the tls certificate that contains the password encoded in Base64 format.

- **`additionalCAs`**

  The subfolder where any intermediate CA certificate extracted by the truststore is stored.

Extract certificates from a previous product version level

About this task

You can extract certificates from a previous product version level by completing the following steps:

1. From Flexnet or from HCL Software, download the 10.2.5 installation package: *HWA_10.2.4_<component>_<operating_system>.zip*
2. Extract the content, browse to the path `<IMAGE_DIR>/TWS/<OPERATING_SYSTEM>_<ARCHITECTURE>/Tivoli_LWA_<operating_system>/TWS/bin/` , and copy the following files:
   - certman
   - certman.extract.json
   - certman.generate.json
   - certman.import.json
   - certman.verify.json
   - certman.version.json
3. Paste the Certman files into the following path: `TWS_INST_DIR/TWS/bin`, where `TWS_INST_DIR` is the HCL Workload Automation installation directory.

   Note: For UNIX systems, ensure that all the files have the ownership of the user who installed the master domain manager and the correct permissions (775 for certman and 644 for the json files).

4. Extract the certificates by running the following command:

```
certman extract -outpath <output path> [-storepasswd <pw>] [-agentscope] [-wauser <user>] [-wagroup
 <group>] [-workdir <working directory>] [-cachain-splitted]
```

   Where:

   outpath

   Specify the folder where to store the certificates.

   storepasswd

   Optionally, specify the password of the keystore on the master domain manager.

   Note: For version 9.4.x, this parameter is required.

   agentscope

   Optionally, specify that the action performed by the command applies to the keystore of an agent.

> **Note:** To target the keystore of a master domain manager, omit the `agentscope` option and run the command separately.

wauser

Optionally, specify the TWS_user that must be set as owner of the output files.

wagroup

Optionally, specify the TWS_user that must be set as group of the output files..

> **Note:** To specify an owner and group in **wauser** and **wagroup** parameters, the user who launches Certman must have the permissions to change the owner and group on output files.

workdir

Optionally, specify the working directory used by the command for storing data while running. When the command stops running, the working directory is deleted. Ensure you have write access to the specified directory and enough space is available.

cachain-splitted

Optionally, specify the CA chain to be splitted into multiple files. By default, it is false.

Results

The following output files are the certificates you can find in the specified output folder:

- **`ca.crt`**

  The file that contains the intermediate CA certificate and ends up with the `Root ca`.

  > **Note:** If you enabled the **cachain-splitted** parameter, the `ca.crt` contains only the `Root ca`. The intermediate CA certificates are stored in the additionalCAs subfolder.

- **`tls.crt`**

  The certificate signed and validated by the CA.

- **`tls.key`**

  The private key of the tls certificate.

- **`tls.sth`**

  The stash file of the tls certificate that contains the password encoded in Base64 format.

- **`additionalCAs`**

  The subfolder where any intermediate CA certificate extracted by the truststore is stored.

Verify the validity of certificates

About this task

If you want to verify whether certificates are in a valid **.pem** format, are not expired, or have a stash that matches the private key password, you can use Certman to verify the validity.

1. Browse to the following path: `<image_location>/TWS/<interp_name>/Tivoli_LWA_<interp_name>\TWS\bin`
2. Check the validity by running the following command:

   ```
   certman verify -inpath <input path> -keypasswd <key pwd> [-minkeysize <minimum key size>] [-workdir
    <working directory>]
   ```

   Where:

   inpath

   Specify the folder that contains the following certificates:

   - `tls.crt`
   - `tls.key`
   - `tls.sth`
   - `ca.crt`

   keypasswd

   Specify the password used to encrypt the private key.

   minkeysize

   Optionally, specify the minimum size of the key. The default value is 1024.

   workdir

   Optionally, specify the working directory used by the command for storing data while running. When the command stops running, the working directory is deleted. Ensure you have write access to the specified directory and enough space is available.

Results

Using the verify command, the following checks are performed:

- The key password.
- The stash password (if the `tls.sth` file is available).
- The certificate expiration date.
- The key length.
- The certificate and key in **.pem** format.
- The private key and public key match.
- The public key in **tls.key** and public key in **tls.crt** match.
- The correctness of the **tls.crt** in the CA chain.
- The purposes of the **tls.crt** file, which must be suitable for both client and server connections.

Import certificates into the trustore

About this task

You can use Certman to import a CA chain into the truststore of the Dynamic Workload Console, of a master domain manager or of an agent.

If certificates being imported are part of a chain consisting of 3 or more certificates (one Root CA, followed by one or more Intermediate CAs, followed by the end user certificate), then the `ca.crt` must contain the `Root ca` certificate only. Any Intermediate CA certificates must be stored in the additionalCAs subfolder, which therefore becomes a mandatory subfolder. Each Intermediate CA must be stored in the additionalCAs subfolder in its own file.

Note: From V10.2.3, if certificates being imported are part of a chain, the ca.crt can contain also the intermediate CAs. In this case, it must begin with one or more intermediate CA certificates and end with the `Root ca`.

1. Browse to one of the following installation bin paths, according to the component on which you want to import the CA chain:

   Master domain manager

   `<MDM_INST_PATH>/TWS/bin/certman`, where `<MDM_INST_PATH>` is the master domain manager installation directory.

   Dynamic Workload Console

   `<DWC_INST_PATH>/bin/certman`, where `<DWC_INST_PATH>` is the Dynamic Workload Console installation directory.

   Agent

   `<AGENT_INST_PATH>/TWS/bin/certman`, where `<AGENT_INST_PATH>` is the agent installation directory.

2. Import the CA chain by running the following command:

   ```
   certman import (-inpath <input path> [-storepasswd <store pwd>][-all -keypasswd <key pwd>]|-url
    <host:port> -storepasswd <store pwd>) -alias <alias> [-forcealias] [-agentscope] [-updatedepot]
    [-workdir <working directory>]
   ```

   Where:

   inpath

   Specify the folder that contains the CA chain.

   storepasswd

   Optionally, specify the password of the trustore on the master domain manager.

   all

   Optionally, import the certificate, the key and the CA chain.

keypasswd

Specify the password used to encrypt the private key. If `all` is specified, this value is mandatory.

url

The URL of a server that contains the CA chain to be imported (for example, the master domain manager server).

Where

host

The fully qualified host name or IP address of the server.

port

The HTTPS port.

alias

Specify an alias to be used in the truststore file during the import.

forcealias

Optionally, specify an alias to be used in the trustore file that overwrites the existing alias. Use this parameter if the master domain manager already communicates with the Dynamic Workload Console.

agentscope

Optionally, specify that the action performed by the command applies to the truststore of an agent.

Note: To target the trustore of a master domain manager, omit the `agentscope` option and run the command separately.

updatedepot

Optionally, update the master domain manager `depot` folder located at: `<TWSDATA>/ssl/depot`

workdir

Optionally, specify the working directory used by the command for storing data while running. When the command stops running, the working directory is deleted. Ensure you have write access to the specified directory and enough space is available.

Results

The CA chain has been imported in the Dynamic Workload Console, master domain manager or agent.

Remove an alias from the keystore or truststore

About this task

You can remove an alias from the keystore or truststore on a master domain manager, an agent, or the Dynamic Workload Console by completing the following steps:

1. Browse to one of the following installation bin paths, according to the component from which you want to delete the alias:

   Master domain manager

   `<MDM_INST_PATH>/TWS/bin/certman`, where `<MDM_INST_PATH>` is the master domain manager installation directory.

   Dynamic Workload Console

   `<DWC_INST_PATH>/bin/certman`, where `<DWC_INST_PATH>` is the Dynamic Workload Console installation directory.

   Agent

   `<AGENT_INST_PATH>/TWS/bin/certman`, where `<AGENT_INST_PATH>` is the agent installation directory.

2. Remove the alias by running the following command:

   ```
   certman delete -alias <alias> -storetype <KEY|TRUST|ALL> [-storepasswd <store pwd>] [-agentscope]
    [-workdir <work dir>]
   ```

   Where:

   alias

   Specify the alias to remove.

   storetype

   Specify `KEY` to remove the alias from the keystore, `TRUST` to remove the alias from the truststore, or `ALL` to remove the alias from both the keystore and trustore.

   Note: Specifying `ALL` without the alias being actually present on both the keystore and the trustore, leads to the failure of the `certman delete` command.

   storepasswd

   Optionally, specify the password of the keystore or trustore on the master domain manager.

agentscope

Optionally, specify that the action performed by the command applies to the keystore or truststore of an agent.

Note: To target the keystore or trustore of a master domain manager, omit the `agentscope` option and run the command separately.

workdir

Optionally, specify the working directory used by the command for storing data while running. When the command stops running, the working directory is deleted. Ensure you have write access to the specified directory and enough space is available.

Results

Using the delete command, the specified alias is removed.

Certificate rotation

Procedure to rotate certificates in your environment

About this task

The procedure explained below is one of several procedures you can perform to achieve the same results and is intended only as an example. In this procedure, it is assumed your certificates have been signed by a Certificate Authority (CA) you created for this purpose. For more information about using an external CA or manually modifying all the keystores and key databases, see Replacing Default SSL Certificates with CA Signed Custom Certificates.

You can extend the certificate customization also to backup master domain manager, dynamic domain manager, and Dynamic Workload Console.

To customize the certificates for communication between master domain manager and dynamic agent, perform the following steps:

1. On the master domain manager, generate a self-signed certificate or issue a certificate sign request to a CA and import the certificate into `TWSServerKeyFile.p12`. For example, you can generate the private key to be used for signing the custom certificate by issuing the following command:

   ```
   openssl genrsa -des3 -out tls.key <key_size>
   ```

   where <key_size> must be equal to or major than 2048.
2. Create the certificate sign request:

   ```
   openssl req -new -key tls.key -out tls.csr -config <TWS_DATA_DIR>/ssl/openssl.cnf
   ```
3. Send the `.csr` to the CA:

   ```
   openssl x509 -req -in tls.csr -days 3650
   -CA ca.crt -CAkey ca.key -CAcreateserial -out tls.crt
   ```

4. After receiving back the signed certificate, you can import the custom certificate along with its private key into `TWSServerKeyFile.p12`, as follows:
    a. Create a single file containing both:

        ```
        cat tls.key tls.crt > tls.tot
        ```

    b. Export the `tls.tot` file to a PKCS12 keystore for the master domain manager:

        ```
        openssl pkcs12 -export -out TWSServerKeyFile.p12 -in tls.tot -name server
        ```

    c. Replace the existing `TWSServerKeyFile.p12` file located in *TWA_DATA_DIR*`/usr/servers/engineServer/resources/security` with the file you created in step 4b.
    d. Export the `tls.tot` file created in step 1 to a PKCS12 keystore for the dynamic agent:

        ```
        openssl pkcs12 -export -out TWSClientKeyStore.p12 -in tls.tot -name client
        ```

    e. Replace the existing `TWSClientKeyStore.p12` file located in *TWA_DATA_DIR*`/ssl/certs` with the file you created in step 4d.
5. On the master domain manager, import the CA certificate in the `TWSServerTrustFile.p12`:

    ```
    keytool -importcert -file ca.crt -keystore TWSServerTrustFile.p12
    -alias ca -trustcacerts
    ```

6. Export the certificates, as described below. Steps 6 on page 326 to 8 on page 326 ensure the handshake for APIKey authentication with JWT between master domain manager and agent completes correctly.

    ```
    keytool -exportcert -keystore
     $WA_DATADIR/usr/servers/engineServer/resources/security/TWSServerKeyFile.p12
    -storepass password -storetype pkcs12 -file /tmp/tls.crt -alias server -noprompt
    ```

7. Import the certificates, as follows:

    ```
    keytool -importcert -keystore
     $WA_DATADIR/usr/servers/engineServer/resources/security/TWSServerTrustFile.p12
    -storepass password -storetype pkcs12 -file /tmp/tls.crt -alias mpjwtkey -noprompt
    ```

8. Edit the value of the **mp.jwt.trust.key** variable from the **twstrustkey** to **mpjwtkey** in the `jwt_variables.xml` file located inside the WebSphere Application Server Liberty Base `overrides` folder. For more information about templates, see the topic about configuring HCL Workload Automation using templates in *Administration Guide*.

    If you do not remember what the public certificate alias is called, run the following command to retrieve the list of certificates within the keystore:

    ```
    keytool -list -keystore $WA_DATADIR/usr/servers/engineServer/resources/security/TWSServerKeyFile.p12
    -storepass password -storetype pkcs12
    ```

9. Replace the existing `TWSServerTrustFile.p12` located in *TWA_DATA_DIR*`/usr/servers/engineServer/resources/security` with the file you created in step 5.
10. On the master domain manager, edit the *TWA_DATA_DIR*`/broker/config/BrokerWorkstation.properties` file and update the list of authorized Common Names for the dynamic domain manager (broker). Append the Common Name used for the custom certificate to the `Broker.AuthorizedCNs` property:

    ```
    Broker.AuthorizedCNs=Server;ServerNew;new_CN
    ```

11. Run the AgentCertificateDownloader script on the dynamic agent. The script connects to the master domain manager, downloads the certificates in `.PEM` format (`tls.key, tls.crt, ca.crt` files), and deploys them to the agent. The certificates must be available on the master domain manager in a specific path. For more information, see the section about the AgentCertificateDownloader script in *HCL Workload Automation: Planning and Installation*.

Scenario: SSL Communication across the fault-tolerant agent network

SSL communication is enabled by default for fresh installations, however, you might need to set it up at upgrade time, as described in the following topics.

You can enable the SSL connection using OpenSSL Toolkit for the following components:

- Master domain manager and its domain managers
- Master domain manager and fault-tolerant agents in the master domain
- Master domain manager and backup master domain manager
- Domain manager and fault-tolerant agents that belong to that domain

The default certificates and the custom certificates are located in the `<`*`TWSDATA`*`>\ssl\certs` directory.

Using SSL for netman and conman

HCL Workload Automation provides a secure, authenticated, and encrypted connection mechanism for communication across the network topology. This mechanism is based on the Secure Sockets Layer (SSL) protocol and uses the OpenSSL Toolkit, which is automatically installed with HCL Workload Automation.

The SSL protocol is based on a private and public key methodology. SSL provides the following authentication methods:

CA trusting only

Two workstations trust each other if each receives from the other a certificate that is signed or is trusted. That is, if the CA certificate is in the list of trusted CAs on each workstation. With this authentication level, a workstation does not perform any additional checks on certificate content, such as the distinguished name. Any signed or trusted certificate can be used to establish an SSL session. See Setting local options on page 48 for a definition of the caonly option used by the **`ssl auth mode`** keyword.

Check if the distinguished name matches a defined string

Two workstations trust each other if, after receiving a trusted or signed certificate, each performs a further check by extracting the distinguished name from the certificate and comparing it with a string that was defined in its local options file. See Setting local options on page 48 for a definition of the string option.

Check if the distinguished name matches the workstation name

Two workstations trust each other if, after receiving a signed or trusted certificate, each performs a further check by extracting the distinguished name from the certificate and comparing it with the unique ID of the workstation that sent the certificate. You can obtain the unique ID by using the ;showid composer filter. Only if the unique ID is empty, you can use the name of the workstation instead of the unique ID.

See Setting local options on page 48 for a definition of the cpu option.

To provide SSL security for a domain manager attached to z/OS® in an end-to-end connection, configure the OS/390® Cryptographic Services System SSL in the HCL Workload Automation code that runs in the OS/390® USS UNIX® shell in the HCL Workload Automation for Z server address space. See the HCL Workload Automation for Z documentation.

When configuring SSL you can:

Use the same certificate for the entire network

If the workstations are configured with CA trusting only, they accept connections with any other workstation that sends a signed or trusted certificate. To enforce the authentication you define a name or a list of names that must match the contents of the certificate distinguished name (DN) field in the `localopts` file of each workstation.

Use a certificate for each domain

Install private keys and signed certificates for each domain in the network. Then, configure each workstation to accept a connection only with partners that have a particular string of the certificate DN field in the `localopts` file of each workstation.

Use a certificate for each workstation

Install a different key and a signed certificate on each workstation and add a Trusted CA list containing the CA that signed the certificate. Then, configure each workstation to accept a connection only with partners that have their workstation name specified in the `Symphony` file recorded in the DN field of the certificate.

Configuring SSL attributes

Use the composer command line or the Dynamic Workload Console to update the workstation definition in the database. See the section about workstation definition in *User's Guide and Reference* for further information.

Configure the following attributes:

secureaddr

Defines the port used to listen for incoming SSL connections. This value is read when the **securitylevel** attribute is set.

For workload broker workstations

Ignore this attribute.

For remote engine workstations using HTTPS protocol to communicate with the remote engine

Specify the HTTPS port number of the remote engine.

For other types of workstations

Specify the value assigned in the `localopts` file for variable *nm ssl port*. The value must be different value from the value assigned to *nm port* variable in the `localopts` file.

If **securitylevel** is specified, but this attribute is missing, the default value for this field is `31113`. Specify a value in the range from `1` to `65535`.

See the *Administration Guide* for information about SSL authentication and local options set in the `TWS_home/localopts` configuration file.

securitylevel

Specifies the type of SSL authentication for the workstation. Do not specify this attribute for a workstation with type `broker`. It can have one of the following values:

enabled

The workstation uses SSL authentication only if its domain manager workstation or another fault-tolerant agent below it in the domain hierarchy requires it.

on

The workstation uses SSL authentication when it connects with its domain manager. The domain manager uses SSL authentication when it connects to its parent domain manager. The fault-tolerant agent refuses any incoming connection from its domain manager if it is not an SSL connection.

force

The workstation uses SSL authentication for all of its connections and accepts connections from both parent and subordinate domain managers.

force_enabled

The workstation uses SSL authentication for all of its connections to all target workstations which are set to this value. The workstation tries to establish a connection in FULLSSL mode and, if the attempt fails, it tries to establish an unsecure connection. If you plan to set different security levels between master domain manager and fault-tolerant agents, ensure all these components are at version 95 Fix Pack 4 or later. For versions earlier than 95 Fix Pack 4, the same security level is required for master domain manager and fault-tolerant agents.

If this attribute is omitted, the workstation is not configured for SSL connections and any value for **secureaddr** is ignored. Make sure, in this case, that the *nm ssl port* local option is set to 0 to ensure that **netman** process does not try to open the port specified in **secureaddr**. See the *Administration Guide* for information about SSL authentication and local options set in the `TWS_home/localopts` configuration file.

The following table describes the type of communication used for each type of **securitylevel** setting.

Table 60. Type of communication depending on the security level value

Value set on the Fault-tolerant Agent (or the Domain Manager)	Value set on its Domain Manager (or on its Parent Domain Manager)	Type of connection established
Not specified	Not specified	TCP/IP
Enabled	Not specified	TCP/IP
On	Not specified	No connection
Force	Not specified	No connection

Table 60. Type of communication depending on the security level value (continued)

Value set on the Fault-tolerant Agent (or the Domain Manager)	Value set on its Domain Manager (or on its Parent Domain Manager)	Type of connection established
Not specified	On	TCP/IP
Enabled	On	TCP/IP
On	On	SSL
Force	On	SSL
Not specified	Enabled	TCP/IP
Enabled	Enabled	TCP/IP
On	Enabled	SSL
Force	Enabled	SSL
Not specified	Force	No connection
Enabled	Force	SSL
On	Force	SSL
Force	Force	SSL
force_enabled	force_enabled	SSL

The value for **securitylevel** is not specified for dynamic workload broker workstations.

The following example shows a workstation definition that includes the security attributes:

```
cpuname MYWIN
os WNT
node apollo
tcpaddr 30112
secureaddr 32222
for maestro
autolink off
fullstatus on
securitylevel on
end
```

Configuring full SSL security

This section describes how to implement full SSL security when using an SSL connection for communication across the network by netman and conman. It contains the following topics:

- Overview on page 331
- Setting up full SSL security on page 331
- Configuring full SSL support for internetwork dependencies on page 332

Note: The full SSL security feature is not applicable to the communication between dynamic agents and the broker workstation that is defined for the master domain manager or the dynamic domain manager to which the dynamic agents are connected.

Overview

This feature provides the option to set a higher degree of SSL-based connection security on HCL Workload Automation networks in addition to the already available level of SSL security.

If you require a more complete degree of SSL protection, this feature supplies new configuration options to setup advanced connection security, otherwise you can use the standard settings documented above in this chapter.

The Full SSL security enhancements

Full SSL security support provides the following enhancements:

- TCP ports that can become security breaches are no longer left open.
- Traveling data, including communication headers and trailers, is now *totally* encrypted.

Compatibility between SSL support levels

The non-full and the full SSL support levels are mutually exclusive. That is, they cannot be configured simultaneously and cannot be enabled at the same time. If you enable full SSL support for an HCL Workload Automation network, any connection attempts by agents that are not configured for full SSL will be rejected by agents with full SSL support enabled. Vice versa, agents configured for full SSL support cannot communicate with the rest of a network set up for non-full SSL support.

Setting up full SSL security

About this task

To set full SSL connection security for your network, you must configure the following options:

enSSLFullConnection (or `sf`)

Use `optman` on the master domain manager to set this global option to `Yes` to enable full SSL support for the network. For more information, see Setting global options on page 12.

nm SSL full port

If you defined the SSL port at installation time using the **netmanssl port** parameter, no further action is required. For more information about the **netmanssl port** parameter, see the section about agent and master installation parameters in *HCL Workload Automation: Planning and Installation*.

If you have not defined the SSL port at installation time, edit the `localopts` file on every agent of the network (including the master domain manager) to set this local option to the port number used to listen for incoming SSL connections. For more information, see Setting local options on page 48. Take note of the following:

- This port number is to be defined also for the `SECUREADDR` parameter in the workstation definition of the agent. For more information, see the topic about workstation definition in *User's Guide and Reference*.
- Check that the `securitylevel` parameter in the workstation definition of each workstation using SSL is set at least to *enabled*. For more information, see the topic about workstation definition in *User's Guide and Reference*.
- In a full SSL security setup, the `nm SSL port` local option is to be set to zero. For more information, see Setting local options on page 48.
- You must stop netman (**conman shut;wait**) and restart it (**StartUp**) after making the changes in `localopts`.

Configuring full SSL support for internetwork dependencies

About this task

The network agent that resolves internetwork dependencies requires a particular setup for full SSL support.

To enable a network agent for full SSL support:

1. Configure both the hosting and the remote fault-tolerant agents for full SSL support.
2. On the hosting fault-tolerant agent copy or move the `netmth.opts` file from the *`DATA_DIR`*`/config` to the *`DATA_DIR`*`/methods` directories and add (and configure) the following options:

   SSL remote CPU

   The workstation name of the remote master or fault-tolerant agent.

   SSL remote full port

   The port number defined for full SSL support on the remote master or fault-tolerant agent.

   The local options that specify the private key and certificate on the hosting fault-tolerant agent

   These are documented in the Setting local options on page 48).

   Note that if the hosting fault-tolerant agent hosts more than one network agent, the *`DATA_DIR`*`/methods` directory contains one `netmth.opts` file for every defined network agent. In this case the complete name of each `netmth.opts` file must become:

   ```
   network-agent-name_netmth.opts
   ```

   If the *`DATA_DIR`*`/methods` directory contains both *`network-agent-name`*`_netmth.opts` and `netmth.opts` files, only *`network-agent-name`*`_netmth.opts` is used. If multiple agents are defined and the directory contains only `netmth.opts`, this file is used for all the network agents.

The following example adds full SSL support to the example described in *A sample network agent definition* in *User's Guide and Reference*:

- This is the workstation definition for the `NETAGT` network agent:

```
CPUNAME NETAGT
 DESCRIPTION "NETWORK AGENT"
```

```
 OS OTHER
 NODE MASTERA.ROME.ITALY.COM
 TCPADDR 31117
 FOR maestro
  HOST MASTERB
  ACCESS NETMTH
END
```

- These are the full SSL security options in the `netmeth.opts` file of `NETAGT`:

```
######################################################
# Remote cpu parameters
######################################################

SSL remote full port = 31119
SSL remote CPU = MASTERA

######################################################
# Configuration Certificate
######################################################

SSL key                 ="C:\TWS\installations\SSL\XA.key"
SSL certificate         ="C:\TWS\installations\SSL\XA.crt"
SSL CA certificate      ="C:\TWS\installations\SSL\VeriSte.crt"
SSL key pwd             ="C:\TWS\installations\SSL\XA.sth"
SSL certificate chain   ="C:\TWS\installations\SSL\TWSCertificateChain.crt"
SSL random seed         ="C:\TWS\installations\SSL\random_file.rnd"
SSL auth mode           =cpu
SSL auth string        =tws
```

Note: The SSL configuration certificate options must refer to the private key and certificate defined on the hosting fault-tolerant agent.

- This is the workstation definition for `MASTERA` (the remote workstation):

```
CPUNAME MASTERA
  OS WNT
  NODE 9.168.68.55 TCPADDR 31117
  SECUREADDR 31119
  DOMAIN NTWKA
  FOR MAESTRO
    TYPE MANAGER
    AUTOLINK ON
    BEHINDFIREWALL OFF
    SECURITYLEVEL FORCE_ENABLED
    FULLSTATUS ON
    SERVER H
END
```

Chapter 7. Data maintenance

This chapter describes how to maintain your HCL Workload Automation database and other data files. The database is hosted on either the DB2® or Oracle RDBMS infrastructure, as you determined when you installed it. You should use the documentation of DB2® or Oracle for general instructions on database maintenance. This chapter describes the maintenance activities that are specific to HCL Workload Automation.

It comprises the following sections:

Backing up and restoring the database

To minimize downtime during disaster recovery, back up your master data files frequently to either offline storage or a backup master domain manager.

Using a backup master domain manager with a backup database

Set up a backup master domain manager that accesses a different database than the master domain manager, and get your database administrator to set up a mirror of the master domain manager's database onto the backup master domain manager's database. In this way your backup master domain manager not only receives copies of all the processing messages, as is provided for by the setting of the *FullStatus* attribute on the backup master domain manager, but is also able to access the mirrored database. The mirror frequency must be set high enough to match the frequency with which you change the database.

For more information about how to use a backup master domain manager, see Switching the master to a backup on page 390.

Backing up the configuration files

The configuration files used by HCL Workload Automation are found in the following places:

`useropts`

<user_home_dir>/TWS

For the `localopts`, `Sfinal`, `Security` and `*.msg` files

On Windows operating systems

<*TWA_home*>

On UNIX operating systems

<*TWA_DATA_DIR*>

HCL Workload Automation configuration files

On Windows operating systems

<*TWA_home*>`\TWS\mozart\*.*`

On UNIX operating systems

<*TWA_DATA_DIR*>`/TWS/mozart/*.*`

This directory might contain the following files:

`runmsgno`

This is used for the allocation of unique prompt numbers. On the master domain manager, this file should not be edited manually. This file does not need to be backed up.

`globalopts`

This is used to store a copy of three of the global properties stored in the database. It must be edited only in the circumstances described in Switching the master to a backup on page 390. This file should be backed up if it is edited.

Application server configuration files, such as `TWSConfig.properties`

On Windows operating systems

<TWA_home>`\usr\servers\engineServer\resources\properties`

On UNIX operating systems

<*TWA_DATA_DIR*>`/usr/servers/engineServer/resources/properties`

Forecast plan files

On Windows operating systems

<TWA_home>`\TWS\schedForecast`

On UNIX operating systems

<*TWA_DATA_DIR*>`/TWS/schedForecast`

Archived plan files.

On Windows operating systems

<TWA_home>`\TWS\schedlog`

On UNIX operating systems

<*TWA_DATA_DIR*>`/TWS/schedlog`

Trial plan files

On Windows operating systems

TWA_home`\TWS\schedTrial`

On UNIX operating systems

<*TWA_DATA_DIR*>/TWS/schedTrial

A detailed list of all files is not supplied, as there are too many files. Back up all the files in these directories.

Note: The **wa_pull_info** tool is provided for sending information to support, but can also be used to perform a backup of a DB2® database and some of the configuration files. .For more information, contact Software Support.

Backing up log files

If you use **wa_pull_info** for backup, you do not need to separately backup these files. For more information,contact software support.

Reorganizing the DB2 database

The database requires routine maintenance, as follows:

DB2®

The DB2® database has been set up to maintain itself, so there is little user maintenance to do. Periodically, DB2® checks the database by running an internal routine. DB2® determines when this routine must be run using a default policy. This policy can be modified, if need be, or can be switched off so that DB2® does not perform internal automatic maintenance. Using the statistical information that DB2® discovers by running this routine, it adjusts its internal processing parameters to maximize its performance.

This routine has also been made available for you to run manually in the case either where you feel that the performance of DB2® has degraded, or because you have just added a large amount of data , and anticipate performance problems. The routine is imbedded in a tool called dbrunstats, which can be run to improve performance while DB2® is processing data without causing any interruption.

It is also possible to physically and logically reorganize the database using the dbreorg script. This effectively re-creates the *tablespace* using its internal algorithms to determine the best way to physically and logically organize the tables and indexes on disk. This process is time-consuming, and requires that HCL Workload Automation is down while it is run, but it does provide you with a freshly reorganized database after major changes.

The use of these tools is described in Administrative tasks - DB2 on page 349.

These tools are implementations of standard DB2 facilities. If you are an expert user of DB2 you can use the standard facilities of DB2® to achieve the same results. For details go to DB2® documentation.

Maintaining the file system

Some of the file systems and directories need periodic maintenance. The details are given under the following topics:

- Avoiding full file systems on page 337
- Log files and archived files on page 343
- Temporary files on page 348
- Managing event message queue file sizes on page 349

Avoiding full file systems

Perhaps the most important maintenance task to perform is that of regularly controlling the file system or systems where HCL Workload Automation is installed, particularly on the master domain manager.

HCL Workload Automation has a number of files that can grow in size, either with more extensive use, such as the Symphony file, or in the event of network problems, such as the message files. If the Symphony file, in particular, cannot be expanded to contain all the required records, it might become corrupted. . If the Symphony file on the master domain manager is corrupted, you have no alternative but to restart HCL Workload Automation, losing the current plan's workload.

It is thus *most important* that you monitor the available space on the file system of the master domain manager where the Symphony file is generated, to ensure that there is always sufficient space for it to expand to cover any workload peaks, and also that there is sufficient space for message files to expand in the event of network problems. Your experience with your workload and your network will guide you to determine what are the acceptable limits of available disk space.

The approximate size of the Symphony file can be estimated in advance. It contains items related both to the plan (see Table 61: Algorithm for calculating the approximate size of the plan data in the Symphony file on page 337) and to the database (see Table 62: Algorithm for calculating the approximate size of the database data in the Symphony file on page 337). Estimate how many items you have in each category, multiply them by the indicated size in bytes, and sum them to find the approximate Symphony file size:

Table 61. Algorithm for calculating the approximate size of the plan data in the Symphony file

Data in Symphony file from the current plan	Bytes per instance
Per Job Scheduler instance:	512
Per job instance:	512
Per job "docommand" string > 40 bytes:	The length of the "docommand" string
Per ad hoc prompt:	512
Per file dependency:	512
Per recovery prompt:	512
Per recovery job:	512

Table 62. Algorithm for calculating the approximate size of the database data in the Symphony file

Data in Symphony file from the database (on the master domain manager)	Bytes per instance
Per workstation:	512
Per resource:	512

Table 62. Algorithm for calculating the approximate size of the database data in the Symphony file (continued)

Data in Symphony file from the database (on the master domain manager)	Bytes per instance
Per user:	256
Per prompt:	512
If the global option `ignoreCalendars` is set to *off*, per calendar:	512

If you find that disk space is becoming too limited, and you cannot dynamically extend it, you must create a backup master domain manager with much more space on its file system and then use the switchmgr command so that the backup becomes your new domain manager. Instructions on how to do this for any domain manager are given in Switching a domain manager on page 384, and in particular for a master domain manager, in Switching the master to a backup on page 390.

Monitoring the disk space used by HCL Workload Automation

You can use event-driven workload automation (EDWA) to monitor the disk space used by HCL Workload Automation and to start a predefined set of actions when one or more specific events take place. This type of event is managed by the TWSApplicationMonitor event provider. These types of events are supported on fault-tolerant agents only and not supported on dynamic agents. You can use EDWA to set up an event rule that monitors the used disk space, to verify that there is enough space to generate the Symphony and log files, and to allow the product to work correctly. For more information about event-driven workload automation, see the section about event-driven workload automation in the *User's Guide and Reference* .

When calculating disk space usage, the SSM agent divides the used space by the total space and then rounds up the result to the next highest integer. This is important to note especially when the usage is close to the specified threshold. See Table 63: Example for the ge operator on page 340 and Table 64: Example for the le operator on page 341 for examples on how this impacts the disk space usage calculation.

The following .XML file contains the definition of a sample event rule to monitor the disk usage percentage. This event rule triggers the MessageLogger action provider to write a message in a log file in an internal auditing database when the event occurs. For more information about the MessageLogger action provider, see User's Guide and Reference :

```
<?xml version="1.0"?>
<eventRuleSet  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules
   http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules/EventRules.xsd">
 <eventRule name="FILESYSTEMFULL" ruleType="filter" isDraft="yes">
  <eventCondition name="twsDiskMonEvt1" eventProvider="TWSApplicationMonitor" eventType="TWSDiskMonitor">
   <scope>
    * Disk is filling up
   </scope>
   <filteringPredicate>
    <attributeFilter name="FillingPercentage" operator="ge">
     <value>usage_percentage</value>
    </attributeFilter>
    <attributeFilter name="Workstation" operator="eq">
     <value>workstation_name</value>
    </attributeFilter>
    <attributeFilter name="SampleInterval" operator="eq">
     <value>sample_interval</value>
    </attributeFilter>
    <attributeFilter name="MountPoint" operator="eq">
     <value>mount_point</value>
    </attributeFilter>

   </filteringPredicate>
  </eventCondition>
  <action actionProvider="MessageLogger" actionType="MSGLOG" responseType="onDetection">
   <scope>
    OBJECT=ADWDAD MESSAGE=Disk is filling up
   </scope>
   <parameter name="ObjectKey">
    <value>object_key</value>
   </parameter>
   <parameter name="Severity">
    <value>message_severity</value>
   </parameter>
   <parameter name="Message">
    <value>log_message</value>
   </parameter>
  </action>
 </eventRule>
</eventRuleSet>
```

where:

usage_percentage

Is the disk usage percentage.

Important: When creating the event rule, always use a whole integer in the range 1-99 (inclusive) to express the threshold value. Fractions, decimals and negative numbers are not supported and the event rule is ignored.

Supported operators are as follows:

ge

causes the event generation when the disk usage exceeds the percentage specified by the threshold value. If the condition described in the rule already exists when you deploy the rule, the related event is generated. If the condition does not exist at the time the rule is deployed, then the event is generated when the disk usage percentage reaches or exceeds the threshold. The event is generated again only if the disk usage percentage subsequently falls below the threshold value and then rises again and either reaches or exceeds the threshold. If you restart the SSM agent and the disk usage percentage is higher than the threshold value, the event is generated again.

Table 63: Example for the ge operator on page 340 provides an example in which the **ge** operator is set to 70%.

Table 63. Example for the ge operator

Mailbox name	Disk usage percentage	Action
Sample(0)	>= 70%	event generated
Sample(0)	< 70%	event not generated
Sample(n-1)	< 70%	event not generated
Sample(n)	>= 70%	event generated
Sample(n+1)	>= 70%	event not gene

Table 63. Example for the ge operator (continued)

Mailbox name	Disk usage percentage	Action
		rated

le

causes the event generation when the disk usage percentage decreases under the threshold value. If the condition described in the rule already exists when you deploy the rule, the related event is not generated. The event is generated only the first time the specified disk usage percentage is reached. The event is generated again only if the disk usage percentage subsequently rises and exceeds the threshold value and then falls below the threshold. If you restart the SSM agent and the disk usage percentage is lower than the threshold value, the event is not generated until the disk usage percentage exceeds the threshold value and then falls below it again. Table 64: Example for the le operator on page 341 provides an example in which the **le** operator is set to 50%:

Table 64. Example for the le operator

Mailbox name	Disk usage percentage	Action
Sample (0)	<= 50%	event not generated
Sample (0)	> 50%	event not generated

Table 64. Example for the le operator (continued)

Mailbox name	Disk usage percentage	Action
Sample (n-1)	> 50%	event not generated
Sample (n)	<= 50%	event generated
Sample (n+1)	<= 50%	event not generated

workstation_name

Is the workstation on which the event is generated.

sample_interval

Is the interval, expressed in seconds, for monitoring the disk usage percentage.

mount_point

Is the mount point of the file system where HCL Workload Automation is installed, for example: "C:" on Windows™ systems or "/" on UNIX™ systems.

object_key

Is a key identifying the object to which the message pertains.

message_severity

Is the severity of the message.

log_message

Is the message to be logged.

Log files and archived files

Log files are produced from a variety of HCL Workload Automation activities. Other activities produce files which are archived after they have been used. The details are given in Table 65: Log and trace file maintenance on page 343.

Starting from version 10.1 Fix Pack 1, when generating a job log in the monitoring section of the Dynamic Workload Console, by default the log is stored in memory. This ensures compliance with the PCI standard.

To change this behavior and have the job log stored in a file, as it was in versions earlier than version 10.1 Fix Pack 1, perform the following steps:

1. Add the **com.ibm.tws.conn.plan.output.logtype** property in the `TWSConfig.properties` file.
2. Set the property to `file`.
3. Stop and start all the HCL Workload Automation processes.

Table 65. Log and trace file maintenance

Activity	Description	Location	Maintenance method
Fault-tolerant agent	Each HCL Workload Automation process logs its activities, writing them in log and trace message files:		
	Log messages These are messages intended for use directly by you, and provide information, errors and warnings about the processes.	**netman**	rmstdlist

Table 65. Log and trace file maintenance (continued)

Activity	Description	Location	Maintenance method
		Other processes **On Windows systems** `TWA_home\TWS\stdlist\logs\`*yyyymmdd*`_TWSMERGE.log` **On UNIX operating systems** <*TWA_DATA_DIR*>`/stdlist/logs/`*yyyymmdd*`_TWSMERGE.log` This is the default situation. You can set an option in the `localopts` file to create separate log files for the major processes.	
	Trace messages These are messages written when a problem occurs that you can probably not solve without the assistance of HCL Software Support.	**netman**	

Table 65. Log and trace file maintenance (continued)

Activity	Description	Location	Maintenance method
		On UNIX operating systems `<`*TWA_DATA_DIR*`>/stdlist/logs/`*yyyymmdd*`_TWSMERGE.log` This is the default situation. You can set an option in the `localopts` file to create separate trace files for the major processes.	
Master domain manager job management	The job manager process on the master domain manager archives the previous period's `Symphony` file.	**On Windows systems** *TWA_home*`>\TWS\schedlog\`*date* **On UNIX systems** `<`*TWA_DATA_DIR*`>/schedlog/`*date*	Manual
Job	Each job that runs under HCL Workload Automation control creates an output file. These files are archived.	**On Windows systems** *TWA_home*`\TWS\stdlist\`*date* **On UNIX systems** `<`*TWA_DATA_DIR*`>/stdlist/`*date* where *date* is in the format `yyyy.mm.dd`	rmstdlist
Dynamic agent	**Log messages**	**Windows** *TWA_home*`>\TWS\stdlist\JM\JobManager_message.log` **UNIX** `<`*TWA_DATA_DIR*`>/stdlist/JM/JobManager_message.log`	Regular housekeeping is performed through the configuration of several parameters. See Regular

Table 65. Log and trace file maintenance (continued)

Activity	Description	Location	Maintenance method
	Trace messages	**Windows** • *TWA_home*>\TWS\stdlist\JM\ITA_trace.log • *TWA_home*>\TWS\stdlist\JM\JobManager_trace.log • *TWA_home*>\TWS\JavaExt\logs\javaExecutor0.log **UNIX** • <*TWA_DATA_DIR*>/stdlist/JM/ITA_trace.log • <*TWA_DATA_DIR*>/stdlist/JM/JobManager_trace.log • <*TWA_DATA_DIR*>/JavaExt/logs/javaExecutor0.log	maintenance on page 101.
	Jobs with advanced options	**Windows** *TWA_home*>\TWS\stdlist\JM\date> **UNIX** <*TWA_DATA_DIR*>/stdlist/JM/*date*> where *date* is in the format yyyy.mm.dd	
Forecast and trial plan creation	The creation of forecast and trial plans require manual maintenance.	**Forecast plan** These files are to be maintained manually	Manual

Table 65. Log and trace file maintenance (continued)

Activity	Description	Location	Maintenance method
		Trail plan These files are to be maintained manually	

Audit	The audit facility writes log files.	**On Windows operating systems**
		`*TWA_home*\TWS\audit`
		On UNIX operating systems
		`<*TWA_DATA_DIR*>/audit`
			Manual

DB2® UDB	DB2® logs its activities.	Information about the location and viewing method for DB2® log files is supplied in the DB2® documentation, in Product Documentation for DB2®. The main file to control is the `db2diag.log` file, which is the most important DB2® diagnostic file, which, without intervention, grows endlessly with no reuse of wasted space. This does not apply, however, to the database log files used by HCL Workload Automation, which are set up for circular reuse of disk space, so they don't grow in size over a maximum value.	See the DB2® documentation.
Oracle database	Oracle logs its activities.	See the Oracle documentation.	See the Oracle documentation.

WebSphere Application Server Liberty	The application server writes log files.	On the server components:
		On Windows operating systems
		`<*TWA_home*>\TWS\stdlist\appserver\engineServer\logs`
		On UNIX operating systems
		`<*TWA_DATA_DIR*>/stdlist/appserver/engineServer/logs`
			Manual

Table 65. Log and trace file maintenance (continued)

Activity	Description	Location	Maintenance method
		On the Dynamic Workload Console: **On Windows operating systems** *TWA_home*>`\stdlist\appserver\dwcServer\logs` **On UNIX operating systems** <*TWA_DATA_DIR*>`/stdlist/appserver/dwcServer/logs`	
Netcool® SSM monitoring agent (not supported on IBM i systems)	The agent writes log files. (`ssmagent.log, traps.log`)	**On Windows operating systems** *TWA_home*>`\TWS\ssm\Log` **On UNIX operating systems** <*TWA_DATA_DIR*>`/EDWA/ssm/Log/`	Manual
Other	Other activities also write trace and log files.	**On Windows operating systems** *TWA_home*>`\TWS\methodes` **On UNIX operating systems** *TWA_home*>`/methods`	Manual

The easiest method of controlling the growth of these directories is to decide how long the log files are needed, then schedule a HCL Workload Automation job to remove any files older than the given number of days. Use the rmstdlist command for the process and job log files, and use a manual date check and deletion routine for the others. Make sure that no processes are using these files when you perform these activities.

See the *User's Guide and Reference* for full details of the rmstdlist command.

Note: The **rmstdlist** command might give different results on different platforms for the same scenario. This is because on UNIX® platforms the command uses the *–mtime* option of the find command, which is interpreted differently on different UNIX® platforms.

Temporary files

The HCL Workload Automation master domain manager uses temporary files, located in `<TWA_home>/TWS/tmp` or `/tmp` and named `TWS<XXXX>`, when compiling new production control databases. These files are deleted when compiling is complete.

This directory also contains the HCL Workload Automation installation files and log files, it is primarily used to handle temporary files that composer CLI creates as a work repository when it is invoked to perform CRUD actions against the HCL Workload Automation modeling objects. Directory rights are set to 777 to allow all users running the composer to have access. For security reasons the composer CLI is defined by using the sticky bit, so the files it creates can be owned by users different from the HCL Workload Automation installation user. HCL Workload Automation conman can be used by any user therefore the folder is 777. If the users eligible to use conman/composer are inserted into the HCL Workload Automation group then the permission can be set to 774. In that way, only these users will be able to run conman/composer commands

Managing event message queue file sizes

This publication contains the following information with respect to managing event message queue file sizes:

- See Planning space for queues on page 294 to learn about planning space for message event queues (and also how to use evtsize to resize the queues
- See Managing the event processor on page 419 to learn about managing the EIF event queue
- See Disk Space on page 459 to learn about the impacts that increased fault tolerance can have on message queues
- See Workload spreading on page 456 to learn about how to avoid bottlenecks in the `Mailbox.msg` queue.

Administrative tasks - DB2®

A set of scripts and SQL files is provided to perform actions such as granting rights or reorganizing the database. These files are located in *`inst_dir`*`/dbtools/db2/script`. To use these files, copy the relevant folder to the database server. The DB2® tools must be run by a user who has the following permissions:

- DB2 administrator permissions – the user must be defined to DB2 as a DB2 Administrator
- Full access (777) to the HCL Workload Automation installation directory

The available files are as follows:

dbrunstats

This script runs the DB2 statistics program, to maximize the performance of DB2.

dbreorg

This script reorganizes the database. See Reorganizing the DB2 database on page 352 for a full description of how to use the tool.

dbgrant

This script adds grants to new users on HCL Workload Automation DB schema for the views that can be used to generate reports

To find out how to perform specific administrative tasks on DB2®, see:

- Changing DB2 passwords on page 350
- Locating the DB2 tools on page 350

- User permissions for running the DB2 tools on page 350
- Running DB2 maintenance manually on page 351
- Reorganizing the DB2 database on page 352
- Monitoring the lock list memory on page 353

Changing DB2® passwords

About this task

You can change passwords used by DB2®, such as the <*TWS_user*> password or the passwords of the user IDs used by HCL Workload Automation to access the database, as described in Changing key HCL Workload Automation passwords on page 405. If you need to change other DB2-related passwords, follow the instructions in the DB2® documentation.

After you have changed the password, modify the **db.password** parameter in the `datasource_db2.xml` configuration file, as described in Changing the properties for the database on page 412. You can optionally encrypt the password, as described in the topic about the secure script in *HCL Workload Automation: Planning and Installation*.

Locating the DB2® tools

About this task

HCL Workload Automation is supplied with a small set of tools that you use to perform the following administrative tasks for DB2®:

- Run the DB2® statistics program, to maximize the performance of DB2® (`dbrunstats`). See Running DB2 maintenance manually on page 351 for a full description of how to use the tool.
- Reorganize the database (`dbreorg`). See Reorganizing the DB2 database on page 352 for a full description of how to use the tool.

The tools are available in the following directory:

```
inst_dir/TWS/dbtools/db2
```

Copy the tools to the DB2 server workstation where the HCL Workload Automation database is located.

Note: The tools in this directory might include some that are for the use of HCL Software Support:

dbmove

Do not run this script. To do so might damage or overwrite the data in your database.

User permissions for running the DB2® tools

The DB2® tools must be run by a user who has the following permissions:

- DB2® administrator permissions – the user must be defined to DB2® as a DB2® Administrator
- Full access (777) to the HCL Workload Automation installation directory

Running DB2® maintenance manually

At installation, DB2® automatic maintenance is switched on, which means that DB2® periodically checks to see if it needs to collect new database statistics, so that it can perform the maintenance, adjusting the performance parameters to maximize performance.

This section describes how to perform the DB2® maintenance process on demand, instead of waiting for DB2® to do it according to its automatic maintenance policy. The process is run by the dbrunstats tool which you can run whenever you need to, without stopping DB2® or interrupting its processing.

To run this tool, follow this procedure:

1. Locate the DB2® tools: see Locating the DB2 tools on page 350.
2. Check that the user who is going to run the procedure has the appropriate rights (see User permissions for running the DB2 tools on page 350)
3. On the DB2 server, open a DB2® shell, as follows:

   UNIX™

   Follow these steps:

   a. Issue the command su - db2inst1, or change to the subdirectory `sqllib` of the home directory of the owner of the DB2® instance (by default *db2inst1*)
   b. Launch the command . ./db2profile

   Windows™

   Select from the **Start** menu, **Programs** →; **IBM DB2** →; **Command Line Tools** →; **Command Window**
4. Check that the command shell is correctly initialized by issuing the command db2, and checking that the command is recognized.
5. Issue the command quit to leave the DB2® Processor mode.
6. From within the shell, browse to the directory where you copied the script.
7. Run the script:

   UNIX™

   dbrunstats.sh database [user [password]]

   Windows™

   dbrunstats database [user [password]]

   where:

   database

   The name of the database. The default name is TWS. Supply this value unless you have changed it.

   user

   The DB2® administration user. If this is omitted, the ID of the user running the command will be used.

password

The password of the DB2® administration user. If this is omitted, it will be requested interactively.

The script runs, giving you various messages denoting its progress and successful conclusion. At the end (it is not particularly time-consuming) the database performance parameters have been reset to maximize performance.

Reorganizing the DB2® database

About this task

Using this tool, the database physically reorganizes the data tables and indexes, optimizing disk space usage and ease of data access. The process is time-consuming, requires that the database is backed up, and that HCL Workload Automation is stopped. However, at the end you have a database that is completely reorganized.

To reorganize the database follow this procedure:

1. Stop WebSphere Application Server Liberty and appservman by running the following command:

   ```
   conman "stopappserver;wait"
   ```

   See Starting and stopping the application server and appservman on page 431 for full details.
2. Back up the HCL Workload Automation database. Follow the instructions in the database vendor documentation, as appropriate.
3. Check that the user who is going to run the procedure has the appropriate rights (see User permissions for running the DB2 tools on page 350)
4. On the DB2 server where you copied the script, open a DB2® shell, as follows:

   UNIX™

   Follow these steps:

   a. Issue the command su - db2inst1, or change to the subdirectory `sqllib` of the home directory of the owner of the DB2® instance (by default *db2inst1*)
   b. Launch the command . ./db2profile

   Windows™

   Select from the **Start** menu, **Programs** →; **IBM DB2** →; **Command Line Tools** →; **Command Window**
5. Check that the command shell is correctly initialized by issuing the command db2, and checking that the command is recognized.
6. Issue the command quit to leave the DB2® Processor mode.
7. From within the shell, browse to the directory where you copied the script.
8. Run the script:

   UNIX™

   dbreorg.sh database [user [password]]

   Windows™

   dbreorg database [user [password]]

where:

database

The name of the database. The default name is TWS. Supply this value unless you have changed it.

user

The DB2® administration user. If this is omitted, the ID of the user running the command will be used.

password

The password of the DB2® administration user. If this is omitted, it will be requested interactively.

The script runs, giving you various messages denoting its progress and successful conclusion.

9. Restart WebSphere Application Server Liberty and appservman by running the following command:

```
conman "startappserver;wait"
```

See Starting and stopping the application server and appservman on page 431 for full details.

Monitoring the lock list memory

About this task

If the memory that DB2® allocates for its lock list begins to be fully used, DB2® can be forced into a "*lock escalation*", where it starts to lock whole tables instead of just individual table rows, and increasing the risk of getting into a deadlock.

This happens especially when there are long transactions, such as the creation or extension of a plan (production, trial, or forecast).

To avoid this problem occurring, set the automatic notification in the DB2® Health Center, so that you can be advised of any lock list problems building up.

However, if you think that deadlock situations have been occurring, follow this procedure to verify:

1. With the WebSphere Application Server Liberty active, log on as DB2® administrator to the DB2® server, for example,

   su - db2inst1

2. Run the following command to determine where the HCL Workload Automation database is located:

   db2 list active databases

   The output might be as follows:

```
Database name                    = TWS
Applications connected currently = 2
Database path                    = /home/db2inst1/db2inst1/NODE0000/SQL00002/
```

3. Run:

   cd <Database_path>/db2event/db2detaildeadlock

4. Connect to the HCL Workload Automation database, for example:

```
db2 connect to TWS
```

5. Flush the event monitor that watches over deadlocks (active by default) with the following:

   ```
   db2 flush event monitor db2detaildeadlock
   ```

6. Disconnect from the database with:

   ```
   db2 terminate
   ```

7. Obtain the event monitor output with:

   ```
   db2evmon -path . > deadlock.out
   ```

   The file `deadlock.out` now contains the complete deadlock history since the previous flush operation.

8. To find out if there have been deadlocks and when they occurred, run:

   ```
   grep "Deadlock detection time" deadlock.out
   ```

   The output might be as follows:

   ```
   Deadlock detection time: 11/07/2008 13:02:10.494600
   Deadlock detection time: 11/07/2008 14:55:52.369623
   ```

9. But the fact that a deadlock occurred does not necessarily mean that the lock list memory is inadequate. For that you need to establish a relationship with lock escalation. To find out if there have been lock escalation incidents prior to deadlocks, run:

   ```
   grep "Requesting lock as part of escalation: TRUE" deadlock.out
   ```

   The output might be as follows:

   ```
   Requesting lock as part of escalation: TRUE
   Requesting lock as part of escalation: TRUE
   ```

   If there has been lock escalation related to deadlocks, it is a good idea to modify the values of the following parameters.

   LOCKLIST

   This configures, in 4KB pages, the amount of memory allocated to locking management

   MAXLOCKS

   This configures the percentage of the memory that a single transaction can use, above which DB2® escalates, even though the memory might not be full

10. To determine the values currently being applied to the HCL Workload Automation database, do the following:

    ```
    db2 get db cfg for TWS | grep LOCK
    ```

    The output might be as follows:

```
Max storage for lock list (4KB)              (LOCKLIST) = 8192
Percent. of lock lists per application       (MAXLOCKS) = 60
Lock timeout (sec)                        (LOCKTIMEOUT) = 180
```

The example shows the typical output for the HCL Workload Automation database if no modification has taken place to these values:

- "8192" = 4KB x 8192 pages = 32 MB of memory
- "60" = 60% – the percentage of memory that a single transaction can occupy before triggering an escalation
- "180" = 3 minutes of timeout for the period a transaction can wait to obtain a lock

11. The most straightforward action to take is to double the amount of memory to 64MB, which you do with the command:

    db2 update db cfg for TWS using LOCKLIST 16384 immediate

12. Alternatively, you can set DB2® to automatically modify the LOCKLIST and MAXLOCKS parameters according to the amount of escalation being experienced and the available system memory. This self-tuning is a slow process, but adapts the database to the needs of the data and the available system configuration. It is done by setting the values of these parameters to AUTOMATIC, as follows:

    db2 update db cfg for TWS using LOCKLIST AUTOMATIC immediate

    DB2® responds with messages telling you that MAXLOCKS has also been set to AUTOMATIC:

    ```
    SQL5146W "MAXLOCKS" must be set to "AUTOMATIC" when "LOCKLIST" is "AUTOMATIC".
    ```

    ```
    "MAXLOCKS" has been set to "AUTOMATIC"
    ```

    Note: The self-tuning facility is only available from V9.1 of DB2®.

Administrative tasks - Oracle

A script is provided to perform actions such as granting rights or reorganizing the database. The file is located in `*inst_dir*/dbtools/oracle/script`

Oracle

dbgrant

This script grants the user permissions for the Dynamic Workload Console views. See the Dynamic Workload Console online help for full details.

A second script named **dbmove** is available, but it is to be run only if requested by Software Support.

Note: Do not run the dbmove script unless instructed to do so by Software Support. Running the script autonomously might damage or overwrite the data in your database.

Ensure you have the following permissions before running the scripts:

- Oracle administrator permissions – the user must be defined to Oracle as an administrator
- Full access (777) to the HCL Workload Automation installation directory

You can perform a number of other tasks on Oracle databases as follows:

Changing the Oracle access password

For more information, see Changing the properties for the database on page 412.

Maintaining the Oracle database

Like DB2, Oracle has a routine that regularly maintains the database. Similarly, this too can be run manually. The tool is invoked as follows:

```
dbms_stats.gather_schema_statsschema_owner
```

See the Oracle documentation for full details of how and when to run it.

Obtaining information about the HCL Workload Automation databases installed on an Oracle instance

To determine which HCL Workload Automation databases are installed on an Oracle instance, do the following:

```
su - oracle (UNIX only)
 sqlplus system/system_password@service_name
SQL> select * from all_tws_schemas;
```

The output should look like the following:

```
SCHEMA_NAME
------------------------------
MDL
mdm10.2.5 <TWS_user>
```

Note:

1. More than one instance of HCL Workload Automation can be shared in one instance of Oracle, using different schemas.
2. In Oracle, the concept of "schema" and "user" are the same, so dropping an Oracle schema means dropping an Oracle user, which you do as follows:

   ```
   SQL> drop user MDL cascade;
   ```

Customizing your RDBMS server

About this task

You can perform several operations to customize your database, such as:

- Upgrading the JDBC drivers. For more information, see the FAQ about customizing the JDBC drivers in *Planning and Installation Guide*.
- Changing the instance owner, by editing the **db.user** property in the `datasource_<db_vendor>.xml` file. For more information, see Changing the properties for the database on page 412.

- Relocating the DB to the another host by editing the **db.serverName** property in the `datasource_<db_vendor>.xml` file. For more information, see Changing the properties for the database on page 412.
- Changing the database port, database name, or database access credentials by editing the **db.portNumber**, **db.databaseName**, **db.user**, and **db.password** properties in the `datasource_<db_vendor>.xml` file. For more information, see Changing the properties for the database on page 412.

If you want to upgrade your database version, change the instance owner, or relocate it to a different host, complete the following steps:

1. If you are changing DB2®, check the *node directory* and *database directory* and make a note of the current configuration. To do this, issue the following commands at the DB2® command-line:

   ```
   db2 list database directory
   ```

   where the `show detail` attribute is specified to give the full information in the directory.

   Make a note of the displayed details.
2. Stop the application server, using the command

   ```
   conman stopappserver ;wait
   ```
3. Make the upgrade, instance owner change, or relocation, of the database following the instructions from your database supplier.
4. If you have changed the database access credentials, you will need to update the application server's security properties, as described in Changing the security settings on page 418.
5. Reconfigure the database for HCL Workload Automation, as follows:

   DB2®

   a. Check the *node directory* and *database directory*, as you did in step 1 on page 357
   b. If necessary, modify the data displayed by these commands to match the data you noted in step 1 on page 357. If you are not certain of how to do this, contact HCL Software Support for assistance.

   Oracle

   Check the Oracle Listener and make sure that the service name is correctly specified.
6. Restart the database.
7. Restart the application server, using the command:

   ```
   conman startappserver ;wait
   ```

Connecting the master domain manager to a new database

Connecting the master domain manager to a new database involves reconfiguring the system to recognize and interact with the new database. This process typically includes updating connection strings, ensuring database drivers are compatible, migrating relevant data, and validating connectivity. This ensures the master domain manager can continue to perform its tasks without interruption, leveraging the new database resources and capabilities.

About this task

This section applies to HCL Workload Automation master domain managers and their backups. It documents how to connect the master domain manager and backup master domain manager to a new database. If your environment contains pools, see Connecting the master domain manager to a new database when using pools on page 361.

Note: This cloning procedure does not clone the following information from the source environment:

- The preproduction plan
- The history of job runs and job statistics
- The audit records
- The state of running event rule instances. This means that any complex event rules, where part of the rule has been satisfied prior to cloning of the environment, are generated as new rules after the cloning procedure. Even if the subsequent conditions of the event rule are satisfied, the record that the first part of the rule was satisfied is no longer available, so the rule will never be completely satisfied.

Follow this procedure to export data from your previous database, import it to a new one, and connect the master domain manager and backup master domain manager to the new database.

1. Install the new database.
2. Run the configureDb command to create the new database on the new target RDBMS. For more information, see the section about the configureDB script in *HCL Workload Automation: Planning and Installation*.
3. On the master domain manager, extract all scheduling object definitions and global options from the previous RDBMS running the dataexport command.

   ```
   dataexport source_dir export_dir
   ```

   where:

   source_dir

   Is the *TWS_HOME* directory of HCL Workload Automation.

   export_dir

   Is the directory where the export files are created.

   For example:

   ```
   dataexport.cmd F:\IWS1025\twsDBuser F:\IWS1025\export
   ```

   For more information, see the section about the dataexport command in *User's Guide and Reference*.
4. Verify that the following files were created in `export_dir`:
   - `acls.def`
   - `calendars.def`

- `erules.def`
- `folders.def`
- `globalOpts.def`
- `jobs.def`

  Note: The record length supported by DB2® is 4095 bytes, but it decreases to 4000 bytes with Oracle. When you migrate your job definitions to Oracle, any job with task string (scripts or commands) exceeding 4000 bytes in length are not migrated. In this case, the data import utility replaces the job definition with a dummy job definition and sets the job priority to `0`, guaranteeing that successors are not run.

- `parms.def`
- `prompts.def`
- `rcgroups.def`
- `resources.def`
- `scheds.def`
- `sdoms.def`
- `srols.def`
- `topology.def`
- `users.def` (includes encrypted user passwords)
- `vartables.def`

5. Run exportserverdata to export dynamic domain manager data from the database. This command is not documented because it is an internal command. The command is located in `/opt/HCL/TWA/TDWB/bin/`. For the purpose of this procedure, use the following syntax:

   ```
   ./exportserverdata.sh -dbUsr username -dbPwd password
   [-exportFile exportFile]
   ```

   where:

   dbUsr

   Is the database user. This parameter is required.

   dbPwd

   Is the database user password. This parameter is required.

   -exportFile

   Is the file name to which the server data is to be exported from the database. This parameter is required if you use a file different from the default `server.properties` file, located in the same path as the exportserverdata command. If you use a file other than `server.properties`, specify the name and path of the file.

6. Stop WebSphere Application Server Liberty on the master domain manager and backup master domain manager.
7. Reconfigure the master domain manager and backup master domain manager by performing the following changes:

a. Edit the WebSphere Application Server Liberty `datasource.xml` file, so that it points to the new RDBMS. For more information, see Changing the properties for the database on page 412.
b. In the *TWA_DATA_DIR*`/usr/servers/engineServer/resources/properties/TWSConfig.properties` file, uncomment the `com.ibm.tws.dao.rdbms.rdbmsName` line and specify the name of your RDBMS. For more information, see Changing the properties for the database on page 412.
c. In the *TWA_DATA_DIR*`/broker/config/DAOCommon.properties` file, change the `com.ibm.tdwb.dao.rdbms.rdbmsName` line to specify the name of your RDBMS.
d. In the *TWA_DATA_DIR*`/broker/config/CLIConfig.properties` file, change the following strings:

  com.ibm.tdwb.dao.rdbms.rdbmsName

    to specify the name of your RDBMS.

  com.ibm.tdwb.dao.rdbms.jdbcPath

    to specify the path to your JDBC connection string.

  com.ibm.tdwb.dao.rdbms.jdbcDriver

    to specify the name of the class implementing your JDBC driver.

e. In the *TWA_DATA_DIR*`/usr/servers/engineServer/jvm.options` file, specify the path to the database drivers.
f. In the TWA_home`/TDWB/bin/tdwb_env.sh` file, specify the path to the database drivers.

8. Run addserverdata to add dynamic domain manager data in the database. The command is located in `/opt/HCL/TWA/TDWB/bin/`. This command is not documented because it is an internal command. For the purpose of this procedure, use the following syntax:

```
./addserverdata.sh -dbUsr username -dbPwd password
-importFile importFile {[-MDM true]}
```

where:

dbUsr

Is the database user. This parameter is required.

dbPwd

Is the database user password. This parameter is required.

-importFile

Is the file name from which the server data is to be imported into the database. This parameter is required if you use a file different from the default `server.properties` file, located in the same path as the addserverdata command. If you use a file other than `server.properties`, specify the name and path of the file.

MDM

Specifies whether the workstation where you run the command is the master domain manager. This parameter is required on the master domain manager and is optional on the dynamic domain manager.

9. Restart WebSphere Application Server Liberty on the master domain manager.

10. Reload the shell session for the changes to take effect and check that the environment is working correctly by running the following commands:

    ```
    optman ls
    composer display cpu=MDM_name
    ```

11. Wait for the dynamic workload broker definition to be created, and subsequently for all dynamic agents to register with the master domain manager.
12. Run the dataimport command to import data from your previous RBDMS.

    ```
    dataimport.cmd F:\IWS1025\twsDBuser F:\IWS1025\export
    ```

    where:

    source_dir

    is the *TWS_HOME* directory of the source environment instance of HCL Workload Automation, which corresponds to *installation_dir*/TWS.

    export_dir

    is the directory where you copied the object definitions and the global options retrieved from the source environment.

    For more information, see the section about the dataimport command in *User's Guide and Reference*.
13. Run the following command to align the plan with the information inserted in the database:

    ```
    Jnextplan -for 0000 -noremove
    ```

14. Run the following command to extend the plan by 24 hours:

    ```
    JnextPlan -to startOfDay tz your_timezone -noremove
    ```

    For more information about Jnextplan, see the section about the JnextPlan command in *User's Guide and Reference*.

    Result

    The master domain manager and backup master domain manager are now configured to connect to the new database and your environment is up and running.

    If you need to connect the Dynamic Workload Console to a new database, see the topic about connecting the Dynamic Workload Console to a new database in Planning and Installation.

Connecting the master domain manager to a new database when using pools

Connecting the master domain manager to a new database involves reconfiguring the system to recognize and interact with the new database. This process typically includes updating connection strings, ensuring database drivers are compatible, migrating relevant data, and validating connectivity. This ensures the master domain manager can continue to perform its tasks without interruption, leveraging the new database resources and capabilities.

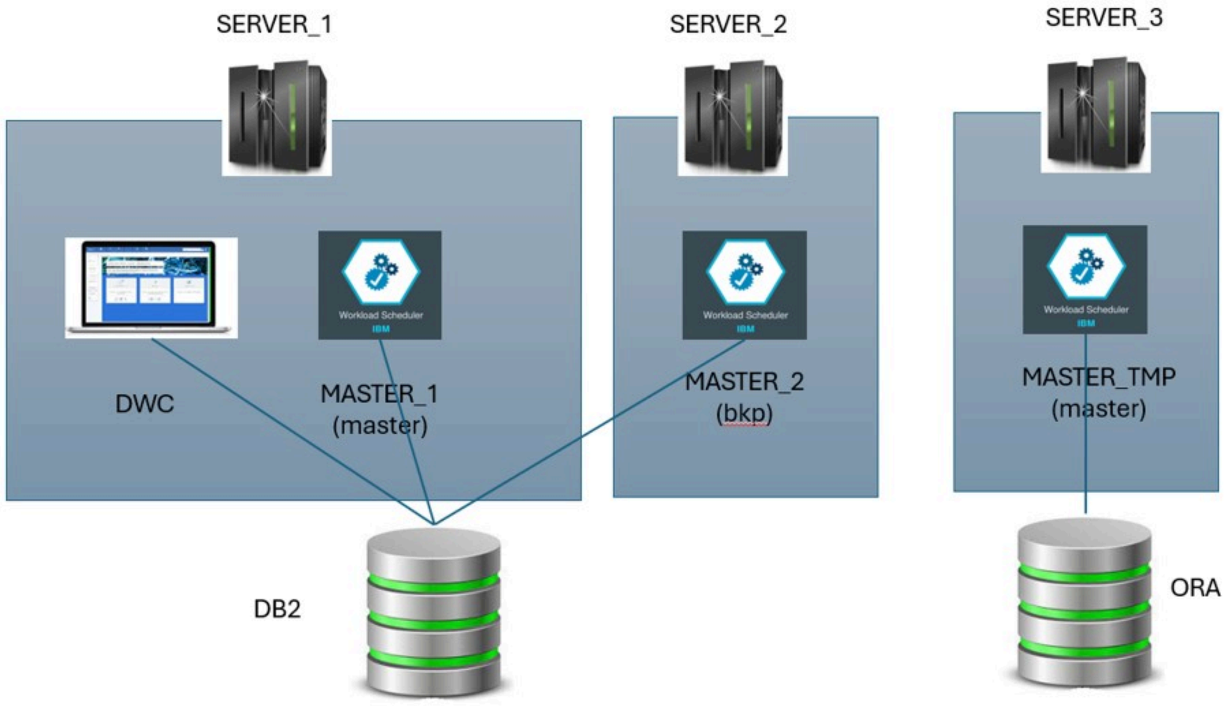
About this task

This section applies to HCL Workload Automation master domain managers and their backups. It documents how to connect the master domain manager and backup master domain manager to a new database.

Note: This cloning procedure does not clone the following information from the source environment:

- The preproduction plan
- The history of job runs and job statistics
- The audit records
- The state of running event rule instances. This means that any complex event rules, where part of the rule has been satisfied prior to cloning of the environment, are generated as new rules after the cloning procedure. Even if the subsequent conditions of the event rule are satisfied, the record that the first part of the rule was satisfied is no longer available, so the rule will never be completely satisfied.

Follow this procedure to export data from your previous database, import it to a new one, and connect the master domain manager and backup master domain manager to the new database. This procedure involves your existing master domain manager, **MASTER_1**, and a new one, **MASTER_TMP**, which is installed solely to perform the initial configuration of the new database, ensuring it is properly set up before data import.



1. Install the new database.
2. Install a new master domain manager, named **MASTER_TMP**. This master domain manager is installed only to perform the initial configuration of the new database, ensuring it is properly set up before importing data.
3. Run the configureDb command to create the new database on the new target RDBMS. For more information, see the section about the configureDB script in *HCL Workload Automation: Planning and Installation*.

4. On **MASTER_1**, extract all scheduling object definitions and global options from the previous RDBMS running the dataexport command.

   ```
   dataexport source_dir export_dir
   ```

   where:

   source_dir

   Is the *TWS_HOME* directory of HCL Workload Automation.

   export_dir

   Is the directory where the export files are created.

   For example:

   ```
   dataexport.cmd F:\IWS1025\twsDBuser F:\IWS1025\export
   ```

   For more information, see the section about the dataexport command in *User's Guide and Reference*.
5. Verify that the following files were created in `export_dir`:
   - `acls.def`
   - `calendars.def`
   - `erules.def`
   - `folders.def`
   - `globalOpts.def`
   - `jobs.def`

     Note: The record length supported by DB2® is 4095 bytes, but it decreases to 4000 bytes with Oracle. When you migrate your job definitions to Oracle, any job with task string (scripts or commands) exceeding 4000 bytes in length are not migrated. In this case, the data import utility replaces the job definition with a dummy job definition and sets the job priority to `0`, guaranteeing that successors are not run.

   - `parms.def`
   - `prompts.def`
   - `rcgroups.def`
   - `resources.def`
   - `scheds.def`
   - `sdoms.def`
   - `srols.def`
   - `topology.def`
   - `users.def` (includes encrypted user passwords)
   - `vartables.def`
6. On **MASTER_1**, run exportserverdata to export dynamic domain manager data from the database. This command is not documented because it is an internal command. The command is located in `/opt/HCL/TWA/TDWB/bin/`. For the purpose of this procedure, use the following syntax:

```
./exportserverdata.sh -dbUsr username -dbPwd password
[-exportFile exportFile]
```

where:

dbUsr

Is the database user. This parameter is required.

dbPwd

Is the database user password. This parameter is required.

-exportFile

Is the file name to which the server data is to be exported from the database. This parameter is required if you use a file different from the default `server.properties` file, located in the same path as the exportserverdata command. If you use a file other than `server.properties`, specify the name and path of the file.

7. **On MASTER_1**, relocate the definitions of the specified components from file `topology.def` to a newly created file named `topology_env.def`, placing it in a different folder.
   - MASTER_1 (modify type from MANAGER to FTA)
   - MASTER_1_1
   - MASTER_2
   - MASTER_2_1
   - SERVER_XA

   Agent definitions remain in the original `topology.def` file.
8. On **MASTER_TMP**, run the addserverdata command. The command is located in `/opt/HCL/TWA/TDWB/bin/`. This command is not documented because it is an internal command. For the purpose of this procedure, use the following syntax:

   ```
   ./addserverdata.sh -dbUsr <ORACLE_user> -dbPwd <ORACLE_password>
   -importFile <file_created_by_exportserverdata>
   ```
9. On **MASTER_TMP**, import the master domain manager definition, using the following command:

   ```
   composer replace topology_env.def
   ```
10. Modify the definitions of **MASTER_1** to **MANAGER** and **MASTER_TMP** to **FTA**.
11. On **MASTER_1**, set the fence to **HIGH**, by running the following command:

    ```
    conman fence @ HI noask
    ```
12. Stop WebSphere Application Server Liberty on both **MASTER_1** and **MASTER_2** (**MASTER_2** is the backup master domain manager).
13. Reconfigure the master domain manager and backup master domain manager by performing the following changes:

a. Edit the WebSphere Application Server Liberty `datasource.xml` file, so that it points to the new RDBMS. For more information, see Changing the properties for the database on page 412.
b. In the *TWA_DATA_DIR*`/usr/servers/engineServer/resources/properties/TWSConfig.properties` file, uncomment the `com.ibm.tws.dao.rdbms.rdbmsName` line and specify the name of your RDBMS. For more information, see Changing the properties for the database on page 412.
c. In the *TWA_DATA_DIR*`/broker/config/DAOCommon.properties` file, change the `com.ibm.tdwb.dao.rdbms.rdbmsName` line to specify the name of your RDBMS.
d. In the *TWA_DATA_DIR*`/broker/config/CLIConfig.properties` file, change the following strings:

   com.ibm.tdwb.dao.rdbms.rdbmsName

   to specify the name of your RDBMS.

   com.ibm.tdwb.dao.rdbms.jdbcPath

   to specify the path to your JDBC connection string.

   com.ibm.tdwb.dao.rdbms.jdbcDriver

   to specify the name of the class implementing your JDBC driver.

e. In the *TWA_DATA_DIR*`/usr/servers/engineServer/jvm.options` file, specify the path to the database drivers.
f. In the TWA_home`/TDWB/bin/tdwb_env.sh` file, specify the path to the database drivers.

14. On On **MASTER_2**, run addserverdata to add dynamic domain manager data in the database. The command is located in `/opt/HCL/TWA/TDWB/bin/`. This command is not documented because it is an internal command. For the purpose of this procedure, use the following syntax:

```
./addserverdata.sh -dbUsr username -dbPwd password
-importFile importFile {[-MDM true]}
```

where:

dbUsr

Is the database user. This parameter is required.

dbPwd

Is the database user password. This parameter is required.

-importFile

Is the file name from which the server data is to be imported into the database. This parameter is required if you use a file different from the default `server.properties` file, located in the same path as the addserverdata command. If you use a file other than `server.properties`, specify the name and path of the file.

MDM

Specifies whether the workstation where you run the command is the master domain manager. This parameter is required on the master domain manager and is optional on the dynamic domain manager.

15. On **MASTER_1** restart WebSphere Application Server Liberty on the master domain manager.

16. On **MASTER_1**, reload the shell session for the changes to take effect and check that the environment is working correctly by running the following commands:

    ```
    optman ls
    composer display cpu=MDM_name
    ```

17. Remove the workstation definitions for **SERVER_TMP** and **SERVER_TMP_1**.
18. On **MASTER_1**, run the dataimport command to import data from your previous RBDMS.

    ```
    dataimport.cmd F:\IWS1025\twsDBuser F:\IWS1025\export
    ```

    where:

    source_dir

    is the *TWS_HOME* directory of the source environment instance of HCL Workload Automation, which corresponds to *installation_dir*/TWS.

    export_dir

    is the directory where you copied the object definitions and the global options retrieved from the source environment.

    For more information, see the section about the dataimport command in *User's Guide and Reference*.
19. Restart **MASTER_2**.
20. On **MASTER_1**, set the fence to 0, as follows:

    ```
    conman fence @ 0 noask
    ```

21. Run the following command to align the plan with the information inserted in the database:

    ```
    Jnextplan -for 0000 -noremove
    ```

22. Run the following command to extend the plan by 24 hours:

    ```
    JnextPlan -to startOfDay tz your_timezone -noremove
    ```

    For more information about Jnextplan, see the section about the JnextPlan command in *User's Guide and Reference*.

    Result

    The master domain manager and backup master domain manager are now configured to connect to the new database and your environment is up and running. Your pools are migrated to the new database.

    If you need to connect the Dynamic Workload Console to a new database, see the topic about connecting the Dynamic Workload Console to a new database in Planning and Installation.

Auditing facilities

Describes the audit facilities to track changes in the database and the plan, as well as those that track changes to objects involved in dynamic workload scheduling.

In the Dynamic Workload Console, operators and schedulers can review all changes to scheduling objects, both in the database and in the plan, discover which user performed a specific change, and when the change was performed. Admnistrators can request that each user provide a justification when making changes to an object and log this information in audit trails. Application developers and schedulers can compare and restore previous versions of each changed object, and promote the job workflow from development to test or production environments.

For more information, see the section about keeping track of changes in *Dynamic Workload Console User's Guide*.

Audit trails are useful to check enforcement and effectiveness of IT controls, for accountability, and vulnerability and risk analysis. IT organizations can also use auditing of security-related critical activities to aid in investigations of security incidents. When a security incident occurs, audit trails enable analysis of the history of activities (who did what, when, where, and how) that occurred prior to the security incident, so appropriate corrective actions can be taken. For these reasons, audit trails might need to be archived and accessible for years.

For more information about maintaining audit trails, see Database and plan audit on page 367.

Database and plan audit

An auditing option is available to track changes to the database and the plan. It is disabled by default. It is described in these sections:

- Enabling and storing audit trails on page 367
- Audit log header format on page 369
- Audit log body format on page 370
- Sample audit log entries on page 374

Enabling and storing audit trails

You can maintain audit trails for information stored in the database and in the plan. By default, auditing is enabled. To disable auditing, use the following global options:

enDbAudit

Enables auditing of the information available in the database.

enPlanAudit

Enables auditing of the information available in the plan.

For more information about global options, see Global options - detailed description on page 27.

You can store auditing information in a file, in the HCL Workload Automation database, or in both. To define in which type of store to log the audit records, use the **auditStore** global option. For more information about global options, see Global options - detailed description on page 27. When auditing database information, all the user modifications are logged, including the current definition of each modified database object. If an object is opened and saved, the action is logged even if no modification was made. When auditing plan information, all the user modifications to the plan are logged. Actions are logged whether or not they are successful.

Choose the storage location of audit records according to the type of information you are auditing, whether it is database or plan:

auditing of the information available in the database (enDbAudit global option)

You can track changes to the database in a file, in the database itself, or in both. To define which type of store to log the audit records, use the **auditStore** global option. For more information about global options, see Global options - detailed description on page 27. All the user modifications are logged, including the current definition of each modified database object. If an object is opened and saved, the action is logged even if no modification was made.

auditing of the information available in the plan (enPlanAudit global option)

You can track changes to the plan in a file. When you enable auditing of the information available in the plan, the information is saved to a file. All the user modifications to the plan are logged. Actions are logged whether or not they are successful.

Storing auditing information in a file (auditStore=file)

This storage location is available when you audit information in the database (**enDbAudit** global option) and in the plan (**enPlanAudit** global option). Choose to store auditing information in a file by setting the **auditStore** global option to `file`. For more information about the **auditStore** global option, see Global options - detailed description on page 27.

Each audit log provides audit information for one day, from 00:00:00 UTC to 23:59:59 UTC regardless of the time zone of the local workstation, but the log file is created only when an action is performed or the WebSphere Application Server Liberty is started.

The files are called `yyyymmdd`, and are created in the following directories:

`<TWA_home>/TWS/audit/plan <TWA_home>/TWS/audit/database`

Audit entries are logged to a flat text file in `.json` format on the master domain manager and backup master domain manager.

The log formats are the same for both plan and database. The logs consist of a header portion which is the same for all records, an action ID, and a section of data that varies according to the action type. All data is kept in clear text and formatted to be readable and editable from a text editor such as vi or notepad.

For more information about the details available in the logs, see Audit log header format on page 369 and Audit log body format on page 370.

Note: For modify commands, two entries are made in the log for resources, calendars, parameters, and prompts. The modify command is displayed in the log as a combination of the delete and add commands.

See the following example of a `.json` file:

```
{"timestamp": "20210917062859", "auditType": "CONMAN", "objectType": "PLWKSTN",
"actionType": "MODIFY", "workstationName": "WA_DWB", "userName": "wauser", "frameworkUser":
"", "objectName": "WA_DWB", "actionDependentContents": "link @;noask"}
```

Storing auditing information in the database (auditStore=db)

This storage location is available when you audit information in the database (**enDbAudit** global option). Choose to store auditing information in the database by setting the **auditStore** global option to `db`. For more information about the **auditStore** global option, see Global options - detailed description on page 27.

The AUDIT_STORE_RECORDS_V table is created in the HCL Workload Automation database.

For more information, see the section about the AUDIT_STORE_RECORDS_V table in *HCL Workload Automation: Database Views*.

Storing auditing information both in the database and in a file (auditStore=both)

This storage location is available when you audit information in the database (**enDbAudit** global option). Choose to store auditing information both in the database and in a file by setting the **auditStore** global option to `both`. For more information about the **auditStore** global option, see Global options - detailed description on page 27.

For details about how the information is stored, see Storing auditing information in the database (auditStore=db) on page 369 and Storing auditing information in a file (auditStore=file) on page 368.

Audit log header format

Each log file starts with a header record that contains information about when the log was created and whether it is a plan or database log.

The header record fields are separated by a colon (:), as follows:

```
HEADER:<GMT_date>:<GMT_time>:<local_date>:<local_time>:<object_type>: >
   <workstation>:<user_ID>:<version>: <level>
```

Log Type

HEADER

GMT Date

The GMT date when the log file was created.

GMT Time

The GMT time when the log file was created.

Local Date

The local date when the log file was created. The local date is defined by the time zone option of the workstation.

Local Time

The local time when the log file was created. The local time is defined by the time zone option of the workstation.

Object Type

DATABASE for a database log file and PLAN for a plan log file.

Workstation Name

The HCL Workload Automation workstation name for which this file was created. Each workstation in the HCL Workload Automation network creates its own log.

User ID

The HCL Workload Automation user ID that created the log file.

Version

The version of the file.

Level

The logging level.

Audit log body format

The audit log formats are basically the same for the plan and the database. The log consists of a timestamp, a series of tags which identify the audit, object, and action type, and data sections that vary with the action type. The data is in clear text format and each data item is separated by a comma (,).

The log file entries are in the following format:

```
"timestamp":"timestamp", "auditType":"audit_type",
"objectType":"object_type", "actionType":"action_type",
"workstationName""workstation_name", "userName": "user_name",
"frameworkUser": "framework_user", "objectName":"object_name"
"actionDependentContents": "action-dependent_fields"
```

The log files contain the following information:

timestamp

Displays the date and time the action was performed in GMT time. The format is *yyyy-mm-dd:hh-mm-ss*.

auditType

Displays an eight-character value indicating the source of the log record. The following log types are supported:

CONMAN

conman command text

DATABASE

Database action

HEADER

The log file header

MAKESEC

makesec run

PARMS

Parameter command text

PLAN

Plan action

RELEASE

release command text

STAGEMAN

stageman run

objectType

Displays the type of the object that was affected by an action, from the following:

DATABASE

Database definition (for header only)

DBCAL

Database calendar definition

DBDOMAIN

Database domain definition

DBJBSTRM

Database Job Scheduler definition

DBJOB

Database job definition

DBPARM

Database parameter definition

DBPROMPT

Database prompt definition

DBRES

Database resource definition

DBSEC

Database security

DBUSER

Database user definition

DBVARTAB

Database variable table definition

DBWKCLS

Database workstation class definition

DBWKSTN

Database workstation definition

PLAN

Plan (for header only)

PLDOMAIN

Plan domain

PLFILE

Plan file

PLJBSTRM

Plan Job Scheduler

PLJOB

Plan job

PLPROMPT

Plan prompt

PLRES

Plan resource

PLWKSTN

Plan workstation

actionType

Displays what action was performed on the object. The appropriate values for this field are dependent on which action is being performed.

For the plan, the "*action_type*" can be ADD, DELETE, MODIFY, or INSTALL.

For the database, the ADD, GET, DELETE and MODIFY actions are recorded for workstation, workstation classes, domains, users, jobs, job streams, calendars, prompts, resources and parameters in the database.

The "**actionType**" field also records the installation of a new Security file. When **makesec** is run, HCL Workload Automation records it as an INSTALL action for a Security definition object.

LIST and DISPLAY actions for objects are not logged.

For parameters, the command line with its arguments is logged.

workstationName

Displays the HCL Workload Automation workstation from which the user is performing the action.

userName

Displays the logon user who performed the particular action. On Windows® operating systems, if the user who installed WebSphere Application Server Liberty was a domain user, for Log Types **stageman** and **conman** this field contains the fully qualified user ID *domain\user*.

frameworkUser

Displays the framework user.

objectName

Displays the fully qualified name of the object. The format of this field depends on the object type as shown here:

DATABASE

N/A

DBCAL

"calendar"

DBDOMAIN

"domain"

DBJBSTRM

"workstation"#"job_stream"

DBJOB

"workstation"#"job"

DBPARM

"workstation"#"parameter"

DBPROMPT

"prompt"

DBRES

"workstation"#"resource"

DBSEC

N/A

DBUSER

[*"workstation"#*]*"user"*

DBVARTAB

"variable_table"

DBWKCLS

"workstation_class"

DBWKSTN

"workstation"

PLAN

N/A

PLDOMAIN

"domain"

PLFILE

"workstation"#"path"(*"qualifier"*)

PLJBSTRM

"workstation"#"job_stream_instance"

PLJOB

"workstation"#"job_stream_instance"."job"

PLPROMPT

[*"workstation"#*]*"prompt"*

PLRES

"workstation"#"resource"

PLWKSTN

"workstation"

actionDependentContents

Displays the action-dependent data fields. The format of this data is dependent on the "*actionType*" field.

Sample audit log entries

This is a sample database audit log:

```
HEADER  |20080207|084124|20080207|094124|DATABASE|      |WK1|         | | |Version=A1.0| Level=1

DATABASE|20080207|084124|20080207|094124|DBRES   |ADD   |WK1|operator1| |res=WK1#RESOURCE    |

DATABASE|20080207|100524|20080207|110524|DBWKSTN |MODIFY|WK1|operator1| |ws=TIVOLI10         |

DATABASE|20080207|100525|20080207|110525|DBWKSTN |MODIFY|WK1|operator1| |ws=ASLUTRI1         |

DATABASE|20080207|100525|20080207|110525|DBWKSTN |MODIFY|WK1|operator1| |ws=WK1              |
DATABASE|20080207|100526|20080207|110526|DBDOMAIN|MODIFY|WK1|operator1| |dom=MASTERDM        |

DATABASE|20080207|100610|20080207|110610|DBWKSTN |MODIFY|WK1|operator1| |ws=TIVOLI10         |

DATABASE|20080207|100610|20080207|110610|DBWKSTN |MODIFY|WK1|operator1| |ws=ASLUTRI1         |

DATABASE|20080207|100611|20080207|110611|DBWKSTN |MODIFY|WK1|operator1| |ws=WK1              |

DATABASE|20080207|100611|20080207|110611|DBWKSTN |ADD   |WK1|operator1| |ws=WK2              |

DATABASE|20080207|100612|20080207|110612|DBDOMAIN|MODIFY|WK1|operator1| |dom=MASTERDM        |
```

This is a sample plan audit log:

```
HEADER  |20080207|100758|20080207|110758|PLAN    |       |WK1|admin| |     |Version=A1.0|Level=1

STAGEMAN|20080207|100758|20080207|110758|PLAN    |INSTALL|WK1|admin| |C:\HCL\TWS\oper1\Symphony|
                     AWSBHV030I The new Symphony file is installed.

STAGEMAN|20080207|100758|20080207|110758|PLAN    |INSTALL|WK1|admin| |C:\HCL\TWS\oper1\Sinfonia|
                  AWSBHV036I Multi-workstation Symphony file copied to C:\HCL\TWS\oper1\Sinfonia

STAGEMAN|20080207|100758|20080207|110758|ADITLEVL|MODIFY |WK1|admin| |                         |
                     AWSBHV077I Audit level changing from 0 to 1.

CONMAN  |20080207|100800|20080207|110800|PLWKSTN |MODIFY |   |admin| |WK1                      |
                     continue & start

CONMAN  |20080207|100941|20080207|110941|PLWKSTN |MODIFY |   |admin| |SLUTRI1                  |
                     limit cpu=slutri1;10

PLAN    |20080207|101018|20080207|111018|PLWKSTN |MODIFY |WK1|oper1| |WK1                      |
                     limit cpu=SLUTRI1;20

PLAN    |20080207|101028|20080207|111028|PLDOMAIN|MODIFY |WK1|oper1| |ECCOLO                   |
                     reply ECCOLO;yes
```

A **ResetPlan** command run against the current production plan is stored in the plan audit log file as follows:

```
STAGEMAN|20080207|100758|20080207|110758|PLAN|DELETE|WK1|admin|
     |/home/WK1/schedlog/M200803140127|
                     AWSBHV025I The old Symphony file renamed /home/WK1/schedlog/M200803140127
```

Enabling audit for authentication events

This topic explains how to enable audit for authentication events.

To enable audit for authentication events, proceed as follows:

1. Go to `<DWC_DATA_dir>/usr/servers/dwcServer/configDropins/overrides`

   Note: On Windows systems, the path is: `<DWC_home>\usr\servers\dwcServer\configDropins\overrides`

2. Create a file named **auditing.xml** and paste the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<server>
<!-- Enable features -->
<featureManager>
<feature>audit-1.0</feature>
</featureManager>

<!-- Enabling Audit Filehandler-->>
<auditFileHandler maxFiles="100" maxFileSize="10" compact="true">
<events name="AuditEvent_1" eventName="SECURITY_AUTHN" outcome="SUCCESS"/>
<events name="AuditEvent_2" eventName="SECURITY_AUTHN" outcome="REDIRECT"/>
<events name="AuditEvent_3" eventName="SECURITY_AUTHN" outcome="DENIED"/>
</auditFileHandler>
</server>
```

3. Stop and start WebSphere Application Server Liberty. For further information about the stop and start commands, see Application server - starting and stopping on page 428.
4. Access the Dynamic Workload Console.

You successfully enabled the audit for authentication events. To see the authentication logs, open the **audit.log** file located at the following path:

On Windows operating systems

`TWA_home>\stdlist\appserver\dwcServer\logs`

On UNIX operating systems

`<TWA_DATA_DIR>/stdlist/appserver/dwcServer/logs`

For further information about audit logs, see WebSphere Application Server Liberty documentation.

Keeping track of database changes using audit reports

To keep always track of the changes that impact objects stored in the database, you can use the following audit reports, which can be run in batch mode using the command line interface:

General audit report

The report provides information about objects that have been modified in the database. More specifically, it details who made the change, on which objects, and when.

Details report

The report provides further details about the changes implemented. It specifies who made the change, on which objects, when, and what has been changed. More specifically it shows the object definition before and after the change.

You can run these reports on DB2 and Oracle databases.

A sample business scenario

The administrator of an insurance company needs to keep track of all the changes impacting the insurance policies, conditions and terms of all the customers registered in the company database. To do it, the administrator periodically runs the audit general and details reports.

To satisfy this request, he creates an audit general report that provides details about which TWS objects have been modified in the database, who modified them and on which date. Then, to find out more details about the changes, he also creates an audit details report.

To accomplish his task, he runs the following steps:

1. He customizes the property files related to the audit reports, specifying the format and content of the report output.
2. He schedules jobs to obtain the reports:
    a. The first job generates an audit to be saved locally.
    b. The second job runs a detail report overnight to retrieve more details about the specific changes implemented. The report output is sent using an mail to the analyst. The information collected is used to keep all the insurance branch offices updated with any change and news.
3. The administrator adds the two jobs to a job stream scheduled to run weekly and generates the plan.

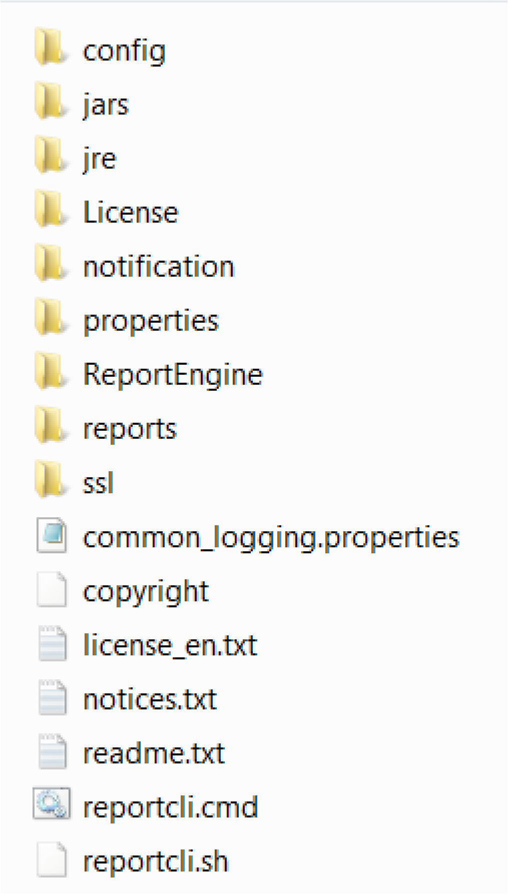
Setting up for command line audit reporting

About this task

Before running these reports you must perform a few setup steps:

1. The software needed to run these reports is contained in a package named `TWSBatchReportCli` included in the HCL Workload Automation installation image, in the `TWSBatchReportCli` directory. If you plan to run them from within a scheduled job, extract the package file on one of the operating systems listed at Dynamic Workload Console Detailed System Requirements.

   After extracting the package, you obtain the following file structure:



config
jars
jre
License
notification
properties
ReportEngine
reports
ssl
common_logging.properties
copyright
license_en.txt
notices.txt
readme.txt
reportcli.cmd
reportcli.sh

Because the native UNIX™ tar utility does not support long file names, if you are extracting the files on AIX® systems, ensure that the latest GNU version of tar (gtar) is installed to extract the files successfully.

Note:

a. Make sure you run the following commands in the directory where you extracted the files:

   On UNIX™

   ```
   chmod -R +x *
   chown -R username *
   ```

   On Windows™

   Ensure HCL Workload Automation is installed.

   ```
   setown -u username *
   ```

   Where *username* is the HCL Workload Automation user that will run the reports.

b. If you plan to schedule jobs that run these reports, the system where you extract the package must be accessible as network file system from a fault-tolerant agent defined in the local scheduling environment.

2. If you use an Oracle database, download the JDBC drivers required by your Oracle server version.

3. Copy the JDBC drivers in the `report_cli_installation_dir\jars` directory and in `report_cli_installation_dir\ReportEngine\plugins\org.eclipse.birt.report.data.oda.jdbc_4.2.1.v20120820\drivers` directory. The report cli automatically discovers the two jar files.
4. Configure the template file `.\config\common.properties` by specifying the following information.
    a. If you use an Oracle database, connect to the database where the historical data are stored as follows:
        i. Retrieve the location of the Oracle JDBC drivers. This information is stored in the **com.ibm.tws.webui.oracleJdbcURL** property in the `TWSConfig.properties` file, located in

            On Windows operating systems

            <TWA_home>\usr\servers\engineServer\resources\properties

            On UNIX operating systems

            <*TWA_DATA_DIR*>/usr/servers/engineServer/resources/properties

            For more information about this file, see Configuring for an Oracle database on page 177.
        ii. Specify the location of the Oracle JDBC drivers in the **PARAM_DataSourceUrl** property in the `common.properties` file.

        No customization is required if you use DB2.
    b. Set the date and time format, including the time zone. The file `.\config\timezone.txt` contains a list of time zones supported by HCL Workload Automation and the information on how to set them. The time zone names are case sensitive.
    c. Make the report output available on the URL specified in **ContextRootUrl** field. This is an example of the configuration settings:

```
######################################################################
# HTTP Server information
######################################################################

 #Specify the context root where the report will be available
 #To leverage this possibility it needs to specify in the report output dir
 #the directory that is referred by your HTTP Server with this contect root

 ContextRootUrl=http://myserver/reportoutput
```

    In this case, en sure that the *output_report_dir* specified when running the reports command points to the same directory specified in the **ContextRootUrl**.
    d. Send the report output using a mail. This is an example of the configuration settings:

```
######################################################################
# Email Server configuration
######################################################################
 PARAM_SendReportByEmail=true

 #SMTP server
 mail.smtp.host=myhost.mydomain.com
 #IMAP provider
 mail.imap.socketFactory.fallback=false
 mail.imap.port=993
 mail.imap.socketFactory.port=993
 #POP3 provider
```

```
mail.pop3.socketFactory.fallback=false
mail.pop3.port=995
mail.pop3.socketFactory.port=995

######################################################################
# Email properties
######################################################################
PARAM_EmailFrom=user1@your_company.com
PARAM_EmailTo=user2@your_company.com,user3@your_company.com
PARAM_EmailCC=user4@your_company.com
PARAM_EmailBCC=user5@your_company.com
PARAM_EmailSubject=Test send report by email
PARAM_EmailBody=This is the report attached
```

An explanation of all the customizable fields is contained in the template file.

Running audit reports from the command line

To run audit report on the database, you must first enable the audit feature and configure the audit options described in Global options - detailed description on page 27.

The `\reports\templates` directory contains a sample template file for each type of report.

Before running any of these reports, ensure that you customize the corresponding template file, either `ad.properties` or `ag.properties`.

In that file, named `*report_name*.properties`, you can specify:

- The information to display in the report header.
- How to filter the information to display the expected result.
- The format and content of the report output.

For more information about the specific settings see the explanation provided in the template file beside each field.

After you set up the environment as it is described in Setting up for command line audit reporting on page 377, and you configured report template file, use the following syntax to run the report:

reportcli -p *report_name.property*
[**-o** *output_report_dir*]
[**-r** *report_output_name*]
[**-k** key=*value*]
[**-k** key=*value*]
....

where:

-p *report_name.property*

Specifies the path name to the report template file.

-o *routput_report_dir*

Specifies the output directory for the report output.

-r *report_output_name*

Specifies the name of the report output.

-k key=*value*

Specifies the value of a settings. This value override the corresponding value, if defined, in the `common.properties` file or in the `report_name.properties` file.

Examples

1. In this example the `reportcli.cmd` is run with the default parameter:

   ```
   reportcli.cmd -p D:\ReportCLI\TWSReportCli\reports\templates\ag.properties
   -r audit1
   ```

2. In this example the `reportcli.cmd` is run using the `-k` parameter to override the values set for **PARAM_DateFormat** in the `.\config\common.properties` file:

   ```
   reportcli.cmd -p D:\ReportCLI\TWSReportCli\reports\templates\ag.properties
   -r audit2 -k PARAM_DateFormat=short
   ```

3. In this example the `reportcli.cmd` is run using the `-k` parameter to override he format specified for the report output in the `.properties` file:

   ```
   ./reportcli.sh -p /TWSReportCli/REPCLI/reports/templates/wwr.properties
   -r audit3 -k REPORT_OUTPUT_FORMAT=html -k OutputView=charts
   ```

Note: If the report is run through a HCL Workload Automation job, the output of the command is displayed in the job output.

Collecting job metrics

You can run the following SQL queries on the Workload Scheduler data base to retrieve the number of jobs run by HCL Workload Automation over a period of time. One query determines the number of jobs run by specific workstations, while the other query determines the number of jobs run on the entire HCL Workload Automation domain. You can run the queries from the command line interface of your database or you can add them in the Dynamic Workload Console to create your custom SQL reports, as described in section creating a task to create custom SQL reports in *Dynamic Workload Console User's Guide*.

Job metrics queries for DB2

Use the following SQL query to find the number of jobs run on specific workstations:

```
SELECT year(job_run_date_time) AS Year, month(job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM mdl.job_history_v
WHERE workstation_name IN ('WKS_1', 'WKS_2', 'WKS_N')

GROUP BY year(job_run_date_time), month(job_run_date_time)
```

where 'WKS_1', 'WKS_2', 'WKS_N' are the names of the workstations that ran the jobs you want counted.

Use the following SQL query to find the number of jobs run on the entire HCL Workload Automation domain:

```
SELECT year(job_run_date_time) AS Year, month(job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM mdl.job_history_v
GROUP BY year(job_run_date_time), month(job_run_date_time)
```

Job metrics queries for DB2 for zOS

Use the following SQL query to find the number of jobs run on specific workstations:

```
SELECT year(job_run_date_time) AS Year, month(job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM mdl.job_history_v
WHERE workstation_name IN ('WKS_1', 'WKS_2', 'WKS_N')
GROUP BY year(job_run_date_time), month(job_run_date_time)
```

where 'WKS_1', 'WKS_2', 'WKS_N' are the names of the workstations that ran the jobs you want counted.

Use the following SQL query to find the number of jobs run on the entire HCL Workload Automation domain:

```
SELECT year(job_run_date_time) AS Year, month(job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM mdl.job_history_v
GROUP BY year(job_run_date_time), month(job_run_date_time)
```

Job metrics queries for Oracle database

Use the following SQL query to find the number of jobs run on specific workstations:

```
SELECT EXTRACT(year FROM job_run_date_time) AS Year,
EXTRACT(month FROM job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM job_history_v
WHERE workstation_name IN ('WKS_1', 'WKS_2', 'WKS_N')
or (workstation_name = '-' and JOB_STREAM_WKS_NAME_IN_RUN in('WKS_1', 'WKS_2', 'WKS_N'))
GROUP BY EXTRACT(year FROM job_run_date_time), EXTRACT(month FROM job_run_date_time);
```

where 'WKS_1', 'WKS_2', 'WKS_N' are the names of the workstations that ran the jobs you want counted.

Use the following SQL query to find the number of jobs run on the entire HCL Workload Automation domain:

```
SELECT EXTRACT(year FROM job_run_date_time) AS Year,
EXTRACT(month FROM job_run_date_time) AS Month,
cast (count(job_run_date_time) AS INT) AS JobNbr FROM job_history_v
GROUP BY EXTRACT(year FROM job_run_date_time), EXTRACT(month FROM job_run_date_time);
```

Chapter 8. Administrative tasks

This chapter describes how to perform some specific administrative tasks on HCL Workload Automation, as follows:

The tasks

Switching a domain manager on page 384

Change a domain manager or dynamic domain manager, either in the event of the failure of the computer where it is installed, or as part of a planned replacement activity.

Switching the master to a backup on page 390

Change a master domain manager, either in the event of the failure of the computer where it is installed, or as part of a planned replacement activity.

Changing key HCL Workload Automation passwords on page 405

Change the password of the TWS_user, or any other of the users that have an infrastructure role in HCL Workload Automation.

Unlinking and stopping HCL Workload Automation on page 411

The correct procedure to unlink the master domain manager from its agents and stop the master processing.

Changing the properties for the database on page 412

If you need to change the host, port or name of the database, effect the change in the application server, where the data source configuration is maintained.

Changing the workstation host name or IP address on page 414

Change the host name or IP address of a workstation.

Changing the security settings on page 418

If you need to update the properties that define your SSL connection or authentication mechanism, you need to make the changes in the WebSphere Application Server Liberty.

Managing the event processor on page 419

If you are using event-driven workload automation, you will need to perform periodic maintenance on the event processor.

Application server tasks

The following tasks might need to be performed on the application server:

Application server - starting and stopping on page 428

How to stop and start the application server when you need to.

Application server - automatic restart after failure on page 429

The application server is managed by a utility that restarts it if it stops for any reason (subject to a configurable policy). This section describes how to modify the policy and deal with any situations that the policy cannot handle.

Application server - encrypting the profile properties files on page 432

Several of the application server configuration files contain passwords. To avoid that these remain in the files in plain text, run a utility to encrypt them.

Application server - configuration files backup and restore on page 432

The application server configuration manages the data source and security aspects of your HCL Workload Automation environment. The files should be regularly backed up and when necessary can be restored.

Application server - changing the host name or TCP/IP ports on page 432

If you need to change the host or ports used by the application server, follow the correct procedure.

Application server - changing the trace properties on page 433

The application server has a trace facility. This section describes how to increase the trace level to obtain more information for troubleshooting, and how to reduce the level to improve performance.

Changing the application server properties

Several of the above tasks require you to run a common procedure based on templates whereby you:

1. Copy the template file from the `templates` folder to a working folder.
2. Edit the template file in the working folder with the desired configuration.
3. Optionally, create a backup copy of the relevant configuration file present in the `overrides` directory in a different directory. Ensure you do not copy the backup file in the path where the template files are located.
4. Copy the updated template file to the `overrides` folder. Maintaining the original folder structure is not required.
5. Changes are effective immediately.

This procedure is fully described in Configuring HCL Workload Automation using templates on page 422.

Switching a domain manager

About this task

Being prepared for network problems makes recovery easier. Set up a backup domain manager for each domain manager in your network to more easily ensure that HCL Workload Automation peak job scheduling loads are met. Choose any fault-tolerant agent in the domain to be a backup domain manager.

A domain manager might need to be changed because you want it to run on a different workstation, or it might be forced on you as the result of network linking problems or the failure of the domain manager workstation itself. This section, and its subsections, describes how to prepare for and use a backup domain manager. However, if the domain manager to be changed is a master domain manager or dynamic domain manager, there are some specific additional steps to perform; see Switching the master to a backup on page 390.

Running without a domain manager has the following effects:

- Agents and subordinate domain managers cannot resolve inter-workstation dependencies, because activity records broadcast by the master domain manager are not being received.
- The upward flow of events is interrupted. This impacts events that report the status of jobs, job streams and dependencies defined on workstations in the HCL Workload Automation network hierarchy under the failed domain manager.
- Standard agents that are hosted by the failed domain manager cannot perform any processing, since they depend on the domain manager for all scheduling and job launching.

If the problem is expected to be of short duration, you can wait for the problem to be resolved and HCL Workload Automation will recover on its own, as described in the *Troubleshooting Guide* in the section about network linking problems. If you are uncertain about the duration, or if you want to restore normal agent operation, you must switch to a backup, as described in the following sections.

Ensure that the *FullStatus* mode is selected in the backup workstation definition. For more information about workstation properties, see the section about workstation definition in *User's Guide and Reference*.

Also ensure that the backup domain manager is synchronized with respect to time with the domain manager. The most secure way is to use a Network Time Protocol Server to control the time on both systems, with the same repeat interval.

Network security is enforced using IP address validation. As a consequence, workstation linking (autolink option or link command) might fail if an agent has an old `Symphony` file that does not contain the new domain manager. If a connection fails, remove the old `Symphony` file on the agent and retry the connection.

For more information about the autolink option, see the section about workstation definition in *User's Guide and Reference*.

For more information about the link command, see the section about the link command in *User's Guide and Reference*.

Simplified procedure for switching a domain manager

Use one of these procedures when you have a short-term loss of a domain manager.

Using the command line

See the procedure described under the switchmgr command in *User's Guide and Reference*.

Using the Dynamic Workload Console

1. In the navigation bar at the top, click **Monitoring and Reporting > Orchestration Monitor**.
2. Select an engine.
3. From the drop-down menu, select **Workstation**.
4. From the **Query** drop-down list, select a query to monitor workstations.

5. Click **Run** to run the monitoring task.
6. From the table containing the list of workstations, select a workstation and click **More Actions > Become Master Domain Manager**.

Domain managers remain switched until you perform another switch manager operation, or run JnextPlan. To return to the original domain manager without running JnextPlan, repeat this procedure.

Here is the procedure to follow every time you switch the master domain manager or dynamic domain manager if you run dynamic scheduling in your network:

1. Set the job fence to **go** priority level. For further details, see the fence command in *User's Guide and Reference*.
2. Switch the master domain manager or dynamic domain manager to a backup workstation. Use either the `conman switchmgr` command or the Dynamic Workload Console. For more information about both methods, see the procedure described under the switchmgr command in *User's Guide and Reference*.
3. Once the switch has been performed, restore the job fence to zero. For further details, see the fence command in *User's Guide and Reference*.

Complete procedure for switching a domain manager

This section summarizes the steps required to replace a running domain manager with its backup and to complete the procedure by restoring the original domain manager to its function. Follow these steps to make sure that no overlapping problems arise with obsolete versions of the Symphony file. You can also follow these steps to switch a master domain manager or a dynamic domain manager. The steps are documented for four scenarios:

Planned outage

The domain manager is replaced with its backup for planned maintenance work (for example, an upgrade of the operating system).

Unplanned outage

The domain manager is replaced with its backup because of an unexpected failure or malfunction.

Short-term

The domain manager is expected to return to service before the next new production period turnover (run of the JnextPlan job).

Long-term

The domain manager is not expected to return to service before the next new production period turnover (run of the JnextPlan job).

Table 66. Complete procedure for switching a domain manager in case of a planned outage.

Planned outage	
Short-term	**Long-term**
1. Switch the domain manager to a backup workstation. Use either the `conman switchmgr` command or the	1. Switch the domain manager to a backup workstation. Use either the `conman switchmgr` command or the Dynamic Workload

Table 66. Complete procedure for switching a domain manager in case of a planned outage. (continued)

Planned outage	
Dynamic Workload Console. For more information about both methods, see the `Switching a master domain manager or dynamic domain manager` chapter in the Administration Guide. 2. Check that the message boxes for the domain manager undergoing maintenance are large enough not to fill up before it is restored. Increase their size if necessary.	Console. For more information about both methods, see the `Switching a master domain manager or dynamic domain manager` chapter in the Administration Guide. 2. Check that the message boxes for the domain manager undergoing maintenance are large enough not to fill up before it is restored. Increase their size if necessary.
3. Shut down HCL Workload Automation processing on the domain manager undergoing maintenance.	3. Shut down HCL Workload Automation processing on the original domain manager undergoing maintenance.
	4. In the HCL Workload Automation database assign the role of domain manager to the backup workstation. This is done by changing the workstation type in the database from MANAGER to FTA on the original domain manager and from FTA to MANAGER on the backup.
	5. Set the workstation running the original domain manager to `ignore`, using either the `composer mod cpu <`*`workstation_name`*`>` command or the Dynamic Workload Console.
	6. Run JnextPlan to generate the new production plan so that the backup master domain manager is removed from the plan.
When ready to restore the ownership of the domain to the original domain manager:	**When ready to restore the ownership of the domain to the original domain manager:**
	7. Remove the `ignore` flag from the workstation running the original domain manager.
	8. Run JnextPlan to generate the new production plan so that the backup master domain manager is reinserted in the plan.
4. Switch from the backup workstation to the domain manager using one of the methods indicated in step 1.	9. Reassign ownership of the domain to the original domain manager in the HCL Workload Automation database. This is done by changing the workstation type in the database from MANAGER to FTA on the original backup and from FTA to MANAGER on the original domain manager 10. **Optional**: In the original domain manager, remove the `conman start` command from the init procedure and delete any existing copies of the Symphony, `Sinfonia`, and message box files.
5. Link the domain manager from the master to download a fresh version of the Symphony file.	

Table 66. Complete procedure for switching a domain manager in case of a planned outage. (continued)

Planned outage	
	Note: This step is recommended to avoid that any outdated symphony present in the computer is automatically triggered at the first startup. You can add `conman start` again later.
	11. Switch from the backup workstation to the domain manager using one of the methods indicated in step 1.
	12. Link the domain manager from the master to download a fresh version of the Symphony file.

Table 67. Complete procedure for switching a domain manager after an unplanned outage.

Unplanned outage	
Short-term	**Long-term**
1. Switch the domain manager to a backup workstation. Use either the `conman switchmgr` command or the Dynamic Workload Console. For more information about both methods, see the `switchmgr` command in *User's Guide and Reference*. 2. Check that the message boxes for the failing domain manager are large enough not to fill up before it is restored. Increase their size if necessary.	1. Switch the domain manager to a backup workstation. Use either the `conman switchmgr` command or the Dynamic Workload Console. For more information about both methods, see the `switchmgr` command in *User's Guide and Reference*. 2. Check that the message boxes for the failing domain manager are large enough not to fill up before it is restored. Increase their size if necessary.
	3. In the HCL Workload Automation database assign the role of domain manager to the backup workstation. This is done by changing the workstation type in the database from MANAGER to FTA on the original domain manager and from FTA to MANAGER on the backup.
	4. Set the workstation running the failing domain manager to `ignore`, using either the `composer mod cpu <workstation_name>` command or the Dynamic Workload Console.
	5. Run JnextPlan to generate the new production plan so that the backup master domain manager is removed from the plan.
When ready to restore the ownership of the domain to the original domain manager:	**When ready to restore the ownership of the domain to the original domain manager:**

Table 67. Complete procedure for switching a domain manager after an unplanned outage. (continued)

Unplanned outage	
3. **Optional**: - A. In the original domain manager, remove the `conman start` command from the init procedure and delete any existing copies of the Symphony, `Sinfonia`, and message box files. **Note:** This step is recommended to avoid that any outdated symphony present in the computer is automatically triggered at the first startup. You can add `conman start` again later. B. After an unplanned outage, the fault-tolerant agent needs a new Symphony file. Perform the following steps on the current master domain manager: ◦ i. Verify that the current master domain manager is linked to all agents except the old master domain manager. ◦ ii. Shut down all HCL Workload Automation processes (unlink from all agents). ◦ iii. Rename `Sinfonia` as `Sinfonia.orig`. ◦ iv. Copy `Symphony` to `Sinfonia`. You now have identical `Symphony` and `Sinfonia` files.	6. Remove the `ignore` flag from the workstation running the original domain manager.
	7. Run JnextPlan to generate the new production plan so that the backup master domain manager is reinserted in the plan.
4. Switch from the backup workstation to the domain manager using one of the methods indicated in step 1.	8. Reassign ownership of the domain to the original domain manager in the HCL Workload Automation database. This is done by changing the workstation type in the database from MANAGER to FTA on the original domain manager and from FTA to MANAGER on the backup.
5. Link the domain manager from the master to download a fresh version of the Symphony file.	

Table 67. Complete procedure for switching a domain manager after an unplanned outage. (continued)

Unplanned outage	
	9. **Optional:** In the original domain manager, remove the `conman start` command from the init procedure and delete any existing copies of the Symphony, `Sinfonia`, and message box files. **Note:** This step is recommended to avoid that any outdated symphony present in the computer is automatically triggered at the first startup. You can add `conman start` again later.
	10. Switch from the backup workstation to the domain manager using one of the methods indicated in step 1 on page 388.
	11. Link the domain manager from the master to download a fresh version of the Symphony file.

Switching the master to a backup

About this task

A backup workstation for the master domain manager and another workstation for the dynamic domain manager are an essential asset to ensure business continuity and data integrity in your environment.

There are two ways in which the switchover to a backup master domain manager can occur:

A manual, planned switchover procedure

You can switch the master domain manager to a backup master domain manager at any time, either for a short term or for a long term (the original master is not expected to return to service before the next new production period turnover), using the switchmgr command.

An automatic failover process

Starting with version 9.5 Fix Pack 2, you can rely on the automatic failover feature, where, given a list of available backups, the workload is switched over to the backup. See Automatic failover on page 393 for more information.

When selecting a workstation to be a backup, the same rules apply to both the automatic failover and the manual switching of the master. Backup workstations must have compatible operating systems with the master and the backup master domain manager must be installed on a system that is not currently defined in the workload scheduling network. For more information about these topics see Selecting a workstation for the backup master domain manager on page 391 and Changing an agent to become a backup master domain manager on page 391.

In the following topics you can find information about how to enable the automatic failover process, and how to manually switch a master domain manager, and a dynamic domain manager for a Z controller.

If you lose or want to plan to change a master domain manager or dynamic domain manager, the same comments in the section Switching a domain manager on page 384 apply, but in addition, consider the sub-topics in this section.

Selecting a workstation for the backup master domain manager

It is the normal process to install a backup master domain manager when you set up your scheduling network. However, if you have not done so, and decide later that you need a backup master domain manager, you have two options:

- Install a backup master domain manager on a system that is not currently in the workload scheduling network. For the detailed procedure, see the section about installing the master domain manager and backup master domain manager*Planning and Installation Guide*.
- Promote an agent to backup master domain manager. This option is time-consuming and requires you to interrupt your workload scheduling activities, but if you want to do it, follow the procedure described in this section.

Regardless of the option you choose, the following are some prerequisites to consider for the backup workstation:

- Choose compatible operating systems. Since you must transfer files between the master domain manager and its backup, the workstations must have compatible operating systems. Do not combine UNIX™ with Windows™ workstations, and in UNIX™, do not combine big-endian workstations (AIX®) with little-endian workstations (most Intel™-based operating systems, including Windows™ and Linux™).

  See the HCL Workload Automation Detailed System Requirements for details of the prerequisite requirements of a backup master domain manager.
- Ensure the master domain manager and the backup master domain manager have `FullStatus` turned on in the workstation definition. See Setting up a backup master domain manager on page 392
- Copy any necessary files, such as, the security file and localopts file, to the backup workstation. See Copying files to use on the backup master domain manager on page 392.

Changing an agent to become a backup master domain manager

You *cannot* change an agent to become a backup master domain manager, using a command or procedure that allows continuity of workload scheduling activities.

Instead, if you need to change an agent workstation to become the backup master domain manager, you must interrupt the workload scheduling activities. The procedure is as follows:

1. Check that the workstation satisfies the prerequisites for a backup master domain manager. See HCL Workload Automation Detailed System Requirements for more information.
2. If it does, stop and disable all workload scheduling operations on the workstation
3. Uninstall the agent, following the instructions in the section about uninstalling agents in *Planning and Installation Guide*.
4. Install the backup master domain manager on the system where the agent was installed, following the instructions in the section about installing the master domain manager and backup master domain manager in *Planning and Installation Guide*.

5. Ensure that the database entry for the workstation is correct for a backup master domain manager. See the section about workstation definition in *User's Guide and Reference* for information about the workstation definition
6. Define and start any workload scheduling operations you require on the workstation in its new role.

Setting up a backup master domain manager

Ensure that the master domain manager and the backup master domain manager have *FullStatus* turned on in the workstation definition. This is important if you need to resort to long-term recovery, where the backup master domain manager generates a `Symphony` file (runs JnextPlan). If *FullStatus* is not turned on, the former master domain manager shows up as a regular fault-tolerant agent after the first occurrence of JnextPlan. During normal operations, the JnextPlan job automatically turns on the *FullStatus* flag for the master domain manager, if it is not already turned on. When the new master domain manager runs JnextPlan, it does not recognize the former master domain manager as a backup master domain manager unless the flag is enabled. The former master domain manager does not have an accurate `Symphony` file when the time comes to switch back. For more information about workstation properties, see the section about workstation definition in *User's Guide and Reference*.

Also ensure that the backup master domain manager is synchronized with respect to time with the master domain manager. The securest way is to use a Network Time Protocol Server to control the time on both systems, with the same repeat interval.

Copying files to use on the backup master domain manager

To back up the important master domain manager files to the backup master domain manager, use the following procedure:

1. Copy the `Security` file from the master domain manager to backup domain manager in the following path:

   On Windows operating systems

   <*TWA_home*>`\TWS`

   On UNIX operating systems

   TWA_DATA_DIR

   Add a suffix to the file so that it does not overwrite the Security file on the backup domain manager, for example, `Security_from_MDM`.
2. Copy all files in the following path:

   On Windows operating systems

   <*TWA_home*>`/TWS/mozart`

   On UNIX operating systems

   TWA_DATA_DIR`/mozart`
3. Copy the `localopts` file (see Setting local options on page 48 for the location). Add a suffix to the file so that it does not overwrite the `localopts` file on the backup master domain manager; for example, `localopts_from_MDM`.

This procedure must be performed each production period, or whenever there are significant changes to any objects. It can be incorporated into a script.

In addition to these required files, you might also want to copy the following:

- Any scripts you might have written.
- Archived Symphony files, for reference.
- Log files, for reference.

Note: Another approach could be to place all of the above files on a separately mountable file system, that could easily be unmounted from the master domain manager and mounted on the backup master domain manager in the event of need. You would almost certainly want to backup these files in addition, to protect against loss of the separately mountable file system.

To prevent the loss of messages caused by a master domain manager error, you can use the fault-tolerant switch-manager facility.

Automatic failover

Switching a master domain manager to a backup master domain manager.

Recovery is easy when you are prepared for potential problems. If the master domain manager becomes unavailable, to ensure continuous operations, a long-term switchmgr operation is triggered and the workload is automatically switched to an eligible backup master domain manager.

Similarly, the backup event processors automatically detect if the event processor is unavailable, and a long-term switcheventprocessor command is triggered.

This is the default behavior for a complete fresh installation of V9.5 Fix Pack 2 or later, but it can be enabled for back-level environments that are upgraded to V9.5 Fix Pack 2 or later. For more information, see Enabling automatic failover on page 395.

Note: If you perform a fresh installation of a backup master domain manager at the V9.5 Fix Pack level in an existing back-level environment, the automatic failover feature is disabled. To enable it, follow this procedure. The feature is enabled by default for only a complete fresh installation.

You can optionally define potential backups for both the master domain manager and the event processor in two separate lists, adding preferential backups at the top of the lists. The backup engines monitor the behavior of the master domain manager and event processor to detect anomalous behavior and then attempt to recover. Each component plays a role in either detecting a failure or recovering from it:

- Each backup master domain manager monitors the status of the active master domain manager.
- The master domain manager (active or backup) is made to be self-aware. It monitors the status of its fault-tolerant agent to check on the status of processes such as, Batchman, Mailman and Jobman. If at least one of these processes are down, the master domain manager makes 3 attempts to restart them. If the 3 attempts fail, a long-term switchmgr operation is triggered and the workload is automatically switched to an eligible backup master domain manager, while the event processor remains unchanged.

- If the WebSphere Application Server Liberty goes down, the watchdog process attempts to restart it. If the attempt fails, long-term switchmgr and switcheventprocessor commands are triggered, moving both the master domain manager and event processor to their backups.
- If the active master domain manager cannot be automatically restored within 5 minutes (the threshold after which the master is declared unavailable), then a permanent switch to a backup is automatically triggered by any of the backup candidates when one or more of the following conditions persist:
  - The fault-tolerant agent, WebSphere Application Server Liberty, or both are still down.
  - The engine is unable to communicate with the database, for example, due to a network outage.

  If you have defined potential backups in a list, and a switch after 5 minutes is not possible with the first backup in the list because it is unavailable, then an attempt is made to contact the remaining backups in the list, following the order specified in the list, until an available backup is found to perform the switch. In this case, 5 minutes pass between each attempt.

The list for potential event processor backups is a separate list from the potential master domain manager backups, because you might have a workstation that can serve as the event manager backup, but you do not want it to act as a potential master domain manager backup. If the event manager fails, but the master domain manager is running fine, then only the event manager switches to a backup manager defined in the list of potential backups. The same happens if the master domain manager fails and the event manager is running fine.

You can track detected failures and the actions taken by checking the `messages.log` file located in the path:

- <*TWA_DATA_DIR*>`/stdlist/appserver/engineServer/logs/messages.log`
- <*TWA_home*>`\TWS\stdlist\appserver\engineServer\logs\messages.log`

Note: On Linux® and UNIX®, for a fresh installation, an extended agent is installed with the master domain manager which is used to communicate where to run the FINAL job stream, along with its jobs. With an extended agent, $MASTER can be used to indicate that the agent's host workstation is the master domain manager. If the role of the master is switched to a backup, then the new master is represented by $MASTER. This supports both a short-term and long-term switch for the automatic failover feature. If you are upgrading from a version earlier than 9.5 Fix Pack 2, then you must define the extended agent manually.

On Windows™ workstations, the FINAL job stream is not defined on the extended agent, but remains on the master domain manager. The FINAL and FINALPOSTREPORTS job streams and jobs need to be moved from the master to the extended agent workstation. For this reason, only a short-term switch can be performed automatically and the long-term switch must be performed manually as documented in Extended loss or permanent change of master domain manager on page 400 and in Complete procedure for switching a domain manager on page 386. See also the switchmgr command in the *User's Guide and Reference* that contains both the command-line syntax, as well as the procedural steps to perform the switch from the Dynamic Workload Console.

Enabling automatic failover

To enable automatic failover, configure one or more backup engines, and set the related global options (**workstationEventMgrListInAutomaticFailover** and **workstationMasterListInAutomaticFailover**) using the optman command, so that when the active master becomes unavailable, a long-term switchmgr operation is triggered.

Before you begin

Ensure that the master domain manager and the back master domain managers were installed using the same user (UID) and group (GID).

About this task

If you performed an upgrade from Version 9.5 or 9.5 Fix Pack 1, the automatic failover feature is disabled, but it can be enabled following a few simple steps outlined in this task. Automatic failover is, instead, enabled by default for a fresh installation of Version 9.5 Fix Pack 2 and later, and any backup master domain manager installed and configured with Fix Pack 2 is an eligible backup. If you subsequently disabled this feature, you can use the following procedure to re-enable it. You can also use this procedure to define a list of preferred eligible backups, excluding any backups you do not want to consider as an eligible backup. If an eligible workstation is defined in a folder, use the composer li ws @;showid command to retrieve the ID of the workstation you plan to define as backup. You can also configure a separate list of potential backups for the event processor.

1. Ensure the local option, mm resolve master, in the localopts file, is set to `no` on both the master domain manager and on all eligible backup master domain managers.
2. Optional. Define a list of potential backups for the master domain manager and the event manager.
    a. Update the global option, workstationMasterListInAutomaticFailover, on the master domain manager to specify a list of workstations to be considered as eligible backups for the master domain manager. Edit the value of this option by adding a list of workstations, separated by commas, starting with your preferred choices at the top of the list. The list includes the current master domain manager. If no workstations are specified in this list, then the first backup master domain manager to detect that the master is down, performs the switch.
    b. Specify potential backups for the event processor by editing the value for the workstationEventMgrListInAutomaticFailover global option. Add a list of workstations, separated by commas, starting with your preferred choices at the top of the list. The list includes the current event manager workstation.
3. Set the following global options to "`yes`": enAutomaticFailover | af and enAutomaticFailoverActions | aa using the optman chg command. For example:

```
optman chg af=yes
optman chg aa=yes
```

4. Restart WebSphere Application Server Liberty

Important: Complete the remaining steps only if they are not already present in your environment.

5. If not already present on the master domain manager, create a new workstation with the following specifications:
    - Type: `Extended Agent`
    - Access method: `unixlocl`
    - Host: `$MASTER`

   For example, if you create a workstation named, MDM_XA, with these specifications, the following is the workstation definition:

   ```
   CPUNAME MDM_XA
     DESCRIPTION "Workload Scheduler Virtual Master"
     OS OTHER
     NODE mdm_xa TCPADDR 31111
     FOR MAESTRO HOST $MASTER ACCESS "unixlocl"
       TYPE X-AGENT
       AUTOLINK OFF
       BEHINDFIREWALL OFF
       FULLSTATUS OFF
   END
   ```

6. Set the FINAL and FINALPOSTREPORTS job streams on the master domain manager to "`draft`". Draft job streams are not added to the preproduction plan.

   ```
   composer mod jS=FINAL
   composer mod js=FINALPOSTREPORTS
   ```

   For example, the following is an extract from the definition for the FINAL job stream:

   ```
   SCHEDULE MDM#FINAL
   DESCRIPTION "Added by composer."
   DRAFT
   ON RUNCYCLE RC1 "FREQ=DAILY;"
   AT 2359
   CARRYFORWARD
   FOLLOWS MDM#FINAL.SWITCHPLAN PREVIOUS
   :
   ```

   The following example is an extract from the definition for the FINALPOSTREPORTS job stream:

   ```
   SCHEDULE MDM#FINALPOSTREPORTS
   DESCRIPTION "Added by composer."
   DRAFT
   ON RUNCYCLE RC1 "FREQ=DAILY;"
   SCHEDTIME 2359
   CARRYFORWARD
   FOLLOWS MDM_XA#FINAL.SWITCHPLAN PREVIOUS
   :
   ```

7. If not already present, make the following changes to the `Sfinal` file:

   a. Create a backup of the `Sfinal` file. For example:

   ```
   cp  /<TWA_home>/TWS/Sfinal   /<TWA_home>/TWS/Sfinal.orig
   ```

   b. Add the new extended agent workstation to the FINAL and FINALPOSTREPORTS job stream definitions.

c. Substitute the SCRIPTNAME keyword with DOCOMMAND in all of the jobs defined in the FINAL and FINALPOSTREPORTS job streams.

d. Ensure the path to the scripts launched by the jobs in the FINAL and FINALPOSTREPORTS job streams use the variable, *UNISONHOME*.

e. Submit the `composer add Sfinal` command to generate the FINAL and FINALPOSTREPORTS job streams on the new extended agent workstation if they do not already exist.

f. Verify that the new extended agent workstation has been added to the `Sfinal` file. The following example is an extract of the modified `Sfinal` file containing the addition of the MDM_XA extended agent workstation, the substitution of the SCRIPTNAME keyword with DOCOMMAND in all jobs defined in FINAL and FINALPOSTREPORTS job stream definitions, and the use of the *UNISONHOME* variable in place of the path to the scripts:
FINAL:

```
SCHEDULE MDM_XA#FINAL ON EVERYDAY
         AT 2359
         CARRYFORWARD
FOLLOWS MDM_XA#FINAL.SWITCHPLAN  PREVIOUS
...
...
...
         STARTAPPSERVER DOCOMMAND
"${UNISONHOME}/../appservertools/startAppServer.sh"
         STREAMLOGON wa95ids
         RECOVERY CONTINUE
         MAKEPLAN DOCOMMAND "${UNISONHOME}/MakePlan"
         STREAMLOGON wa95ids
         RCCONDSUCC "(RC=0) OR (RC=4)"
         FOLLOWS STARTAPPSERVER
         SWITCHPLAN DOCOMMAND "${UNISONHOME}/SwitchPlan"
         STREAMLOGON wa95ids
        FOLLOWS MAKEPLAN
...
...
...
END
```

FINALPOSTREPORTS:

```
SCHEDULE  MDM_XA#FINALPOSTREPORTS ON EVERYDAY
         SCHEDTIME 2359
         CARRYFORWARD
FOLLOWS MDM_XA#FINAL.SWITCHPLAN  PREVIOUS
...
...
...
         CHECKSYNC DOCOMMAND "${UNISONHOME}/bin/planman checksync"
         STREAMLOGON wa95ids
         RECOVERY CONTINUE
         CREATEPOSTREPORTS DOCOMMAND "${UNISONHOME}/CreatePostReports"
         STREAMLOGON wa95ids
         RECOVERY CONTINUE
         UPDATESTATS DOCOMMAND "${UNISONHOME}/UpdateStats"
         STREAMLOGON wa95ids
         RECOVERY CONTINUE
         FOLLOWS CHECKSYNC
...
...
...
END
```

8. Submit the `composer add Sfinal` command and then verify that the FINAL and FINALPOSTREPORTS job streams and the related jobs, are correctly defined on the extended agent workstation. The following is an example of the correct output:

```
...
...
...
/
-add Sfinal
AWSJCL003I The command "add" completed successfully on object "jd=MDM_XA#STARTAPPSERVER".
AWSJCL003I The command "add" completed successfully on object "jd=MDM_XA#MAKEPLAN".
AWSJCL003I The command "add" completed successfully on object "jd=MDM_XA#SWITCHPLAN".
AWSJCL003I The command "add" completed successfully on object "js=MDM_XA#FINAL".
AWSJCL003I The command "add" completed successfully on object "jd=MDM_XA#CHECKSYNC".
AWSJCL003I The command "add" completed successfully on object "jd=MDM_XA#CREATEPOSTREPORTS".
AWSJCL003I The command "add" completed successfully on object "jd=MDM_XA#UPDATESTATS".
AWSJCL003I The command "add" completed successfully on object "js=MDM_XA#FINALPOSTREPORTS".
AWSBIA090I For file "Sfinal": errors 0, warnings 0.
AWSBIA288I Total objects updated: 8
```

9. Compare the two copies of the FINAL and FINALPOSTREPORTS job streams and make any necessary changes to those on the extended agent workstation, for example, the job stream submit time, run cycles, or any other custom changes to personalize the schedule.
10. Submit JnextPlan with the -noremove options to update the plan with the new extended agent workstation:

```
JnextPlan -for 0000 -noremove
```

11. If JnextPlan runs correctly, proceed to delete the FINAL and FINALPOSTREPORTS job streams previously set to "draft" on the master domain manager.

```
composer del FINALPOSTREPORTS
composer del FINAL
```

12. Delete the FINAL and FINALPOSTREPORTS job streams from the plan as follows.

```
conman "canc FINALPOSTREPORTS"
conman "canc FINAL"
```

13. Modify the new job stream definitions for the FINAL and FINALPOSTREPORTS job streams, setting the limit to "`0`":

```
SCHEDULE MDM_XA#FINAL
DESCRIPTION "Added by composer."
ON RUNCYCLE RC1 "FREQ=DAILY;"
AT 2359
CARRYFORWARD
FOLLOWS MDM_XA#FINAL.SWITCHPLAN PREVIOUS
LIMIT 0
:
MDM_XA#STARTAPPSERVER
```

14. Submit first the FINAL, and then the FINALPOSTREPORTS job streams into the current plan.

    ```
    conman sbs MDM_XA#FINAL
    conman sbs MDM_XA#FINALPOSTREPORTS
    ```

15. Verify that the start time and date for the FINAL and FINALPOSTREPORTS job streams are correct by submitting the conman showschedules command.
16. Reset the value of the limit job stream keyword for the FINAL and FINALPOSTREPORTS job streams, both in the database and in the plan.

    ```
    conman "limit MDM_XA#FINAL ;10"
    conman "limit MDM_XA#FINALPOSTREPORTS ;10"
    ```

    Both job streams should be in WAITING (HOLD internal status), awaiting execution time.
17. Archived plans, forecast and trial plans are stored on the master domain manager where the plans run. To make these plans available on the backup master domain manager, either store the plan in a single shared folder, or create a job that synchronizes the plans between the master domain manager and the backup master domain manager.

What to do next

To enable the new master to access the plans that ran on the original master (the current plan is visible because it is synchronized with the backup), configure a job that copies the plans from the original master to the new master.

After an automatic failover, if you would like to subsequently return service to the original master, you must perform a manual switch. See Manually switching the master on page 399.

Manually switching the master

A manual switchover from the primary master domain managerto a backup master domain manager is invoked through the switchmgr command. The backup master domain manager becomes the current, active master connected to the HCL Workload Automation database.

There are four main use case scenarios that can prompt the need for switch of the master to a backup:

Planned outage

The domain manager is replaced with its backup for planned maintenance work (for example, an upgrade of the operating system).

Unplanned outage

The domain manager is replaced with its backup because of an unexpected failure or malfunction.

Short-term

The domain manager is expected to return to service before the next new production period turnover (run of the JnextPlan job).

Long-term

The domain manager is not expected to return to service before the next new production period turnover (run of the JnextPlan job).

Short-term switch of a master domain manager

Use the procedure described in Simplified procedure for switching a domain manager on page 385 when you have a short-term loss of a master domain manager.

Master domain managers remain switched until you perform another switch manager operation. To return to the original master domain manager, repeat this procedure before the next production period turnover, unless you do not expect the master domain manager to be available for the next production period turnover (final Job Scheduler and JnextPlan job). In this case, use the procedure in the following section.

Extended loss or permanent change of master domain manager

Use the following procedure to switch to the backup if the original master domain manager is not expected to return to service before the next new production period turnover (final Job Scheduler and JnextPlan job).

On UNIX™ operating systems, use forward slashes in path names.

1. Use the conman **stop** function to stop HCL Workload Automation on the master domain manager and its backup. for more information about the command, see the section about the stop command in *User's Guide and Reference*.
2. If you copied the `Security` file from the master domain manager to the backup master domain manager `with a suffix`, now delete the `Security` file on the backup master domain manager and rename the `Security` file with the suffix as just `Security`.
3. If you copied the `localopts` file from the master domain manager to the backup master domain manager `with a suffix`, now merge the `localopts` file on the backup master domain manager with the `localopts` file from the master domain manager. Look at each property in turn and determine which version you want to keep on what is going to be your new master domain manager. For example, the property thiscpu needs to be the one from the backup master domain manager, but the options for controlling how the processes run can be taken from the master domain manager.
4. On the backup master domain manager cancel the `final` Job Scheduler in the Symphony file (it refers to the next production period's JnextPlan on the old master domain manager).
5. On the backup master domain manager, use composer to modify any important job streams that run on the master domain manager, in particular the `final` Job Scheduler. For each of these, change the workstation name to the name of the backup.
6. Change the workstation definition of the master domain manager from `manager` to `fault-tolerant agent`.
7. Change the workstation definition of the backup master domain manager from `fault-tolerant agent` to `manager`.

   Note: These two steps must be done in the order given, as the system will not allow you to have two managers at the same time.

8. On the backup master domain manager, edit the `*TWA_home*/TWS/mozart/globalopts` file and change the `master` option to the name of the backup master domain manager workstation (this is used mainly for reports production)
9. Use the conman **switchmgr** function to switch to the backup master domain manager. See Simplified procedure for switching a domain manager on page 385.
10. Submit a new *final* Job Scheduler to the new master domain manager (old backup master domain manager).

11. Run JnextPlan -for 0000 on the new master domain manager to generate the new `Symphony` file.
12. Remember to log on to the backup master domain manager when opening the Dynamic Workload Console, first defining a new engine to access it.
13. If the old master domain manager has failed or is being replaced, you can now delete its workstation definition and remove it from the network.

Switching a master domain manager or dynamic domain manager

Switching a master domain manager or dynamic domain manager affects the running dynamic workload broker server.

The installation of a master domain manager or dynamic domain manager and of its backup workstations includes also the installation of a dynamic workload broker server.

You might have to switch the master domain manager or dynamic domain manager because, for example, the system running the current workstation is down. When a conman switchmgr command is submitted, an automatic process is triggered by which the old server stops the dynamic scheduling services and the new server starts a new instance of the dynamic workload broker server when the older server has completed the switch. This process ensures that there is only one active dynamic workload broker server running at a time.

You can configure this automatic switch broker process on instances at version 9.5 or later, by modifying the following properties contained in the `SwitchBroker.properties` file located in *`TWA_DATA_DIR`*`/broker/config/SwitchBroker.properties`:

Table 68. Configurable properties for automatic switch broker process

Property	Description
Master.Switch.HostName	The master domain manager name used to identify the active master.
Master.Switch.ExpiringTime	The number of seconds the new master waits before becoming the active master. The default value is `300` seconds.
Master.Switch.PollingTime	The time interval, in seconds, between the database checks made by the new master on the status updates of the old master. The default is `5` seconds.

When the switchmgr command is submitted, the new master begins monitoring the database (with the frequency specified by **Master.Switch.PollingTime**), to verify when the state changes for the old master. If there is no response from the old master (because of a crash or because the old master is at a product level version earlier than V9.5) then the new master waits for a maximum of two intervals specified by **Master.Switch.PollingTime** (10 seconds), and then automatically promotes itself as the new active master. If, instead, a status update is detected in the database while polling, the new master waits the amount of time specified by **Master.Switch.ExpiringTime** before declaring itself the new active master. As soon as the old master completes the switch procedure, then the new master declares itself as the active master without waiting for the expiry of the **Master.Switch.ExpiringTime**.

The properties must be modified on both the master domain manager or dynamic domain manager and their backups at version 9.5 or later.

Here is the procedure to follow every time you switch the master domain manager or dynamic domain manager if you run dynamic scheduling in your network:

1. Set the job fence to **go** priority level. For further details, see the section about the fence command in *User's Guide and Reference*.
2. Switch the master domain manager or dynamic domain manager to a backup workstation. Use either the `conman switchmgr` command or the Dynamic Workload Console. For more information about both methods, see the procedure described under the switchmgr command in *User's Guide and Reference*.
3. Once the switch has been performed, restore the job fence to zero. For further details, see the procedure described under the switchmgr command in *User's Guide and Reference*.

Switching a dynamic domain manager for a Z controller

As a normal behavior, the dynamic domain manager for a Z controller works by having both the server and processes up and running, while the backup dynamic domain manager for a Z controller works with the processes that are running and the system that is down. You might have to switch the dynamic domain manager for a Z controller to a backup workstation because, for example, the system running the current workstation goes down.

To switch the dynamic domain manager for a Z controller to the backup workstation, perform the following procedure:

1. Stop the server on the dynamic domain manager for a Z controller by issuing the following command:

   On Windows

   ```
   stopAppServer.bat
   ```

   On UNIX

   ```
   stopAppServer.sh
   ```

2. On the backup dynamic domain manager for a Z controller, run the following command:

   On Windows

   ```
   startZosDDM.bat -dbUsr db_user -dbPwd db_user_pwd
   ```

   On UNIX

   ```
   startZosDDM.sh -dbUsr db_user -dbPwd db_user_pwd
   ```

   where:

   dbUsr *db_user*

   The user that has been granted access to the HCL Workload Automation for Z tables on the database server.

dbPwd *db_user_pwd*

The password for the user that has been granted access to the HCL Workload Automation for Z tables on the database server.

3. On the backup dynamic domain manager for a Z controller, start the server by running the following command:

On Windows

```
startAppServer.bat
```

On UNIX

```
startAppServer.sh
```

Cloning scheduling definitions from one environment to another

Replicating scheduling definitions from one environment for example, a test environment, to a different one, for example a production environment.

About this task

This section applies to HCL Workload Automation master domain managers and its backup. It documents how to clone HCL Workload Automation data from one environment to another.

Note: This cloning procedure does not clone the following information from the source environment:

- The preproduction plan
- The history of job runs and job statistics
- The audit records
- The state of running event rule instances. This means that any complex event rules, where part of the rule has been satisfied prior to cloning of the environment, are generated as new rules after the cloning procedure. Even if the subsequent conditions of the event rule are satisfied, the record that the first part of the rule was satisfied is no longer available, so the rule will never be completely satisfied.

With the following steps all scheduling object definitions and global options can be migrated from a source environment named "ENV_1" to a target environment named "ENV_2".

1. In ENV_2, install a fresh instance of an HCL Workload Automation version 10.2.5 master domain manager and it point to its database by defining `MDM_ENV2` as the master domain manager workstation name. The installation process automatically defines the following workstations in the `ENV_2` database:
    - `MDM_ENV2` is the master domain manager workstation name.
    - `MDM_ENV2_DWB` is the broker workstation name.
    - `MDM_ENV2_1` is the agent workstation name.
    - `MASTERAGENTS` is the dynamic pool which includes, by default, `MDM_ENV2_1` dynamic agent workstation.

2. On the ENV_1 master domain manager, run the `dataexport` command or script to export all scheduling object definitions and global options from `ENV_1`. You can find this file in the `bin` subdirectory of the TWA_home directory.

   Run `dataexport` from a Windows™ or UNIX® command prompt as follows:

   ```
   dataexport source_dir export_dir
   ```

   where:

   source_dir

   The *TWS_HOME* directory of the `ENV_1` instance of HCL Workload Automation version 10.2.5.

   Note: The dataexport utility only accepts *source_dir* values that end in *TWS*.

   export_dir

   The directory where the export files are created.

   For example:

   ```
   dataexport.cmd F:\IWS1025\twsDB2user F:\IWS1025\export
   ```

   The object definitions and the global options are retrieved from the `ENV_1` database and placed in the `F:\IWS1025\export` directory.
3. Verify that the following files were created in `export_dir`:
   - `acls.def`
   - `calendars.def`
   - `erules.def`
   - `folders.def`
   - `globalOpts.def`
   - `jobs.def`

     Note: The record length supported by DB2® is 4095 bytes, but it decreases to 4000 bytes with Oracle. When you migrate your job definitions to Oracle, any job with task string (scripts or commands) exceeding 4000 bytes in length are not migrated. In this case, the data import utility replaces the job definition with a dummy job definition and sets the job priority to `0`, guaranteeing that successors are not run.
   - `parms.def`
   - `prompts.def`
   - `rcgroups.def`
   - `resources.def`
   - `scheds.def`
   - `sdoms.def`
   - `srols.def`
   - `topology.def`

- `users.def` (includes encrypted user passwords)
- `vartables.def`

4. Open the *`export_dir`*`\topology.def` file and remove the MASTERAGENTS definition to avoid replacing the same workstation definition that was created when you installed a fresh instance of the HCL Workload Automation version 10.2.5 master domain manager in ENV_2.

   If you plan to dismiss ENV_1 and you want to move the other workstations from ENV_1 to ENV_2, then you do not have to perform any additional steps.

   If you plan to install new agents in ENV_2, then it is recommended to install them using the same **displayname** used by the agent present in ENV_1 so that you do not need to modify the `erules.def`, `jobs.def`, `resources.def`, `scheds.def`, and `users.def` files exported in the previous step. If you do not install them using the same **displayname**, then you must edit these files so that they match the agent **displayname** present in ENV_2.
5. Edit the *`export_dir`*`\users.def` file to specify the current valid password for the Windows™ users.
6. To import the object definitions and the global options retrieved from the ENV_1 into the ENV_2 database, copy the files that were created when you ran the dataexport utility to a directory on the ENV_2 master domain manager and run the `dataimport` command or script to import all scheduling object definitions and global options to the ENV_2 database. You can find this file in the `bin` subdirectory of the TWA_home directory.

   Run `dataimport` from a Windows™ or UNIX® command prompt as follows:

   ```
   dataimport source_dir export_dir
   ```

   where:

   source_dir

   The *TWS_HOME* directory of the ENV_2 instance of HCL Workload Automation version 10.2.5.

   export_dir

   The directory where you copied the object definitions and the global options retrieved from ENV_1.

   For example:

   ```
   dataimport.cmd F:\IWS1025\twsDB2user F:\IWS1025\export
   ```
7. If you want to dismiss the instance of HCL Workload Automation installed in ENV_1 and reuse the same agents in ENV_2, stop the HCL Workload Automation processes running on the master domain manager in ENV_1 to avoid conflicts.

Results

You have now completed cloning scheduling definitions from one environment to another.

Changing key HCL Workload Automation passwords

About this task

When you change passwords for key users in your HCL Workload Automation environment, there are various operations to perform, depending on which user's password is being changed, the type of operating system on which it is deployed, and the type of HCL Workload Automation node where the password is being changed.

Perform one or more of the following steps depending on the characteristics of your environment and your operating systems:

- Change the user password at the operating system level using native commands.

  If you use special characters in the password, ensure you use a "\" (backslash) before the special character. The following rules apply:

  On Windows™ operating systems:

  Passwords for users can include any alphanumeric characters and ()!?=^*/~[]$_+;:.,@`-#.

  On UNIX™ and LINUX systems:

  Passwords for users can include any alphanumeric characters and ()!?=*~_+.-.
- Change the password for the TWS user by changing the WebSphere Application Server Liberty user password. For more information, see the WebSphere Application Server Liberty documentation, for example securityUtility command. For the detailed procedure, see Changing the WebSphere Application Server Liberty password on page 407.
- Change the password used by command-line clients, as described in Change password used by command-line clients to access the master domain manager on page 409.
- Change the password used by fault-tolerant agents, as described in Change password used by fault-tolerant agent systems to access the master domain manager (for conman) on page 409.
- Change the password for the database user, as described in Changing the properties for the database on page 412.
- Change the engine connection parameters in the Dynamic Workload Console, as described in Update the engine connection parameters in the GUIs on page 410.
- On Windows operating systems, change the password of the HCL Workload Automation account in Windows services, as described in Windows - update Windows services on page 410.
- If you run jobs on Windows operating systems, change the password of the streamlogon user of the jobs, as described in Change the HCL Workload Automation user definition on page 410.
- Run the following commands to ensure the previous steps were completed correctly:

  ```
  optman ls
  ```

  ```
  composer mo ws=@
  ```

For all other users of HCL Workload Automation, no action is required if their passwords change.

Note: After changing any password, restart WebSphere Application Server Liberty.

See the following table to determine if a change of password requires actions to be taken for a role on the different HCL Workload Automation components. Look up the role and the component and determine from the corresponding table cell where the changes must be made:

- If the cell contains a `"✓",` make the change on the system where the indicated component is running
- If the cell contains "MDM", make the change on the master domain manager to which the component belongs

Table 69. If and where password changes are required

Role	MDM	BKM	FTA
WebSphere Application Server Liberty user	✓	✓	
Database user	✓	✓	
Streamlogon user (Windows®)	✓	✓	MDM

For example, if you are the TWS_user (the instance owner) of a fault-tolerant agent, you need to implement the password change on the system where the fault-tolerant agent is installed, but if you are also the streamlogon user of jobs running on that system, the changes required for the new password must be applied at the master domain manager to which the fault-tolerant agent belongs.

Changing the WebSphere Application Server Liberty password

Procedure to modify the WebSphere Application Server Liberty password.

To modify the WebSphere Application Server Liberty password, update the `wauser_variables.xml` configuration file. This operation changes the password for the TWS user, which is stored in the configuration file. Ensure you also update the password in the `useropts` file, so that you can continue working with the command-line client. If you have several users using the command-line client, change the password of the TWS user in all `useropts` files (one for each user). For more information, see Change password used by command-line clients to access the master domain manager on page 409.

1. The location of the file to be modified varies depending on whether you have already customized the template or not, as follows:

   If you have no previous customizations defined in the configuration file,

   edit the template located in the `templates` folder.

   Templates for the master domain manager are stored in the following paths:

   On UNIX operating systems

   TWA_home`/usr/servers/engineServer/configDropins/templates`

   On Windows operating systems

   TWA_home`\usr\servers\engineServer\configDropins\templates`

Templates for the Dynamic Workload Console are stored in the following paths:

On UNIX operating systems

DWC_home`/usr/servers/dwcServer/configDropins/templates`

On Windows operating systems

DWC_home`\usr\servers\dwcServer\configDropins\templates`

If you have previous customizations defined in the configuration file,

edit the file located in the `overrides` folder, which is the file currently used by WebSphere Application Server Liberty.

The `wauser_variables.xml` file is located in

On UNIX operating systems

TWA_DATA_DIR`/usr/servers/engineServer/configDropins/overrides`

On Windows operating systems

TWA_home`\usr\servers\engineServer\configDropins\overrides`

On UNIX operating systems

DWC_DATA_dir`/usr/servers/dwcServer/configDropins/overrides`

On Windows operating systems

DWC_home`\usr\servers\dwcServer\configDropins\overrides`

2. Copy the `wauser_variables.xml` file to a temporary directory.
3. Edit the file as necessary, specifying the new password.

   The contents of the `wauser_variables.xml` file is as follows:

```
<server description="wauser_var">
                <variable name="user.twsuser.id" value="$(wlpUser)"/>
                <variable name="user.twsuser.password" value="$(wlpPassword)"/>
                </server>
```

   where:

   user.twsuser.id

   Is the WebSphere Application Server Liberty user ID.

   user.twsuser.password

   Is the new WebSphere Application Server Liberty password.

4. Optionally, create a backup copy of the configuration file in a different directory, if the file is already present.
5. Copy the `wauser_variables.xml` file to the `overrides` folder. Changes are effective immediately.
6. Update the PASSWORD in the `useropts` file on the following workstations:

- on every command-line client that points to your workstation.
- on every fault-tolerant agent in your environment that has an HTTP/HTTPS connection defined in `localopts` that points to your workstation. The HTTP/HTTPS connection is used to submit a predefined job or job stream.
- in the engine connection parameters on every connected Dynamic Workload Console.

If you have more than one instance of HCL Workload Automation on a system, you might have implemented separate user options files to make separate connections. In this case, check the **useropts** key in the `localopts` file on each instance to determine the name of the specific `useropts` file for that instance.

Change password used by command-line clients to access the master domain manager

About this task

If you have changed the password of the WebSphere Application Server Liberty user that command-line clients use to connect to the master domain manager, the connection parameters must be updated.

Follow this procedure:

1. Identify all systems that have a command line client remote connection defined with the master domain manager
2. On these workstations, open the user options files. The user option file is located:

   On Windows operating systems

   ```
   C:\Users\<user>\.TWS\useropts_<user>
   ```

   On Linux operating systems

   ```
   /home/<user>/.TWS/useropts_<user>
   ```

   If you have more than one instance of HCL Workload Automation on a system, you might have implemented separate user options files to make separate connections. In this case, check the **useropts** key in the `localopts` file on each instance to determine the name of the specific `useropts` file for that instance.
3. For each file, locate the password key (encrypted) and change its value to that of the new password in plain text, enclosed in double quotation marks. The password is saved in clear, but will be encrypted at first logon of the User ID.
4. Save the files.

Change password used by fault-tolerant agent systems to access the master domain manager (for conman)

About this task

If you have changed the password of the WebSphere Application Server Liberty user that is used by fault-tolerant agents with an HTTP or HTTPS connection defined in the local options that points to the master domain manager, the connection parameters must be updated.

Follow this procedure:

1. Identify all fault-tolerant agents with an HTTP or HTTPS connection defined in the local options that points to the master domain manager.
2. On these workstations, open the user options file `<Root_home>/.TWS/useropts`
3. Locate the password key (encrypted) and change its value to that of the new password in plain text, enclosed in double quotation marks. The password is saved in clear, but will be encrypted at first logon of the User ID.
4. Save the file.

Update the engine connection parameters in the GUIs

About this task

If you have changed the password of the WebSphere Application Server Liberty user that is used by the Dynamic Workload Console to connect to the distributed engine, the engine connection parameters must be updated, as follows:

1. On each instance of the Dynamic Workload Console locate the page where you modify the distributed engine connection parameters
2. Change the password and submit the page.

Windows™ - update Windows™ services

About this task

On Windows™, the *TWS_user* account is used to start the following services:

- HCL Token Service for *TWS_user*
- HCL Workload Scheduler for *TWS_user*

The password must be updated in the properties of these services, or they are not able to start at next reboot. This is done as follows:

1. Stop all HCL Workload Automation processes. See Unlinking and stopping HCL Workload Automation on page 411 for details.
2. Restart all HCL Workload Automation processes using the StartUp command.

Change the HCL Workload Automation user definition

About this task

If the user ID is used within HCL Workload Automation to run jobs, follow this procedure:

1. Run the composer modify user command. The user details of the selected user are written to a temporary file, which is opened. Fore more information, see the topic about the modify command in *User's Guide and Reference*
2. Edit the password field so that it contains the new password value delimited by double quotation marks characters (").

3. Save the file, and the contents are added to the database.
4. To make the change immediately effective in the current plan, issue the conman altpass command. For more information, see the topic about the altpass command in *User's Guide and Reference*

Unlinking and stopping HCL Workload Automation

About this task

Before you perform an upgrade or uninstall, install a fix pack, or perform maintenance activities, ensure that all HCL Workload Automation processes and services are stopped. Follow these steps:

1. If you have jobs that are currently running on the workstation, wait for them to finish. To determine which are not finished, check for jobs that are in the *exec* state. When there are no jobs in this state, and you have allowed sufficient time for all events to be distributed in your network, you can continue with the rest of the procedure.
2. If the workstation that you want to stop is not the master domain manager, unlink the workstation by issuing the following command from the command line of the master domain manager:

   ```
   conman "unlink workstationname;noask"
   ```

3. Stop WebSphere Application Server Liberty by using the `conman stopappserver` command (see Starting and stopping the application server and appservman on page 431).
4. All HCL Workload Automation processes on the workstation must then be stopped manually. From the command line, while logged on as the <*TWS_user*>, enter the following command:

   ```
   conman “stop;wait”
   ```

5. From the command line, stop the netman process as follows:

   UNIX®

   Run the conman “shut" command.

   Note: Do not use the UNIX® kill command to stop HCL Workload Automation processes.

   Windows®

   From the HCL Workload Automation home directory, run the shutdown.cmd command.

6. To stop dynamic agents, run the ShutDownLwa command.
7. To stop the SSM agent, perform the following steps:
   - On Windows®, stop the service HCL Workload Automation SSM Agent (for <TWS_user>).
   - On UNIX®, run the stopmon command.

What to do next

To verify if there are services and processes still running:

UNIX®

Enter the command: ps -u <TWS_user>Verify that the following processes are not running: netman, mailman, batchman, writer, jobman, JOBMAN, stageman, logman, planman, monman, ssmagent.bin, and appservman.

Windows®

Run **Task Manager**, and verify that the following processes are not running: netman, mailman, batchman, writer, jobman, stageman, JOBMON, tokensrv, batchup, logman, planman, monman, ssmagent, and appservman.

Also, ensure that no system programs are accessing the directory or its sub-directories, including the command prompt and Windows® Explorer.

Changing the properties for the database

About this task

When you installed HCL Workload Automation, you supplied the database name, the server name, port, the host name of the database server, and other information.

If you need to change any of the database properties such as host name, port, instance owner, or database name, use the `datasource_`*`db_vendor`*`.xml` template file to reflect these changes in the application server on the master domain manager or on the Dynamic Workload Console.

If you want to change any of these properties, perform the following steps:

1. Change the configuration of the HCL Workload Automation application server so that it points correctly to the changed database configuration, as follows:
   a. The location of the file to be modified varies depending on whether you have already customized the template or not, as follows:

      If you have no previous customizations defined in the configuration file,

      edit the template located in the `templates` folder.

      Templates for the master domain manager are stored in the following paths:

      On UNIX operating systems

      TWA_home`/usr/servers/engineServer/configDropins/templates`

      On Windows operating systems

      TWA_home`\usr\servers\engineServer\configDropins\templates`

      Templates for the Dynamic Workload Console are stored in the following paths:

      On UNIX operating systems

      DWC_home`/usr/servers/dwcServer/configDropins/templates`

      On Windows operating systems

      DWC_home`\usr\servers\dwcServer\configDropins\templates`

      If you have previous customizations defined in the configuration file,

      edit the file located in the `overrides` folder, which is the file currently used by WebSphere Application Server Liberty.

The `wauser_variables.xml` file is located in

On UNIX operating systems

TWA_DATA_DIR`/usr/servers/engineServer/configDropins/overrides`

On Windows operating systems

TWA_home`\usr\servers\engineServer\configDropins\overrides`

On UNIX operating systems

DWC_DATA_dir`/usr/servers/dwcServer/configDropins/overrides`

On Windows operating systems

DWC_home`\usr\servers\dwcServer\configDropins\overrides`

b. Copy the `datasource_`*`db_vendor`*`.xml` file to a temporary location.

c. Modify the following properties in the file based on the values you changed in your database configuration:

db.serverName

The name or IP address of the database server

db.portNumber

The port number of the database server

db.databaseName

The database name

db.user

The database instance owner

db.password

The database user password. You can optionally encrypt the password, as described in the topic about the secure script in *HCL Workload Automation: Planning and Installation*.

db.driver.path

The path to the JDBC drivers. HCL Workload Automation is supplied using the JDBC driver type 4 for DB2® and type 2 for Oracle. However, each can use the other driver type, if necessary. Software Support might ask you to change to this driver. This procedure must only be performed under the control of Software Support.

db.sslConnection

The setting for the SSL connection. `true` indicates that SSL connection is enabled, `false` indicates that SSL connection is disabled. .

d. Browse to the `overrides` folder:

    e. Create a backup of the `datasource_`*`db_vendor`*`.xml`.
    f. Replace the `datasource_`*`db_vendor`*`.xml` file with the file you edited. The changes are effective immediately.

2. On the master domain manager only, edit the file `CLIConfig.properties`, in the path *TWA_DATA_DIR*`//broker/config`, by updating the value for the property **com.ibm.tdwb.dao.rdbms.jdbcPath** to reflect the JDBC URL specified. The following is an example of a JDBC value:

   ```
   database_type://hostname:port/dbName
   ```

3. On the master domain manager only , edit the file `DAOCommon.properties`, in the path *TWA_DATA_DIR*`//broker/config`, as follows:
   - In the first line specify the **rdbmsName** for all DBs.
   - For Oracle only, change all lines and specify the *TWS_Oracle_User*.
4. On the master domain manager only, edit the file `TWSConfig.properties`, in the path *TWA_DATA_DIR*`/usr/servers/engineServer/resources/properties/TWSConfig.properties`. If you are using MSSQL change the first line. For Oracle only, change all lines. Consider the following example:

   ```
   com.ibm.tws.dao.rdbms.rdbmsName = ORACLE
    com.ibm.tws.dao.rdbms.modelSchema = <TWS_Oracle_User>
    com.ibm.tws.dao.rdbms.eventRuleSchema = <TWS_Oracle_User>
    com.ibm.tws.dao.rdbms.logSchema = <TWS_Oracle_User>
   ```

Changing the workstation host name or IP address

When you change the host name, the IP address or both on the workstations of your HCL Workload Automation environment to have it function properly, you must report the changed value on:

- The WebSphere Application Server Liberty if the following components changed the host name, the IP address or both:
  - Master domain manager
  - Backup master domain manager
  - Connector or Z connector
  - Dynamic Workload Console

  For more information, see Reporting the changes in the WebSphere Application Server Liberty configuration file on page 415.
- The following components if the workstation where you installed the RDBMS changed the host name, the IP address or both:
  - Master domain manager
  - Backup master domain manager
  - Dynamic domain manager
  - Backup dynamic domain manager

  For more information, see Changing the properties for the database on page 412.
- The workstation definitions if you installed the following components:
  - Master domain manager
  - Backup master domain manager

    - Dynamic domain manager
    - Backup dynamic domain manager
    - Fault-tolerant agent and standard agent
    - Domain manager

For more information, see Reporting the changed host name or IP address in the workstation definition on page 416.

Reporting the changes in the WebSphere Application Server Liberty configuration file

About this task

If the host name or IP address is changed for the following components, then you must report the changed value in the WebSphere Application Server Liberty `host_variables.xml` configuration file:

- Master domain manager
- Backup master domain manager
- Dynamic domain manager
- Backup dynamic domain manager
- Dynamic Workload Console

1. Stop the WebSphere Application Server Liberty.
2. Obtain the changed host name, IP address, or both.
3. Verify that the value for the properties listed below were changed with the actual values:
    - Old Hostname
    - New Hostname
    - The host names for the specific port properties

   If these values are different from the actual host name or IP address values, proceed with Step 4 on page 415. If this values are not changed, skip the steps below.
4. Create a back up of and then modify the values in the `host_variables.xml` file located in

   On UNIX operating systems

   ```
   TWA_DATA_DIR/usr/servers/engineServer/configDropins/overrides
   ```

   On Windows operating systems

   ```
   TWA_home\usr\servers\engineServer\configDropins\overrides
   ```
5. Propagate the changes to the interfaces as follows:

   Address of the master domain manager changes

    - On each fault-tolerant agent, dynamic agent, and standard agent you configured to connect to the **conman** command line, update the **host** parameter present in the "**Attributes for CLI connections**" section in the `localopts` file. Usually you have the **host** parameter defined in the `localopts` file of the workstations you use to submit predefined jobs and job streams (sbj and sbs commands).

- On every command-line client, update the **host** parameter present in the "**Attributes for CLI connections**" section in the `localopts` file.
- On the Dynamic Workload Console, update the engine connections.
- On all of the server components:
    a. Run JnextPlan.
    b. Update the value for the **Master.Switch.HostName** parameter in the `SwitchBroker.properties` file located in *TWA_DATA_DIR*`/broker/config/SwitchBroker.properties`.
    c. From the broker command line, run exportserverdata to extract a list of URIs containing the old hostname to a text file, then run importserverdata providing the updated file in input to update the changed host name. For more information about the commands, see the section about using utility commands in the dynamic environment in *User's Guide and Reference*.

Address of the Dynamic Workload Console changes

Notify all the users of the new web address.

Reporting the changed host name or IP address in the workstation definition

About this task

Run this procedure if you changed the host name or IP address on the following components:

- Master domain manager
- Backup master domain manager
- Fault-tolerant agent and standard agent
- Domain manager

To modify the host name or the IP address on the workstation definition, perform the following steps:

1. Use **composer** or the Dynamic Workload Console to check the workstation definition stored in the database for the HCL Workload Automation instance installed on the workstation where the IP address or the host name changed.
2. Verify the **node** attribute contains the new host name or IP address. If this value is changed proceed with Step 3 on page 416. If this value is not changed skip the steps below.
3. Change the value of the **node** parameter with the new value.
4. Refresh the new workstation definition into the plan. Do it immediately if you are changing the host name or the IP address of a master domain manager or a domain manager. If you are changing them on a workstation that is not a master domain manager or a domain manager you can wait the next scheduled plan generation to refresh your workstation definition in the Symphony file. In this case during this production day you cannot run jobs on this workstation. To generate the plan, perform the following steps:
    a. Run **optman ls** and take note of the actual value of the **enCarryForward** parameter.
    b. If this value is not set to **all**, run

    ```
    optman chg cf=ALL
    ```

to set it to **all**

c. Add the new workstation definition to the plan, by running:

```
JnextPlan -for 0000
```

d. Reassign the original value to the **enCarryForward** parameter.

Reporting the changed host name or IP address of the dynamic workload broker server

About this task

The dynamic workload broker server is a component that HCL Workload Automation installs when you install the following components:

- Master domain manager
- Backup master domain manager
- Dynamic domain manager
- Backup dynamic domain manager

If you changed the host name or the IP address on the dynamic workload broker server, or if you installed a new one run the procedure described in Reporting the changes in the WebSphere Application Server Liberty configuration file on page 415.

If you changed the host name or the IP address on a master domain manager or backup master domain manager and you ran the Reporting the changes in the WebSphere Application Server Liberty configuration file on page 415 procedure, skip this section.

If you changed the host name or the IP address on the dynamic domain manager or backup dynamic domain manager you do not need to change the definition of your broker workstation (type **broker**), because the value of the **node** attribute is set to the *localhost* value to allow to switch between the dynamic workload broker server and its backup.

After you ran the procedure, propagate the changes to the dynamic agent and update the **ResourceAdvisorURL** property in the `JobManager.ini` file on each agent connected to that dynamic workload broker server, by performing the following steps:

1. Run the following command to stop the agent:

   ```
   ShutDownLwa
   ```

2. Edit the `JobManager.ini` file and change the host name or the IP address in the **ResourceAdvisorURL** property.
3. Run the following command to start the agent:

   ```
   StartUpLwa
   ```

Perform the following changes:

1. Open the `JobDispatcherConfig.properties` file and change the value of the **JDURL=https://*host_name*** property to reflect the new host name or IP address.
2. Open the `CliConfig.properties` file and change the value of the **ITDWBServerHost=/*host_name*** property to reflect the new host name or IP address.

3. Open the `ResourceAdvisorConfig.properties` file and change the value of the **ResourceAdvisorURL=https://*host_name*** property to reflect the new host name or IP address.
4. From the <*TWA_home*>`/TDWB/bin` directory, run the following command:

   On Windows operating systems:

   ```
   exportserverdata.bat
   ```

   On UNIX and Linux operating systems:

   ```
   exportserverdata.sh
   ```

   This command extracts a list of URIs (Uniform Resource Identifier) of all the dynamic workload broker instances from the HCL Workload Scheduler database and copies them to a temporary file. By default, the list of URIs is saved to the `server.properties` file, located in the current directory.
5. Change all the entries that contain the old host name to reflect the new host name.
6. Place the file back in the database, by running the following command:

   On Windows operating systems:

   ```
   importserverdata.bat
   ```

   On UNIX and Linux operating systems:

   ```
   importserverdata.sh
   ```
7. Stop WebSphere Application Server Liberty, by running the following command:

   ```
   stopAppServer
   ```
8. Start WebSphere Application Server Liberty, by running the following command:

   ```
   startAppServer
   ```

Reporting the changed host name or IP address of the dynamic agent

About this task

If you changed the host name or the IP address on the workstation where you installed the dynamic agent, the changes are automatically reported stopping and starting the agent using the following commands:

1. To stop the agent:

   ```
   ShutDownLwa
   ```
2. To start the agent:

   ```
   StartUpLwa
   ```

Note: Do not modify manually the value of the **node** parameter in the dynamic agent workstation definition.

Changing the security settings

This section describes how to modify the security settings of HCL Workload Automation.

About this task

A number of template files are available for customizing various security settings on the application server:

ssl_variables.xml

auth_basicRegistry_config.xml

File-based authentication

auth_IDS_config.xml

IBM® Directory Server

auth_OpenLDAP_config.xml

OpenLDAP

auth_OpenLDAP_config.xml

Windows Server Active Directory

For the procedure to customize the security settings, see the section about configuring a common LDAP for both the master and the console in *Planning and Installation Guide*.

For the settings related to SSL, see Configuring SSL attributes on page 328. You can also change other settings, such as the active user registry or the local operating system ID and password.

- For more information about the procedure to make any changes to the WebSphere Application Server Liberty properties, see Configuring HCL Workload Automation using templates on page 422.
- To determine which properties are to be changed, see:
    - Configuring authentication on page 265, for information about the properties to be changed to modify your user registry configuration for user authentication.
    - Customizing your RDBMS server on page 356, for information about the procedure to be performed when modifying your RDBMS server.
    - Changing the properties for the database on page 412, for information about changing the database properties such as database name, the server name, port, the host name of the database server.
    - Changing key HCL Workload Automation passwords on page 405, for information about how to change key passwords.
- To change the text file of the current security properties, perform the following steps:
    1. Edit the text file and locate the properties you need to change.
    2. Make any required changes to the properties.

       Do not change any other properties.

Managing the event processor

About this task

The only maintenance issue for the event processor is the management of the EIF event queue, `cache.dat`. The event queue is circular, with events being added at the end and removed from the beginning. However, if there is no room to write an event at the end of the queue it is written at the beginning, overwriting the event at the beginning of the queue.

To increase the size of the event processor queue, follow this procedure:

1. At the workstation running the event processor, browse to the following path:

   On Windows operating systems

   <*TWA_home*>`\TWS\stdlist\appserver\engineServer\temp\TWS\EIFListener`

   On UNIX operating systems

   <*TWA_DATA_DIR*>`/stdlist/appserver/engineServer/temp/TWS/EIFListener`

2. Edit the `eif.templ` file and locate the keyword:

   ```
   BufEvtMaxSize
   ```

3. Increase the value of this keyword, according to your requirements.
4. Stop and restart WebSphere Application Server Liberty using the conman stopappserver and conman startappserver commands (see Starting and stopping the application server and appservman on page 431).

Automatically initializing HCL Workload Automation instances

On UNIX systems, you can automatically initialize HCL Workload Automation instances during operating system startup.

About this task

On UNIX™ systems, you can ensure that your HCL Workload Automation instances are automatically initialized during operating system startup.

For AIX® and some Linux™ operating systems that use a traditional **init** like System V, you can do this by adding an HCL Workload Automation service to the **init** process of your operating system. Use the sample start script **iwa_init_**<*installation user*> located in `TWA_home/TWS/config` and add it to the appropriate run level after customizing it as necessary.

For some Linux™ distributions that use systemd as the default initialization system, such as RedHat Enterprise Linux™ v7.0 and SUSE Linux™ Enterprise Server V12, a sample service file, `iwa.service`, is provided located in the path `TWA_home/TWS/config` that is already configured to support the automatic initialization of HCL Workload Automation instances at startup.

Perform the following steps:

For UNIX operating systems that use the traditional init system such as System V :

Configure the script and then register the service.

1. Create a copy of the **iwa_init_**<`installation user`> script based on your requirements. Provide the following information, depending on the operating system you use:

   Required-Start

   On Linux™ systems, specify the precondition services

   Default-Start

   On Linux™ systems, specify the required runlevels. For example, specify runlevels 2, 3, and 5.

2. Browse to the appropriate system-dependent folder, as follows:

   Supported Linux™ operating systems

   Save the script in the /etc/init.d folder and register the service using the **insserv -v *script_name*** command.

   Supported AIX® operating systems

   Save the script to the appropriate rc*runlevel*.d folder. Rename the script according to the runlevel script definition. For example, S*sequence_numberservice_name*, as in S10iwa_init_<`installation user`>.

Note: If you run this script on a master domain manager or on a backup master domain manager, the script has no effect on the database.

For more information about the **inittab, init.d, insserv**, and **init** commands, see the reference documentation for your operating system.

For Linux™ distributions that use systemd as the default initialization system:

This procedure uses the `iwa.service` sample service that is already customized to automatically initialize HCL Workload Automation instances at system startup.

1. Copy the sample service provided, `iwa.service`, located in the path `TWA_home/TWS/config` to the following path on the Linux™ system `/etc/systemd/system/`
2. To make systemd aware of the service, invoke the systemctl daemon-reload command.
3. Start the service submitting the following command:

   ```
   systemctl start iwa.service
   ```

4. Verify the status by submitting the following command:

   ```
   systemctl status iwa.service
   ```

5. Stop the service by submitting the following command:

   ```
   systemctl stop iwa.service
   ```

6. Finally, enable the service so that it is activated by default when the system boots by submitting the following command:

   ```
   systemctl enable iwa.service
   ```

Configuring HCL Workload Automation using templates

Starting from version 9.5, HCL Workload Automation has moved from WebSphere Application Server to WebSphere Application Server Liberty Base. As a result, the utilities known as wastools have been replaced with a number of templates addressing widely used configurations.

Starting from version 10.2, HCL Workload Automation supports both WebSphere Application Server Liberty Base and Open Liberty. In this topic, the term WebSphere Application Server Liberty is used to indicate both products.

You can now configure WebSphere Application Server Liberty to work with HCL Workload Automation using the templates provided or define your custom `.xml` files containing your own configuration settings. Templates are available for both the master domain manager and the Dynamic Workload Console.

See Table 70: Correspondence between wastools and templates on page 423 to find the mapping between wastools and templates. The table indicates both the file containing the current configuration in use, as well as the template files available to modify the current configuration.

Templates for the master domain manager are stored in the following paths:

On UNIX operating systems

TWA_home`/usr/servers/engineServer/configDropins/templates`

On Windows operating systems

TWA_home`\usr\servers\engineServer\configDropins\templates`

Templates for the Dynamic Workload Console are stored in the following paths:

On UNIX operating systems

DWC_home`/usr/servers/dwcServer/configDropins/templates`

On Windows operating systems

DWC_home`\usr\servers\dwcServer\configDropins\templates`

When you edit the file with your customized settings for the master domain manager, move it to the following paths:

On UNIX operating systems

TWA_DATA_DIR`/usr/servers/engineServer/configDropins/overrides`

On Windows operating systems

TWA_home`\usr\servers\engineServer\configDropins\overrides`

When you edit the file with your customized settings for the Dynamic Workload Console, move it to the following paths:

On UNIX operating systems

DWC_DATA_dir`/usr/servers/dwcServer/configDropins/overrides`

On Windows operating systems

DWC_home`\usr\servers\dwcServer\configDropins\overrides`

Note: Do not edit the files in the `templates` directory because they will be overwritten when upgrading to new version or fix pack.

To configure WebSphere Application Server Liberty to work with HCL Workload Automation, use the template files provided in the `templates` folder or create your custom `.xml` files containing your configuration settings. WebSphere Application Server Liberty retrieves the `.xml` files from the `overrides` folder and applies the configuration settings defined in each file. The file name is irrelevant, because WebSphere Application Server Liberty analyzes each `.xml` file for its contents.

The template files provided refer to commonly used configurations. If you want to implement different configurations, for example a custom authorization mechanism, you can create a custom`.xml` file, containing your configuration settings in this section:

```
<server description="My custom configuration description">

    </server>
```

Ensure you remove any obsolete `.xml` files, to prevent WebSphere Application Server Liberty from parsing unwanted files.

If you use the provided templates, ensure you follow this procedure:

1. Copy the template file from the `templates` folder to a working folder.
2. Edit the template file in the working folder with the desired configuration.
3. Optionally, create a backup copy of the relevant configuration file present in the `overrides` directory in a different directory. Ensure you do not copy the backup file in the path where the template files are located.
4. Copy the updated template file to the `overrides` folder. Maintaining the original folder structure is not required.
5. Changes are effective immediately.

Table 70. Correspondence between wastools and templates

Function	Current configuration file in use	Template file for customization	wastool
Datasource settings	`overrides/datasource.xml`	`templates/datasources/datasource_`*`vendor`*`.xml`	`changeDataSourceProperties` `showDataSourceProperties`
Hostname and port settings	• `overrides/host_variables.xml` • `overrides/ports_variables.xml`	• Not applicable. The `host_variables.xml` file is very simple and therefore is	`changeHostProperties` `showHostProperties`

Table 70. Correspondence between wastools and templates (continued)

Function	Current configuration file in use	Template file for customization	wastool
		located only in the overrides folder. • templates/ports_variables.xml	
Authentication settings	overrides/wauser_variables.xml	Not applicable. The wauser_variables.xml file is very simple and therefore is located only in the overrides folder.	changePassword
Trace settings	Traces are disabled by default, so no file is present in the overrides folder. Copy the trace.xml file to the overrides folder to enable traces.	templates/trace.xml	changeTraceProperties
z/OS engine settings for the Dynamic Workload Console	overrides/connectionFactory.xml	zconnector/connectionFactory.xml	createZosEngine (Dynamic Workload Console installation only)
SSL connections and certificates	• overrides/authentication_config.xml • overrides/ssl_variables.xml	**File-based:** authentication/auth_basicRegistry_config.xml **IBM® Directory Server:** authentication/auth_IDS_config.xml **OpenLDAP:** authentication/auth_OpenLDAP_config.xml	showSecurityProperties changeSecurityProperties

Table 70. Correspondence between wastools and templates (continued)

Function	Current configuration file in use	Template file for customization	wastool

		Microsoft Server Active Directory:
		`authentication/auth_AD_config.xml`
		OpenID:
		`authentication/openid_connect.xml`
		`ssl_variables.xml`	

Json Web Token (JWT) management	`overrides/jwt_variables.xml`	Not applicable	Not applicable

Templates are divided into the following directories:

defaults (files used by the installation process)

- `ports_config.xml`
- `ports_variables.xml`
- `ssl_config.xml`
- `ssl_variables.xml`

overrides (configuration files)

- `authentication_config.xml`
- `connectionFactory.xml` (Dynamic Workload Console installation only)
- `datasource.xml`
- `host_variables.xml`
- `jvm.options`
- `ports_variables.xml`
- `ssl_variables.xml`
- `wauser_variables.xml`

templates (templates available for customization)

authentication

- `auth_AD_config.xml`
- `auth_basicRegistry_config.xml`

    - `auth_IDS_config.xml`
    - `auth_OpenLDAP_config.xml`
    - `openid_connect.xml`

  datasources (one file for each supported database)

    - `datasource_<database_vendor>.xml`

  zconnectors (Dynamic Workload Console installation only)

    - `connectionFactory.xml`

- `ports_variables.xml`
- `ssl_variables.xml`
- `trace.xml`
- `adminCenter.xml` (Not recommended for production environment)

For information about how to change database properties, see Changing the properties for the database on page 412.

For information about authentication settings, see Configuring authentication on page 265.

For information about configuring a z/OS engine, see the section about defining az/OS engine in the Z connector in *HCL Workload Scheduler for Z: Planning and Installation*.

For information about logs and traces, see the section about logging and tracing in *Troubleshooting Guide*.

WebSphere Application Server Liberty tasks

The tasks contained in this chapter might need to be performed for improving WebSphere Application Server Liberty performance.

Moving from WebSphere Application Server Liberty Base to Open Liberty

You can move from using WebSphere Application Server Liberty Base to Open Liberty.

About this task

HCL Workload Automation installation package contains Open Liberty, but both WebSphere Application Server Liberty Base and Open Liberty are supported. If you have a WebSphere Application Server Liberty Base license and the product is already installed in your environment, and you plan to continue using it with HCL Workload Automation, no action is required.

If you have WebSphere Application Server Liberty Base installed in your environment, but you want to move to Open Liberty, follow the procedure described below:

1. Download Open Liberty from Get started with Open Liberty.

   Check the **Detailed System Requirements** document available in Product Requirementsto ensure the latest Open Liberty version is supported by HCL Workload Automation. .

2. Stop the application server as described in the topic about application server - starting and stopping in *Administration Guide*.
   Also stop HCL Workload Automation and all other applications running on the WebSphere Application Server Liberty Base instance.
3. Optionally create a backup of the current WebSphere Application Server Liberty Base instance in a directory different from the Open Liberty installation directory.
4. Uninstall WebSphere Application Server Liberty Base.
5. Perform one of the following actions:
   a. Extract Open Liberty using the root user:

      On Windows operating systems

      ```
      unzip <openliberty_download_dir>\openliberty-<version>.zip
       -d <install_dir>
      ```

      On UNIX operating systems

      ```
      unzip <openliberty_download_dir>/openliberty-<version>.zip
       -d <install_dir>
      ```

   b. Run the following command to assign permissions:

      ```
      chmod 755 -R "wlp_directory"
      ```

   OR

   Extract Open Liberty using the user who is going to install the product, as follows:

   ```
   su - "wauser"
   unzip
   ```

   where:

   <openliberty_download_dir>

      The directory where you downloaded Open Liberty.

   install_dir

      The directory where you want to install Open Liberty.

      Note: Install the new Open Liberty in the exact location of the previous WebSphere Application Server Liberty Base installation.

6. Restart the application server as described in the topic about application server - starting and stopping in *Administration Guide*.
   Also restart HCL Workload Automation and all other applications running on the Open Liberty instance.

Application server - starting and stopping

Use the startAppServer and stopAppServer commands or the equivalent from the Dynamic Workload Console to start or stop the WebSphere Application Server Liberty. For a description of these commands, see *HCL Workload Automation: User's Guide and Reference*.

These commands also stop appservman, the service that monitors and optionally restarts WebSphere Application Server Liberty.

If you do not want to stop appservman, you can issue startAppServer or stopAppServer, supplying the –direct argument. These scripts are located in `TWA_home/appservertools`.

The complete syntax of startAppServer and stopAppServer is as follows:

UNIX™

Start the application server

```
./startAppServer.sh  [-direct]
```

Stop the application server

```
./stopAppServer.sh  [-direct]
```

Note: If your WebSphere Application Server Liberty is installed for the Dynamic Workload Console, use the following syntax:

```
./stopAppServer.sh [-direct]
            [-user <user_ID>
           -password <password>]
```

The user ID and password are optional only if you have specified them in the `soap.client.props` file located in the properties directory of the WebSphere Application Server Liberty profile. Unlike the master domain manager installation, when you install the Dynamic Workload Console the `soap.client.props` file is not automatically customized with these credentials.

Windows™

Start the application server

```
startAppServer.bat [-direct]
```

Stop the application server

```
stopAppServer.bat [-direct
            [-wlpHome <installation_directory>]
            [-options <parameters>]]
```

z/OS

Start the application server

```
./startAppServer.sh  [-direct]
```

Stop the application server

```
./stopAppServer.sh  [-direct]
```

where the arguments are as follows:

–direct

Optionally starts or stops the application server without starting or stopping the application server monitor appservman.

For example, you might use this after changing some configuration parameters. By stopping WebSphere Application Server Liberty without stopping appservman, the latter will immediately restart WebSphere Application Server Liberty, using the new configuration properties.

This argument is mandatory on UNIX™ when the product components are not integrated.

–options *parameters*

Optionally supplies parameters to the WebSphere Application Server Liberty startServer or stopServer commands. See the WebSphere Application Server Liberty documentation for details.

–wlpHome *installation_directory*

Defines the WebSphere Application Server Liberty installation directory, if it is not the default value.

Application server - automatic restart after failure

If you experience any problems with the application server failing, a service is available that not only monitors its status, but can also restart it automatically in the event of failure. The service is called appservman, and it is enabled and controlled by the local options on the computer where the application server is running.

The following sections describe the service, how it works, and how it is controlled:

- Appservman - how it works on page 429
- Controlling appservman on page 430
- Starting and stopping the application server and appservman on page 431
- Monitoring the application server status on page 431
- Obtaining information about application server failures on page 432
- Events created by appservman on page 432

Appservman - how it works

Appservman is a service that starts, stops and monitors the application server. It also optionally restarts the application server in the event that the latter fails. Appservman can be controlled not just from nodes running the application server, but also from any other node running conman.

It is launched as a service by netman when starting HCL Workload Automation, and it itself then launches the application server. Netman also launches it when the conman startappserver command is run.

Appservman is stopped when HCL Workload Automation is shut down. In addition, Netman stops both the application server and appservman when you use the conman stopappserver command, or, on Windows™ only, when you issue the Shutdown –appsrv command.

While it is running appservman monitors the availability of the application server, sending events that report the status of the application server. If the automatic restart facility is enabled, and the application server fails, the service determines from the restart policy indicated in the `localopts` options if it is to restart the application server. If the policy permits, it will restart the application server, and send events to report its actions.

Using the startappserver and stopappserver commands to stop the application server and appservman.

Controlling appservman

Appservman is controlled by the following local options (in the `localopts` file):

Appserver auto restart

Determines if the automatic restart facility is enabled.

The default is *yes*. To disable the option set it to *no*.

Appserver check interval

Determines how frequently the service checks on the status of the application server. You should not set this value to less than the typical time it takes to start the application server on the computer.

The default is every 3 minutes.

Appserver min restart time

Determines the minimum time that must elapse between failures of the application server for the automatic restart to work. This option stops appservman from immediately restarting the application server if it fails on initial startup or when being restarted.

The default is 2 minutes.

Appserver max restarts

Determines the maximum number of times that appservman will automatically restart the application server within a time frame determined by you (Appserver count reset interval).

The default is 5 restarts.

Appserver count reset interval

Determines the time frame for the maximum number of restarts (Appserver max restarts).

The default is 24 hours.

How to use the options

The default settings are a good starting point. Follow the indications below if you are not satisfied that the settings are maintaining the correct availability of the application server:

- If the application server is not restarting after failure, check the following:
  - That the Appserver auto restart is set to `yes`.
  - That the Appserver check interval is not set to too high a value. For example, if this value is set to `50` minutes, instead of the default `3`, an early failure of the application server might wait 45 minutes before being restarted.
  - That Appserver min restart time is sufficient for the application server to fully restart. If, when the server checks the status of the application server, it finds that the application server is still starting up, in some circumstances it is not able to distinguish the starting-up state from the failed state, will report it as failed, and try and restart it again. With the same result. This will continue until Appserver max restarts is exceeded. If this is the case, make Appserver min restart time larger.
- If the application server is failing infrequently, but after several failures is not restarting, set the Appserver max restarts option to a higher value or the Appserver count reset interval to a lower value, or both. In this case it might be advantageous to study the pattern of failures and tailor these options to give you the required availability

Starting and stopping the application server and appservman

If you need to stop and restart WebSphere Application Server Liberty, for example to implement a change in the application server configuration, use the following conman commands:

conman stopappserver[*domain*!]*workstation* [;wait]

This command stops the application server and appservman. You can optionally stop the application server on a remote workstation. The optional ;wait parameter tells conman to suspend processing until the command reports that both the application server and the service have stopped.

conman startappserver[*domain*!]*workstation* [;wait]

This command starts the application server and appservman. You can optionally start the application server on a remote workstation. The optional ;wait parameter tells conman to suspend processing until the command reports that both the application server and the service are up and running.

To stop and start the application server without stopping appservman, see Application server - starting and stopping on page 428.

Monitoring the application server status

To see the current status of the application server at any time view the STATE field in the workstation details.

This field contains a string of characters that provide information about the statuses of objects and processes on the workstation. The state of the application server is a one-character flag in this string, which has one of the following values if the application server is installed:

```
[A|R]
```

where:

A

WebSphere Application Server Liberty is running.

R

WebSphere Application Server Liberty is restarting.

If WebSphere Application Server Liberty is down or if it was not installed, neither value of the flag is present in the STATE entry.

Obtaining information about application server failures

Appservman does not provide information about why the application server has failed. To obtain this information, look in the application server log files .

Events created by appservman

Appservman sends an event called *ApplicationServerStatusChanged* from the `TWSObjectsMonitor` provider to the configured event monitoring process every time the status of the application server changes.

Application server - encrypting the profile properties files

For more information about encrypting passwords in WebSphere Application Server Liberty profile properties files, see the related documentation.

This utility requires the JAVA_HOME environment variable to be set. If you do not have Java installed, you can optionally use the Java version provided with the product and available in:

HCL Workload Automation

`<`*`INST_DIR`*`>/TWS/JavaExt/jre/jre`

Dynamic Workload Console

`<`*`DWC_INST_DIR`*`>/java/jre/bin`

Application server - configuration files backup and restore

On UNIX operating systems, all configuration files are stored in the <*TWA_DATA_DIR*>`/usr` and <*DWC_DATA_dir*>`/usr`.

On Windows operating systems, backup the files in <*TWA_home*>`\usr` and <*DWC_home*>`\usr`.

Application server - changing the host name or TCP/IP ports

To modify the host name of the computer where the application server is installed, or the TCP/IP ports it uses, use the following templates:

host_variables.xml

Specify the hostname in the `variable name` property.

ports_variable.xml

Edit the following properties as required:

- host.http.port
- host.https.port
- host.bootstrap.port
- host.bootstrap.port.sec

To disable a port, set its value to `-1`.

Edit the files as required using the following basic procedure:

1. Copy the template file from the `templates` folder to a working folder.
2. Edit the template file in the working folder with the desired configuration.
3. Optionally, create a backup copy of the relevant configuration file present in the `overrides` directory in a different directory. Ensure you do not copy the backup file in the path where the template files are located.
4. Copy the updated template file to the `overrides` folder. Maintaining the original folder structure is not required.
5. Changes are effective immediately.

For more information about the procedure, see Configuring HCL Workload Automation using templates on page 422.
For more information about WebSphere Liberty configuration, see the related documentation, for example HTTP Endpoint (httpEndpoint) and Configuring an httpEndpoint to use an SSL configuration.

Application server - changing the trace properties

To modify the trace properties, modify the `trace.xml` template as necessary.

Templates for the master domain manager are stored in the following paths:

On UNIX operating systems

TWA_home`/usr/servers/engineServer/configDropins/templates`

On Windows operating systems

TWA_home`\usr\servers\engineServer\configDropins\templates`

Templates for the Dynamic Workload Console are stored in the following paths:

On UNIX operating systems

DWC_home`/usr/servers/dwcServer/configDropins/templates`

On Windows operating systems

DWC_home`\usr\servers\dwcServer\configDropins\templates`

When you edit the file with your customized settings for the master domain manager, move it to the following paths:

On UNIX operating systems

TWA_DATA_DIR`/usr/servers/engineServer/configDropins/overrides`

On Windows operating systems

TWA_home`\usr\servers\engineServer\configDropins\overrides`

When you edit the file with your customized settings for the Dynamic Workload Console, move it to the following paths:

On UNIX operating systems

DWC_DATA_dir`/usr/servers/dwcServer/configDropins/overrides`

On Windows operating systems

DWC_home`\usr\servers\dwcServer\configDropins\overrides`

To modify the `trace.xml` template, follow this procedure:

1. Copy the template file from the `templates` folder to a working folder.
2. Edit the template file in the working folder with the desired configuration.
3. Optionally, create a backup copy of the relevant configuration file present in the `overrides` directory in a different directory. Ensure you do not copy the backup file in the path where the template files are located.
4. Copy the updated template file to the `overrides` folder. Maintaining the original folder structure is not required.
5. Changes are effective immediately.

Chapter 9. Administering an IBM i dynamic environment

On overview on how to administer the HCL Workload Automation IBM i dynamic environment.

To begin scheduling jobs with advanced options on IBM i agents, the agents must be configured.

Configuring the agent on IBM i systems

An overview on how to configure the agent on IBM i systems.

The configuration settings of the agent are contained in the `JobManager.ini` file and in the `JobManagerGW.ini` file (for the path of these files, see the section about installation paths in *HCL Workload Automation: Planning and Installation*).

The configuration files are made up of many different sections. Each section name is enclosed between square brackets and each section includes a sequence of `variable = value` statements.

You can customize properties for the following:

- Log properties
- Trace properties when the agent is stopped. You can also customize traces when the agent is running using the procedure described in Configuring trace properties when the agent is running on page 78.
- Native job executor
- Java™ job executor
- Resource advisor agent
- System scanner

On IBM i systems, the log messages are written in the following file:

```
TWA_DATA_DIR>/stdlist/JM/JObManager_message.log
```

On IBM i systems, the trace messages are written in the following files:

```
<TWA_DATA_DIR>/TWS/stdlist/JM/ITA_trace.log
<TWA_DATA_DIR>/TWS/stdlist/JM/JobManager_trace.log
<TWA_DATA_DIR>/TWS/stdlist/JM/javaExecutor0.log
```

Not all the properties in the `JobManager.ini` file and in the `JobManagerGW.ini` file can be customized. For a list of the configurable properties, see the following sections:

- Configuring log message properties [JobManager.Logging.cclog] on page 75.
- Configuring trace properties when the agent is stopped [JobManager.Logging.cclog] on page 77.
- Configuring common launchers properties [Launchers] on page 81.
- Configuring properties of the native job launcher [NativeJobLauncher] on page 83.
- Configuring properties of the Java job launcher [JavaJobLauncher] on page 86.
- Configuring properties of the Resource advisor agent [ResourceAdvisorAgent] on page 87.
- Configuring properties of the System scanner [SystemScanner] on page 89

Note: In the `JobManager.ini` file and in the `JobManagerGW.ini` file you must refer to Java 64 bit version.

Configuring log message properties [JobManager.Logging.cclog]

About this task

To configure the logs, edit the [JobManager.Logging.cclog] section in the `JobManager.ini` file. This procedure requires that you stop and restart the HCL Workload Automation agent

The section containing the log properties is named:

```
[JobManager.Logging.cclog]
```

You can change the following properties:

JobManager.loggerhd.fileName

The name of the file where messages are to be logged. the default value is

On Windows operating systems

`TWA_home\stdlist\JM\JOBMANAGER-FFDC`

On UNIX operating systems

`$(`*TWA_DATA_DIR*`)/stdlist/JM/JobManager_message.log`

JobManager.loggerhd.maxFileBytes

The maximum size that the log file can reach. The default is **1024000** bytes.

JobManager.loggerhd.maxFiles

The maximum number of log files that can be stored. The default is **3**.

JobManager.loggerhd.fileEncoding

By default, log files for the agent are coded in UTF-8 format. If you want to produce the log in a different format, add this property and specify the required codepage.

JobManager.loggerfl.level

The amount of information to be provided in the logs. The value ranges from 3000 to 7000. Smaller numbers correspond to more detailed logs. The default is **3000**.

JobManager.ffdc.maxDiskSpace

Exceeding this maximum disk space, log files collected by the first failure data capture mechanism are removed, beginning with the oldest files first.

JobManager.ffdc.baseDir

The directory to which log and trace files collected by the ffdc tool are copied. The default directory is

On Windows operating systems

`TWA_home\stdlist\JM\JobManager_message.log`

On UNIX operating systems

`$(`*TWA_DATA_DIR*`)/stdlist/JM/JobManager_message.log`

JobManager.ffdc.filesToCopy

Log and trace files (`JobManager_message.log` and `JobManager_trace.log`) collected by the ffdc tool located in `<`*TWA_home*`>\TWS\stdlist\JM`. The files are available in the following paths:

On Windows operating systems

- TWA_home`/TWS/stdlist/JM/JobManager_message.log`
- TWA_home`/TWS/stdlist/JM/JobManager_trace.log`

On UNIX operating systems

- `$(`*TWA_DATA_DIR*`)/stdlist/JM/JobManager_message.log`
- `$(`*TWA_DATA_DIR*`)/stdlist/JM/JobManager_trace.log`

When a message is logged (JobManager.ffdc.triggerFilter = JobManager.msgIdFilter) that has an ID that matches the pattern "AWSITA*E" (JobManager.msgIdFilter.msgIds = AWSITA*E), which corresponds to all error messages, then the log and trace files (JobManager.ffdc.filesToCopy = "/opt/HCL/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_message.log" "/opt/HCL/TWA_<TWS_user>/TWS/stdlist/JM/JobManager_trace.log") are copied (JobManager.ffdc.className = ccg_ffdc_filecopy_handler) to the directory `JOBMANAGER-FFDC` (JobManager.ffdc.baseDir = /opt/HCL/TWA_<TWS_user>/TWS/stdlist/JM/JOBMANAGER-FFDC). If the files copied exceed 10 MB (JobManager.ffdc.maxDiskSpace = 10000000), then the oldest files are removed first (JobManager.ffdc.quotaPolicy = QUOTA_AUTODELETE).

After installing the z-centric agent or dynamic agent on Windows 2012, the `JobManager_message.log` might not be created. In this case, perform the following procedure:

1. Stop the agent.
2. Create a backup copy of `JobManager.ini`, and edit the original file by changing the row:

   ```
   JobManager.loggerhd.className = ccg_multiproc_filehandler
   ```

   to

   ```
   JobManager.loggerhd.className = ccg_filehandler
   ```
3. Restart the agent.

Configuring trace properties when the agent is stopped [JobManager.Logging.cclog]

How to configure the trace properties when the agent is stopped.

To configure the trace properties when the agent is stopped, edit the [JobManager.Logging] section in the `JobManager.ini` file and then restart the HCL Workload Automation agent.

The section containing the trace properties is named:

```
[JobManager.Logging.cclog]
```

You can change the following properties:

JobManager.trhd.fileName

The name of the trace file. the default path is as follows:

On Windows operating systems

`TWA_home/TWS/stdlist/JM/JobManager_trace.log`

On UNIX operating systems

`$(`*TWA_DATA_DIR*`)/stdlist/JM/JobManager_trace.log`

JobManager.trhd.maxFileBytes

The maximum size that the trace file can reach. The default is 10240000 bytes.

JobManager.trhd.maxFiles

The maximum number of trace files that can be stored. The default is 5.

JobManager.trfl.level

Determines the type of trace messages that are logged. Change this value to trace more or fewer events, as appropriate, or on request from HCL Software Support. Valid values are:

DEBUG_MAX

Maximum tracing. Every trace message in the code is written to the trace logs.

INFO

All *informational*, *warning*, *error* and *critical* trace messages are written to the trace. The default value.

WARNING

All *warning*, *error* and *critical* trace messages are written to the trace.

ERROR

All *error* and *critical* trace messages are written to the trace.

CRITICAL

Only messages which cause the agent to stop are written to the trace.

The output trace (`JobManager_trace.log`) is provided in XML format.

After installing the z-centric agent or dynamic agent on Windows 2012, the `JobManager_trace.log` might not be created. In this case, perform the following procedure:

1. Stop the agent.
2. Create a backup copy of `JobManager.ini`, and edit the original file by changing the row:

   ```
   JobManager.trhd.className = ccg_multiproc_filehandler
   ```

to

```
JobManager.trhd.className = ccg_filehandler
```

3. Restart the agent.

Configuring trace properties when the agent is running

Use the **twstrace** command to set the trace on the agent when it is running.

Using the **twstrace** command, you can perform the following actions on the agent when it is running:

- See command usage and verify version on page 79.
- Enable or disable trace on page 79.
- Set the traces to a specific level, specify the number of trace files you want to create, and the maximum size of each trace file. See Set trace information on page 79.
- Show trace information on page 80.
- Collect trace files, message files, and configuration files in a compressed file using the command line. See Collect trace information on page 80.

You can also configure the traces when the agent is not running by editing the [JobManager.Logging] section in the `JobManager.ini` file as described in Configuring the agent on page 71. This procedure requires that you stop and restart the agent.

twstrace command

Use the **twstrace** command to configure traces, and collect logs, traces, and configuration files (ita.ini and jobManager.ini) for agents. You collect all the information in a compressed file when it is running without stopping and restarting it.

See command usage and verify version

To see the command usage and options, use the following syntax.

Syntax

twstrace -u | **-v**

Parameters

-u

Shows the command usage.

-v

Shows the command version.

Enable or disable trace

To set the trace to the maximum or minimum level, use the following syntax.

Syntax

twstrace -enable | **-disable**

Parameters

-enable

Sets the trace to the maximum level. The maximum level is **1000**.

-disable

Sets the trace to the minimum level. The minimum level is **3000**.

Set trace information

To set the trace to a specific level, specify the number of trace files you want to create, and the maximum size the trace files can reach, use the following syntax.

Syntax

twstrace [**-level** <level_number>] [**-maxFiles** <files_number>] [**-maxFileBytes** <bytes_number>]

Parameters

-level <level_number>

Sets the trace level. Specify a value in the range from 1000 to 3000, which is also the default value. Note that if you set this parameter to 3000, you have the lowest verbosity level and the fewest trace messages. To have a better trace level, with the most verbose trace messages and the maximum trace level, set it to **1000**.

-maxFiles <files_number>

Specify the number of trace files you want to create.

-maxFileBytes <bytes_number>

Set the maximum size in bytes that the trace files can reach. The default is **1024000** bytes.

Show trace information

To display the current trace level, the number of trace files, and the maximum size the trace files can reach, use the following syntax.

Syntax

twstrace -level | **-maxFiles** | **-maxFileBytes**

Parameters

-level

See the trace level you set.

-maxFiles

See the number of trace files you create.

-maxFileBytes

See the maximum size you set for each trace file

Example

Sample

The example shows the information you receive when you run the following command:

```
twstrace -level -maxFiles -maxFileBytes
```

```
AWSITA176I The trace properties are: level="1000",
max files="3", file size="1024000".
```

Collect trace information

To collect the trace files, the message files, and the configuration files in a compressed file, use the following syntax.

Syntax

twstrace -getLogs [**-zipFile** <compressed_file_name>] [**-host** <host_name>] [**-protocol** {http | https } [**-port** <port_number>][**-iniFile** <ini_file_name>]

Parameters

-zipFile <compressed_file_name>

Specify the name of the compressed file that contains all the information, that is logs, traces, and configuration files (ita.ini and jobManager.ini) for the agent. The default is **logs.zip**.

-host <host_name>

Specify the host name or the IP address of the agent for which you want to collect the trace. The default is **localhost**.

-protocol http|https

Specify the protocol of the agent for which you are collecting the trace. The default is the protocol specified in the **.ini** file of the agent.

-port <port_number>

Specify the port of the agent. The default is the port number of the agent where you are running the command line.

-iniFile <ini_file_name>

Specify the name of the **.ini** file that contains the SSL configuration of the agent for which you want to collect the traces. If you are collecting the traces for a remote agent for which you customized the security certificates, you must import the certificate on the local agent and specify the name of the **.ini** file that contains this configuration. To do this, perform the following actions:

1. Extract the certificate from the keystore of the remote agent.
2. Import the certificate in a local agent keystore. You can create an ad hoc keystore whose name must be **TWSClientKeyStore.kdb**.

3. Create an **.ini** file in which you specify:
    - **0** in the **tcp_port** property as follows:

      ```
      tcp_port=0
      ```

    - The port of the remote agent in the **ssl_port** property as follows:

      ```
      ssl_port=<ssl_port>
      ```

    - The path to the keystore you created in Step 2 on page 441 in the **key_repository_path** property as follows:

      ```
      key_repository_path=<local_agent_keystore_path>
      ```

Configuring common launchers properties [Launchers]

About this task

In the `JobManager.ini` file, the section containing the properties common to the different launchers (or executors) is named:

```
[Launchers]
```

The following properties are available:

BaseDir

The installation path of the HCL Workload Automation agent. Do not modify this value.

CommandHandlerMinThreads

Indicates the maximum number of commands that can be run on the agent concurrently. Limits to the number of jobs vary depending on the resources of your workstation, however consider that operations on comdhandler are usually short. The default is **20**. Usually, there is no need to modify this setting, even if you plan a very high workload on the agent. You might want to change it if many commands are run concurrently on the agent, for example, many concurrent requests to retrieve job logs.

CommandHandlerMaxThreads

Indicates the maximum number of commands that can be run on the agent concurrently. Limits to the number of jobs vary depending on the resources of your workstation, however consider that operations on comdhandler are usually short. The default is **100**. Usually, there is no need to modify this setting, even if you plan a very high workload on the agent. You might want to change it if many commands are run concurrently on the agent, for example, many concurrent requests to retrieve job logs.

CpaHeartBeatTimeSeconds

The polling interval in seconds used to verify if the **agent** process is still up and running. If the agent process is inactive the product stops also the **JobManager** process. The default is **30**. Modify only if you use dynamic pools with CPU-based requirements or optimization policies. With a lower value, the agent reacts quickly to CPU modifications, but this might cause unstable values in case of CPU spikes. Lower values causes a higher use of resources on the agent.

DirectoryPermissions

The access rights assigned to the agent for creating directories when running jobs. The default is **0755**. Supported values are UNIX-format entries in hexadecimal notation.

DownloadDir

The name of the directory where the fix pack installation package or upgrade eImage for dynamic agents is downloaded during the centralized agent update process. If not specified, the following default directory is used:

On Windows operating systems:

`TWA_home\TWS\stdlist\JM\download`

On UNIX operating systems:

TWA_DATA_DIR`/TWS/stdlist/JM/download`

The centralized agent update process does not apply to z-centric agents.

ExecutorsMaxThreads

Specifies the maximum number of jobs the dynamic agent can run concurrently. For example, to allow the dynamic agent to run a maximum of 500 jobs concurrently, set this parameter to **500**. The default is **400**.

ExecutorsMinThreads

Specifies the minimum number of jobs the dynamic agent can run concurrently. For example, to allow the dynamic agent to run a minimum of 500 jobs concurrently, set this parameter to **500**. The default is **38**. Modify if the number of expected concurrent jobs is much higher than 38. The agent dynamically allocates more threads if necessary, until it reaches the value specified in **ExecutorsMaxThreads**.

FilePermissions

The access rights assigned to the agent for creating files when running jobs. The default is **0755**. Supported values are UNIX-format entries in hexadecimal notation.

MaxAge

The number of days that job logs are kept (in path *`TWA_home`*`/TWS/stdlidst/JM`) before being deleted. The default is **30**. Possible values range from a minimum of 1 day.

NotifierMaxThreads

Notifier threads are in charge of notifying the dynamic workload broker of each status change in a job. This parameter specifies the maximum number of job status changes that can be notified to the dynamic workload broker.

NotifierMinThreads

Notifier threads are in charge of notifying the dynamic workload broker of each status change in a job. This parameter specifies the minimum number of job status changes that can be notified to the dynamic workload broker. The default value is **3**. Modify this parameters only in case of unexpected errors and after consulting with software support team.

SpoolDir

The path to the folder containing the jobstore and outputs. The default is:

On Windows operating systems

`TWA_home/TWS/stdlist/JM`

On UNIX operating systems

`$(`*`TWA_DATA_DIR`*`/stdlist/JM`

StackSizeBytes

The size of the operating system stack in bytes. The default is **DEFAULT**, meaning that the **agent** uses the default value for the operating system. Do not modify this parameter unless instructed to do so by the software support team. Incorrect values can cause the agent to crash.

Configuring properties of the native job launcher [NativeJobLauncher]

About this task

In the `JobManager.ini` file, the section containing the properties of the native job launcher is named:

```
[NativeJobLauncher]
```

You can change the following properties:

AllowRoot

Applies to UNIX™ systems only. Specifies if the root user can run jobs on the agent. It can be `true` or `false`. The default is false. This property does not apply to IBM i, use the AllowQSECOFR option instead

AllowQECOFR

Applies to IBM i systems only. Specifies if QSECOFR user can run jobs on the agent. It can be `true` or `false`. The default is `true`. Add a line like AllowQSECOFR = `false` to the JobManager.ini file to deny job execution to QSECOFR.

CheckExec

If `true`, before launching the job, the agent checks both the availability and the execution rights of the binary file. The default is `true`.

DefaultWorkingDir

Specifies the working directory of native jobs. You can also specify the value for the working directory when creating or editing the job definition in the Graphical Designer. When specified in the Graphical Designer, this value overrides the value specified for the **DefaultWorkingDir** property. If you do not specify any working directories, the `<`*`TWS_home`*`>\bin` directory is used.

JobUnspecifiedInteractive

Applies to Windows™ operating systems only. Specifies if native jobs are to be launched in interactive mode. It can be `true` or `false`. The default is `false`.

KeepCommandTraces

Set to `true` to store the traces of the method invocation for actions performed on a job definition, for example, when selecting from a picklist. These files are stored in the path `/opt/HCL/TWA_<TWS_user>/TWS/stdlist/JM/r3batch_cmd_exec`. The default setting is `false`.

KeepJobCommandTraces

Set to `true` to store the traces of the method invocation for actions performed on a job instance, for example, viewing a spool list. These files are stored in the .zip file of the job instance. The default setting is `true`.

LoadProfile

Applies to agents on Windows servers only. Specifies if the user profile is to be loaded. It can be `true` or `false`. The default is `true`.

MonitorQueueName

Specifies the name of the queue where the IBM i jobs are monitored. If you do not specify this property, the default queue (QBATCH) is used.

PortMax

The maximum range of the port numbers used by the task launcher to communicate with the Job Manager. The default is 0, meaning that the operating system assigns the port automatically.

PortMin

The minimum range of the port numbers used by the task launcher to communicate with the Job Manager. The default is 0, meaning that the operating system assigns the port automatically.

PostJobExecScriptPathName

The fully qualified path of the script file that you want to run when the job completes. By default, this property is not present in the `JobManager.ini` file. If you do not specify any file path or the script file doesn't exist, no action is taken.

This property applies to dynamic agent and z/OS agent. For details about running a script when a job completes, see *User's Guide and Reference*.

PromotedNice

Used in workload service assurance. This property is not supported on the Agent for z/OS.

For UNIX and Linux operating systems only, assigns the priority value to a critical job that needs to be promoted so that the operating system processes it before others. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time.

Boundary values vary depending upon each specific platform, but generally lower values correspond to higher priority levels and vice versa. The default is -1.

Be aware that:

- The promotion process is effective with negative values only. If you set a positive value, the system runs it with the -1 default value.
- An out of range value (for example -200), prompts the operating system to automatically promote the jobs with the lowest allowed nice value.
- Overusing the promotion mechanism (that is, defining an exceedingly high number of jobs as mission critical and setting the highest priority value here) might overload the operating system, negatively impacting the overall performance of the workstation.

PromotedPriority

Used in workload service assurance. This property is not supported on the Agent for z/OS.

For Windows operating systems only, sets to this value the priority by which the operating system processes a critical job when it is promoted. Applies to critical jobs or predecessors that need to be promoted so that they can start at their critical start time. Valid values are:

- `High`
- `AboveNormal` (the default)
- `Normal`
- `BelowNormal`
- `Low` or `Idle`

Note that if you a set a lower priority value than the one non-critical jobs might be assigned, no warning is given.

RequireUserName

When `true`, requires that you add the user name in the JSDL job definition.

When `false`, runs with the user name used by job manager, that is:

- *TWS_user* on UNIX™ and Linux™ systems
- The local system account on Windows™ systems

The default is `false`.

RunExecutablesAsIBMiJobs

If you set this property to `true`, you can define IBM i jobs as generic jobs without using the XML definition. Generic jobs are automatically converted to IBM i jobs. As a side effect, generic jobs cannot be run when this parameter is enabled (`RunExecutablesAsIBMiJobs=true`). There is no default value because this property is not listed in the `JobManager.ini` file after the agent installation.

If you set this property to `true`, ensure that the user you used to install the agent has been granted the `*ALLOBJ` special authority.

RunInteractiveJobOnInvalidSession

Applies only to native and executable jobs starting interactive programs running on dynamic agents installed on Windows operating systems. Interactive programs run only if the job user has an active session open when the job runs. If there is no active session for the job user, the job behavior is defined by this property, as follows. Set the property to `true` to enable jobs to start interactive programs even if there is no active session for the job user. Set the property to `false` to prevent jobs from starting interactive programs if there is no active session for the job user.

ScriptSuffix

The suffix to be used when creating the script files. It is:

`.cmd`

For Windows™

`.sh`

For UNIX™

VerboseTracing

Enables verbose tracing. It is set to `true` by default.

Configuring properties of the Java™ job launcher [JavaJobLauncher]

About this task

In the `JobManager.ini` file, the section containing the properties of the Java™ job launcher is named:

```
[JavaJobLauncher]
```

You can change the following property:

JVMOptions

The options to provide to the Java™ Virtual Machine used to start job types with advanced options. Supported keywords for establishing a secure connection are:

- htttps.proxyHost
- https.proxyPort

Supported keywords for establishing a non-secure connection are:

- Dhttp.proxyHost
- Dhttp.proxyPort

For example, to set job types with advanced options, based on the default JVM http protocol handler, to the unauthenticated proxy server called with name myproxyserver.mycompany.com, define the following option:

```
JVMOptions = -Dhttp.proxyHost=myproxyserver.mycompany.com -Dhttp.proxyPort=80
```

Configuring properties of the Resource advisor agent [ResourceAdvisorAgent]

About this task

In the `JobManager.ini` and `JobManagerGW.ini` files, the section containing the properties of the Resource advisor agent is named:

```
[ResourceAdvisorAgent]
```

You can change the following properties:

BackupResourceAdvisorUrls

The list of URLs returned by the HCL Workload Automation master in a distributed environment or by the dynamic domain manager either in a z/OS or in a distributed environment. The agent uses this list to connect to the master or dynamic domain manager.

CPUScannerPeriodSeconds

The time interval that the Resource advisor agent collects resource information about the local CPU. The default value is every 10 seconds.

FullyQualifiedHostname

The fully qualified host name of the agent. It is configured automatically at installation time and is used to connect with the master in a distributed environment or with the dynamic domain manager in a z/OS or in a distributed environment. Edit only if the host name is changed after installation.

NotifyToResourceAdvisorPeriodSeconds

The time interval that the Resource advisor agent forwards the collected resource information to the Resource advisor. The default value is every 119 seconds.

ResourceAdvisorUrl

JobManager.ini

The URL of the master in a distributed environment, or of the dynamic domain manager in a z/OS or in a distributed environment, that is hosting the agent. This URL is used until the server replies with the list of its URLs. The value is `https://$(`*`tdwb_server`*`):$(`*`tdwb_port`*`)/JobManagerRESTWeb/JobScheduler/resource`, where:

$(*tdwb_server*)

is the fully qualified host name of the master in a distributed environment or of the dynamic domain manager either in a z/OS or in a distributed environment.

$(*tdwb_port*)

is the port number of the master in a distributed environment or of the dynamic domain manager either in a z/OS or in a distributed environment.

It is configured automatically at installation time. Edit only if the host name or the port number are changed after installation, or if you do not use secure connection (set to `http`). If you set the port number to zero, the resource advisor agent does not start. The port is set to zero if at installation time you specify that you will not be using the master in a distributed environment or the dynamic domain manager either in a z/OS or in a distributed environment.

In a distributed environment, if **-gateway** is set to either `local` or `remote`, then this is the URL of the dynamic agent workstation where the gateway resides and through which the dynamic agents communicate. The value is `https://$(tdwb_server):$(tdwb_port)/ita/JobManagerGW/JobManagerRESTWeb/JobScheduler/resource`, where:

$(tdwb_server)

The fully qualified host name of the dynamic agent workstation where the gateway resides and through which the dynamic agent communicates with the dynamic workload broker.

$(tdwb_port)

The port number of the dynamic agent workstation where the gateway resides and through which the dynamic agent communicates with the dynamic workload broker.

JobManagerGW.ini

In a distributed environment, if **-gateway** is set to `local`, then **ResourceAdvisorUrl** is the URL of the master or dynamic domain manager. The value is `https://$(tdwb_server):$(tdwb_port)/JobManagerRESTWeb/JobScheduler/resource`, where:

$(tdwb_server)

The fully qualified host name of the master or dynamic domain manager.

$(tdwb_port)

The port number of the master or dynamic domain manager.

ScannerPeriodSeconds

The time interval that the Resource advisor agent collects information about all the resources in the local system other than CPU resources. The default value is every 120 seconds.

The resource advisor agent, intermittently scans the resources of the machine (computer system, operating system, file systems and networks) and periodically sends an update of their status to the master or dynamic domain manager either in a z/OS or in a distributed environment.

The CPU is scanned every `CPUScannerPeriodSeconds` seconds, while all the other resources are scanned every `ScannerPeriodSeconds` seconds. As soon as one of the scans shows a significant change in the status of a resource, the resources are synchronized with the master in a distributed environment or the dynamic domain manager either in a z/OS or in a distributed environment. The following is the policy followed by the agent to tell if a resource attribute has significantly changed:

- A resource is added or deleted
- A string attribute changes its value

- A CPU value changes by more than `DeltaForCPU`
- A file system value changes by more than `DeltaForDiskMB` megabytes
- A Memory value changes by more than `DeltaForMemoryMB` megabytes

If there are no significant changes, the resources are synchronized with the HCL Workload Automation master in a distributed environment or with thedynamic domain manager either in a z/OS or in a distributed environment every `NotifyToResourceAdvisorPeriodSeconds` seconds.

Configuring properties of the System scanner [SystemScanner]

About this task

In the `JobManager.ini` file, the section containing the properties of the System scanner is named:

```
[SystemScanner]
```

You can change the following properties:

CPUSamples

The number of samples used to calculate the average CPU usage. The default value is 3.

DeltaForCPU

The change in CPU usage considered to be significant when it becomes higher than this percentage (for example, DeltaForCPU is 20 if the CPU usage changes from 10 percent to 30 percent). The default value is 20 percent.

DeltaForDiskMB

The change in use of all file system resources that is considered significant when it becomes higher than this value. The default value is 100 MB.

DeltaForMemoryMB

The change in use of all system memory that is considered significant when it becomes higher than this value. The default value is 100 MB.

Configuring to schedule job types with advanced options

About this task

You can define job types with advanced options by using the related configuration files. The options you define in the configuration files apply to all job types with advanced options of the same type. You can override these options when defining the job by using the Dynamic Workload Console or, if you are in a distributed environment, the **composer** command.

Configuration files are available on each dynamic agent in TWA_home/TWS/JavaExt/cfg for the following job types with advanced options:

Table 71. Configuration files for job types with advanced options

Job type	File name	Keyword
- Database job type - MSSQL Job	DatabaseJobExecutor.properties	Use the `jdbcDriversPath` keyword to specify the path to the JDBC drivers. Define the keyword so that it points to the JDBC jar files directory, for example: `jdbcDriversPath=c:\\mydir\\jars\\jdbc` The JDBC jar files must be located in the specified directory or its subdirectories. Ensure you have list permissions on the directory and its sub subdirectories. **Note:** For the MSSQL database, use version 4 of the JDBC drivers.
Java™ job type	JavaJobExecutor.properties	Use the `jarPath` keyword to specify the path to the directory where the jar files are stored. This includes all jar files stored in the specified directory and all sub directories.
J2EE job type	J2EEJobExecutorConfig.properties	For more information about the J2EE job type, see the topic about configuring to schedule J2EE jobs in the *HCL Workload Automation: Administration Guide*.

Chapter 10. Performance

This chapter provides information about issues that impact performance. Use this information both to prevent problems occurring and to help resolve problems that occur.

Network traffic

A full description of how a HCL Workload Automation network is structured, and how the different nodes communicate, is provided at the beginning of Network administration on page 277. In particular, see Optimizing the network on page 292 , which explains how to design and operate your HCL Workload Automation network to maximize performance.

Tracing

The performance of any workstation can be impacted by the level of tracing it has to perform.

The performance might also be impacted by the tracing activities on WebSphere Application Server Liberty.

Logging

The performance of any workstation can be impacted by the way the HCL Workload Automation logging mechanism uses memory. The default settings applied in this version are designed to ensure the maximum performance. However, because these defaults are different from the defaults in earlier versions, if you are experiencing performance problems, it is advisable to check that these settings have not been in some way overwritten by the previous values.

Symphony file sizing

To calculate the size of the Symphony file and understand its impact on performance, see Avoiding full file systems on page 337.

Tuning a UNIX™ domain manager to handle large numbers of fault-tolerant agents

The performance of domain managers on UNIX™ is impacted if they are overloaded with jobs. Improvements can be obtained by modifying the kernel parameters. The precise settings differ according to operating system, and you might need to test different settings to obtain optimum performance.

The following is an example of the kernel settings for Linux (kernel t3.10.0-514.e17.x86_64) to handle 500000 jobs per day workload:

```
data seg size=unlimited
scheduling priority=0
file size=unlimited
pending signals=124946
max locked memory=64
max memory size=unlimited
open files=105000
pipe size=8
```

```
POSIX message queues=819200
real-time priority=0
stack size=10240
cpu time=unlimited
max user processes=16384
virtual memory=unlimited
file locks=unlimited
```

Tuning job processing on a workstation

This section explains how to tune selected options in the HCL Workload Automation `localopts` file to improve HCL Workload Automation performance. These options control the period between successive instances of an activity. Table 72: Options for tuning job processing on a workstation on page 453 shows the activities to be tuned, the corresponding option that can be set in the `localopts` file, and how the changed value impacts performance.

Table 72. Options for tuning job processing on a workstation

Activity	Option	Impact on performance
batchman periodically scans the `Symphony` file for jobs ready to be processed.	***bm look***	In all these cases, a shorter time means more frequent scans, using more cpu resources, and impacting other processes that are running. However, it also means that for all activities waiting time is kept to a minimum. If throughput is important and the workstation has plenty of memory, try shortening the times. A longer period between successive activities means jobs take longer to run, because there are longer waits for each activity. However, the reduced frequency of the scans means that more memory is available for jobs because less is being used by these monitoring activities. Consider the meaning of the various options. If your objective is to run the jobs as quickly as possible, but you are not concerned about how quickly the information about completed jobs is distributed, you could reduce the wait periods for *bm look* and *jm read*, but increase the periods for the others. Alternatively, to speed up the overall job processing time (from initial job launch to the update with the completion status), you can tune *bm look*, *jm look*, and *mm read*.
jobman accesses the `Courier.msg` file to see if there are jobs that need to be launched.	***jm read***	
After having launched a job **jobman** checks periodically for job completion status.	***jm look***	
mailman looks periodically in the `Mailbox.msg` for completed jobs.	***mm read***	
batchman checks periodically in `Intercom.msg` for jobs that are complete so that it can update the `Symphony` file.	***bm read***	

If you decide to tune these setting do the following:

- Test the result in a test system before applying changes in your production environment. To get worthwhile results, the test environment must have the same characteristics as the production environment.
- Modify only the parameters that are necessary. It is better to modify one at a time and thoroughly test the change in performance, rather than changing all at once.
- Make a backup copy of the `localopts` file to ensure you can revert to the default options if necessary.

Stop and start the agent to activate changes applied to the `localopts` file.

Tuning plan replication

Tuning plan replication involves configuring specific settings to optimize the process of replicating plan data into the database. Plan replication ensures quick and reliable access to plan data stored in the database. Its main objective is to provide quick response times and increased overall performance. Sometimes, if this synchronization process is not configured appropriately for the size of your workload, you might notice some discrepancies in your environment, such as job status misalignment between the command line (conman) and the monitoring results obtained in the Dynamic Workload Console.

There are a few simple settings you can configure to optimize performance:

Configure the cache size

Add the following properties to the `TWSConfig.properties` file located in the following paths:

On Windows operating systems

<TWA_home>`\usr\servers\engineServer\resources\properties`

On UNIX operating systems

<*TWA_DATA_DIR*>`/usr/servers/engineServer/resources/properties`

```
#Custom property which defines the number of threads and queues needed to
handle the plan updates
com.ibm.tws.planner.monitor.subProcessors=10
```

```
#Custom property which optimizes the DB access for file
dependency status update
com.ibm.tws.planner.monitor.filecachesize=40000
```

```
#Customer property which optimizes the DB access for
job and job stream status update
com.ibm.tws.planner.monitor.cachesize=40000
```

In addition, follow the steps to increase the heap size settings (initialHeapSize = 2048 and maximumHeapSize = 4096) of the application server on the master domain manager as documented in the *Administration Guide*.

Tuning the database

Tuning the database requires specific steps which vary depending on the database. To find detailed and specific information consult the relevant product documentation.

Optimizing the replication of the Symphony file in the database

Tuning DB2 database configuration parameters to improve performance when the Symphony file is replicated in the database.

In a HCL Workload Automation environment where more than 200,000 jobs are scheduled to be submitted, there are several DB2 database configuration parameters than can be tuned to improve performance when the `Symphony` plan is replicated in the HCL Workload Automation database.

The following are the suggested values for a plan with more than 200,000 jobs:

```
LOGBUFSZ = 2150
DBHEAP = AUTOMATIC (or greater than LOGBUFSZ)

LOGFILSIZ = 3000
LOGPRIMARY = 200
LOGSECOND = 40

PAGE_AGE_TRGT_MCR = 120
```

In addition, increase the number of pages (NPAGES) of the **TWS_PLN_BUFFPOOL** parameter to 182000 and the TWS_BUFFPOOL parameter to 50000 by using the ALTER BUFFERPOOL command.

Before changing any of these values, refer to the information about tuning a DB2 database in the relevant product documentation in Product Documentation.

Too many manual job submissions

HCL Workload Automation is designed for maximum efficiency when handling jobs submitted using a scheduled plan. Consequently, it is less adapted to processing manually submitted jobs. Thus, performance can be improved by reducing the number of manually submitted jobs.

Too many file dependency checks

Each file dependency check has an impact on performance. If you design a plan that is constantly checking many file dependencies, you reduce the performance of the workstation where these jobs are being run.

If multiple “opens? files are being used as a dependency, use the “–a? (and) option. For example, to check if three home directories `/tom`, `/dick`, and `/harry` exist, before launching `myjob` issue the following:

```
job2 opens "/users" (-d %p/tom -a -d %p/dick -a -d %p/harry)
```

This checks for all three directories at the same time, instead of looking for each directory separately.

Other factors that impact performance when evaluating file dependencies are the bm check parameters in the `localopts` file. These are documented in the Localopts summary section in the Administration Guide.

Note: In case of file dependencies applied to dynamic agent, it is suggested to keep ratio *number of file dependencies*/*bm check file* less than 0.7

Network configuration availability

After a system reboot, network services might be slow to start and if the agent starts when network services are not yet ready, the agent cannot work properly.

To prevent this problem, the agent waits sixty seconds and then retries to retrieve the network configuration. The agent repeats this operation for 5 times, that is, it waits for a total of 5 minutes for the network configuration to become available. If all attempts fail, the agent stops working.

You can configure the number of times the agent waits for network configuration availability in the **ITA** section of the `ita.ini` file, as follows:

1. Browse to the path where the `ita.ini` file is located:

   On UNIX™ operating systems

   TWA_DATA_DIR`/ITA/cpa/ita/ita.ini`

   On Windows™ operating systems

   TWA_home`TWS\ITA\cpa\config\ita.ini`
2. Edit the setting for the **net_conf_wait** parameter defining the number of times the agent retries to retrieve the network configuration, waiting sixty seconds between each attempt. If the number of attempts you have defined expires without the agent being able to retrieve the network configuration, the agent stops working.

Workload spreading

Whatever jobs you have to schedule, try and spread them out through the production period so that there is no concentration in any one moment. Try also to avoid scheduling activities during times when normal user traffic in the network is very heavy, for example during the morning when users commence working and deal with accumulated emails.

Failure to do this might cause a bottleneck at the Mailbox.msg queue, which causes delays in updating the Symphony file, which in turn creates delays in the availability of job statuses to conman, the Dynamic Workload Console.

Improving job-processing performance

The processing and monitoring of jobs on a workstation is controlled primarily by various parameters in the `localopts` file and the global options maintained by optman. These parameters are described in the *HCL Workload Automation: Planning and Installation Guide*.

If you are experiencing problems of performance when processing and monitoring jobs, contact HCL Software Support for advice about how to tune these parameters in your particular environment to improve performance.

Mailbox caching - advantages and disadvantages

Mailman uses a parameter in the `localopts` file to decide whether to cache mailbox messages: *mm cache mailbox*. This section explains the advantages and disadvantages of the on and off settings of this parameter.

Setting the *mm cache mailbox* parameter to *no*

This means that mailman has to make a separate read action for each message before processing it, and then a separate delete action after successfully processing the message. The I/O activity in performing these activities one message at a time is proportionally high for the amount of data being read. This has an impact on performance. On the other hand, the processing is simple, in that each message is read, processed, and then removed from the mailbox. Any failure of the system at any point means that at most one message is replayed and no data is lost.

Setting the *mm cache mailbox* parameter to *yes* (default)

This means that mailman reads a block of messages into cache memory, processes all of the messages, and then deletes all of them from the mailbox. The advantage in I/O time is clear; reading and deleting a sequential set of messages in one action is a much more efficient use of I/O time, than reading and deleting them one-by-one, meaning improved performance.

However, if there is a failure of mailman or the operating system, the cache is lost. On restarting, mailman rereads the set of messages that were previously in cache, some of which might already have been processed. For example, if mailman reads a block of 32 messages into cache and has processed 30 of them when a problem occurs, when mailman is restarted it rereads those 32 records and has to process 30 duplicates before being able to continue where it stopped.

Most events deal with job state changes, and these events can be repeated without creating any problems, and the critical events mechanism is able to deal with the others. However, there is an impact on performance while this recovery processing is going on, and if the in-built mechanisms cannot handle the message duplication, a more serious error might occur, ultimately involving the full or partial loss of the mailbox contents.

The number of messages being read in one action is configurable, using the parameter *mm cache size*. The default value for this parameter is 32 messages, and the maximum is 512. Setting this parameter to a value higher than the default increases performance during correct working, but decreases the performance in the event of a failure, for the reasons stated above. In addition, the additional cache means that the memory required by the HCL Workload Automation engine also increases. If you have a workstation with limited memory, or memory-heavy applications running, it might be counterproductive to increase the mailbox cache because the operating system might have to start paging the cache memory.

In conclusion, the default setting maximizes performance; only if you start losing events should you set it to *no*.

Setting the synch level parameter

This section describes the impact of the different settings of the *synch level* parameter in the `localopts` file. The *synch level* parameter only impacts UNIX™ environments.

The I/O activity performed by the HCL Workload Automation engine in managing plans, job streams, and jobs, consists in reading from and writing to the `Symphony` file and the event files (`Mailbox.msg`, `Intercom.msg`, and `Courier.msg`). When HCL Workload Automation writes to these files it has more than a straightforward *write* operation to perform. For example, when it writes to the `Mailbox.msg` file it performs the actions described in the following pseudo code:

```
TWS_write_event_lock {
     Lock Mailbox to write
}

TWS_write_event_update {
     Check Available Space
     Write Header
     Write Record
     Update Write Pointer
     Unlock Mailbox
}
```

Each action requires one or more write accesses to the disk. The way these actions are performed with the different synch level options is as follows:

synch level = high

Each write operation on the event files is immediately physically written to disk. This has a heavy impact on performance caused by the high I/O dependency.

synch level = medium

Each write event is considered as a single operation. For example, while `TWS_write_event_lock` contains only one action, `TWS_write_event_update` comprises five actions. With `synch level` at *medium*, the five actions in this write event would be completed in one physical disk access, thus drastically reducing the I/O overhead.

synch level = low (default)

The operating system decides how and when to synchronize the data to disk. The impact of this option is more difficult to assess, because the rules are different for each operating system and file system.

The fault-tolerant switch manager - impact on performance

This section describes the impact that the enablement of the fault-tolerant switch manager feature has on the performance of the general architecture and the individual system. The fault-tolerant switch manager is enabled by setting the `enSwfaultTol` global option to *yes*. When it is set, the master domain manager distributes messages to all fault-tolerant agents with *FullStatus* set to *yes*. This option has not dynamic capabilities and is not designed to work with broker agents.

Enabling this option impacts the following:

- Network traffic
- Disk space

Note: The fault-tolerant switch manager facility is only available if all of the workstations in the domain are at version 8.2, fix pack level 4, or higher.

Network Traffic

Network traffic is unchanged under normal conditions, but is increased during the replay phase, according to your choice and only under special conditions.

The replay phase is an essential part of the processing performed by the switchmgr command. It occurs when the new domain manager processes its Symphony file against its copies of the messages received, as it attempts to update its copy of the Symphony file.

Under normal conditions, the outbound reliability does not create any additional network traffic, because the messages are only stored for an eventual replay operation. The multiple inbound connections do not generate additional traffic because the traffic that was previously copied by the domain manager to the *FullStatus* member is now copied to the *FullStatus* members directly by the fault-tolerant agents.

During the replay phase, the connection protocol initiated by mailman on the backup domain manager includes a new phase for the replay of messages not sent by the failed domain manager. The impact of the message replay might be important, depending on the number of messages "trapped" in the old domain manager.

Disk Space

There are two places within the network where disk space use increases following the activation of the additional fault tolerance.

These places are as follows:

- On the single fault-tolerant agent. Here, in addition to the `tomaster.msg` queue, new queues are created for the other *FullStatus* fault-tolerant agents. These queues need not be considered, because the impact on a single agent is small.
- On the *FullStatus* fault-tolerant agents acting as backup domain managers. Here new ftbox message files are created. Upward traffic to the upper domain manager is in `ftbox/ftup.msg` and downward traffic to the lower domain manager is in `ftbox/ftdown.msg`.

Scalability

In an environment with large numbers of scheduling objects, the following impacts are felt:

- Impact on JnextPlan on page 460
- Impact on reporting on page 460
- Impact on event rule deployment on page 460

The resolution for these problems often includes making the following changes:

- Increasing application server heap size on page 461
- Increasing maximum DB2 log capacity on page 462

Impact on JnextPlan

The main impact on performance caused by a large network of workstations running many jobs over a production period of many days, is on JnextPlan. The key factor is the number of job stream instances that JnextPlan needs to handle. JnextPlan has to process each of these instances, and the time it takes to do so is a factor that can only be reduced by ensuring that the master domain manager and the database are on the most powerful computers possible, and that the communication, whether in local or remote, between the master domain manager and the database is maximized.

However, there are some specific measures that need to be taken as the number of jobs or job stream instances increases:

Number of jobs in the plan exceeds 40 000

In this event you need to increase the Java™ heap size used by the application server. The default is 512 MB, and you should at least double the heap size when job numbers exceed this level. Follow the procedure in Increasing application server heap size on page 461.

You have a large number of job stream instances in the plan

DB2®

The default DB2® transaction log files cannot handle more than the transactions generated by about 180 000 job stream instances. You need to change the parameters that control the log file sizes or the numbers of log files that can be created, or both. Follow the procedure in Increasing maximum DB2 log capacity on page 462.

Oracle

The number of transactions that can be managed by the Oracle log files depends on the way the Oracle database is configured. See the Oracle documentation for more details.

Note: If circumstances change and the number of job stream instances handled by JnextPlan falls below about 180 000, consider resetting the log and application server heap size settings to their default values, to avoid performance problems.

Impact on reporting

When a report is being processed, extra memory is required to handle large numbers of scheduling objects. The critical point is approximately 70 000 objects. This problem can be handled by increasing the Java™ heap size used by the application server. Follow the procedure in Increasing application server heap size on page 461.

Impact on event rule deployment

When deploying large numbers of event rules, extra memory is required. The critical point is approximately 8 000 rules. This problem can be handled by increasing the Java™ heap size used by the application server. Follow the procedure in Increasing application server heap size on page 461.

Increasing application server heap size

Follow this procedure to increase the Java™ heap size:

1. Log on to the computer where HCL Workload Automation is installed as the following user:

   On Windows™ operating systems:

   Any user in the *Administrators* group.

   On UNIX™ operating systems:

   root

2. Stop the WebSphere Application Server Liberty either by using the conman stopappserver command (see Starting and stopping the application server and appservman on page 431) or by running:

   On Windows™ operating systems:

   <*TWA_home*>\server_*wauser*\appservertools\stopAppServer.bat

   On UNIX™ operating systems:

   <*TWA_home*>/server_*wauser*/appservertools/stopAppServer.sh

3. Open the following file:

   On Windows™ operating systems:

   <*TWA_home*>\server_*wauser*\usr\servers\engineServer\configDropins\overrides\jvm.options

   On UNIX™ operating systems:

   <*TWA_DATA_DIR*>/server_*wauser*/usr/servers/engineServer/configDropins/overrides/jvm.options

4. Edit it as follows:

   ```
   -Xms4096m
   -Xmx4096m
   -Xgcpolicy:gencon
   #nursery mem size
   -Xmn1024m
   ```

   Note: In case of high workload (more than 200000 jobs/day) use 6144 as heap size and 1536 as nursery mem size. The above suggested settings must be applied when the RAM configuration value twice the value of the heap size.

5. Save the file `jvm.option`
6. Start the WebSphere Application Server Liberty, either by using the conman startappserver command (see Starting and stopping the application server and appservman on page 431) or by running

Windows operating systems:

<*TWA_home*>\server_*wauser*\appservertools\stopAppServer.bat

UNIX operating systems:

<*TWA_home*>/server_*wauser*/appservertools/stopAppServer.sh

Increasing maximum DB2® log capacity

The HCL Workload Automation DB2® database uses a transaction log the maximum size of which is fundamentally important for the successful running of JnextPlan on very large databases.

The default log consists of 40 primary log files, which are always present, and 20 secondary log files, created on demand. Each file is about 4 MB in size, so the maximum log capacity using all of the "secondary" log files as well as the primary files is (40 + 20) x 4 MB = 240 MB.

The log space used by JnextPlan is dependent on the size of the preproduction plan. Approximately every 1000 job stream instances generate transactions that occupy 1 MB of space in the log file. Thus, the log files by default have a maximum theoretical capacity of 240 000 job stream instances. However, in practice, you should allow for at least 25% more space than this algorithm indicates, so the capacity of the default log files is around 180 000 job stream instances.

If JnextPlan has neared or exceeded that level, you must make more log space available to DB2®.

In addition to performing the above calculation, you can also determine the log space actually used by a specific instance of JnextPlan and base your log size requirement on that figure.

Determining actual DB2® log file usage

The following is the procedure to verify how much space was used by a successful instance of the JnextPlan command:

1. After JnextPlan has run, log on to the computer where the HCL Workload Automation DB2® server is installed, as the DB2® instance owner (UNIX™) or DB2® Administrator (Windows™).
2. Open a DB2® command line window or shell, as follows:

   UNIX™

   Follow these steps:

   a. Issue the command su - db2inst1, or change to the subdirectory `sqllib` of the home directory of the owner of the DB2® instance (by default *db2inst1*)
   b. Launch the command . ./db2profile

   Windows™

   Select from the **Start** menu, **Programs** → **IBM DB2** → **Command Line Tools** → **Command Window**
3. Run the following command:

```
db2 "get snapshot for database on TWS" > snapdb.txt
```

   where "TWS" must be changed to the actual database name if different

4. Open the `snapdb.txt` file and look for a section like this:

```
         Log space available to the database (Bytes)= 244315359
         Log space used by the database (Bytes)     = 484641
         Maximum secondary log space used (Bytes)   = 0
         Maximum total log space used (Bytes)       = 581636
         Secondary logs allocated currently         = 0
```

   The value shown in "Maximum total log space used" is the actual space used for the DB2® logs. This space should be allocated to DB2® using primary log files only: therefore, you should change the number of primary log files and their size as necessary to meet this requirement as a minimum.

   In addition, you are recommended to allocate a secondary log space to DB2®. A good choice for the secondary log files is half the number allocated for the primary files.

The snapshot command described in step 3 on page 462 can be run at any time to keep track of the current usage of the DB2® log space, without a noticeable impact on performance. All metrics shown are useful to monitor the current allocation of DB2® primary and secondary logs at any time, and to determine any required changes.

Procedure for changing the maximum DB2® log capacity

Do this as follows:

1. Log on to the computer where the HCL Workload Automation DB2® server is installed, as the DB2® instance owner (UNIX™) or DB2® Administrator (Windows™).
2. Open a DB2® command line window or shell, as follows:

   UNIX™

   Follow these steps:

   a. Issue the command su - db2inst1, or change to the subdirectory `sqllib` of the home directory of the owner of the DB2® instance (by default *db2inst1*)
   b. Launch the command . ./db2profile

   Windows™

   Select from the **Start** menu, **Programs** → **IBM DB2** → **Command Line Tools** → **Command Window**

3. Run the following commands:

```
db2 update db cfg for <database_name> using LOGFILSIZ <log_file_size>
  db2 update db cfg for <database_name> using LOGPRIMARY <primary_log_files>
  db2 update db cfg for <database_name> using LOGSECOND <secondary_log_files>
```

   where:

   <database_name>

   The name of the database:

- If you are running this from the computer where the DB2® server is installed, the installed default name is *TWS*. Supply this value unless you have changed it.
- You are not recommended to run this procedure from the computer where the DB2® client is installed, but if you do so, the installed default name is *TWS_DB*. Supply this value unless you have changed it.

<log_file_size>

The log file size in 4 KB pages. The default is 1000 (hence the default log file size of 4MB). Look in the DB2® documentation for details of the implications of choosing a larger or a smaller log file size. The maximum value is 262 144 (making the maximum log file size about 1 GB).

<primary_log_files>

The number of primary log files. The default is 40. The total maximum number of log files that DB2® can handle (primary and secondary) is 256. Thus, there is a maximum limit of 256 GB for the log, or approximately 256 million Job Scheduler instances! (maximum 256 files x 1 GB maximum file size)

<secondary_log_files>

The number of secondary log files. The default is 20. If there is enough free space on the file system, these additional log files are dynamically allocated by DB2® as needed (with a small impact on the performance of JnextPlan). Because these are only created if required, it is preferable to increase the number of secondary files, rather than the primary files. Typically, you allocate 50% of the primary log file value.

In making the calculation to allocate the log files, allow at least 25% more space than you think you require, to avoid that any slight miscalculation causes JnextPlan to fail.

Example: if you have determined from the procedure described in Determining actual DB2 log file usage on page 462 that JnextPlan has a current use of 320 MB, you could calculate as follows:

a. Increase 320 MB by 25%, giving 400 MB
b. Determine if you want more log files, or bigger log files, or both, by reference to the DB2® documentation. For example, you could choose to allocate 40 files with a size of 10 MB, 80 files with a size of 5 MB, or 100 files with a size of 4 MB. For the sake of this example, assume you have chosen 80 files with a size of 5 MB, so your LOGPRIMARY value will be 80.
c. Determine the log file size in 4 KB pages to give a log file size of 5 MB - your LOGFILSIZ value will thus be 1250.
d. Determine how many secondary log files are required. If you follow the 50% guideline you will need a LOGSECOND value of 40.

4. Log on to the computer where HCL Workload Automation is installed as the following user:

UNIX™

root

Windows™

Any user in the *Administrators* group.

5. Access the directory: `<TWS_INSTALLATION_PATH>\server_<wauser>\appservertools`
6. Stop the WebSphere Application Server Liberty using the conman stopappserver command (see Starting and stopping the application server and appservman on page 431)
7. On the computer where the DB2® server is installed, stop and start DB2®, as follows:
    a. Ensure that no other applications are using this instance of DB2®, or if they are that they can be stopped.
    b. Issue the following command:

    ```
    db2stop
    ```

    c. Issue the following command:

    ```
    db2start
    ```

    Note: It is strongly recommended that you stop and start DB2®. If this is a problem for you, you must at least disconnect all applications from the DB2® instance and reconnect them. DB2® will apply the new parameters when you reconnect. If necessary, use the following command to force the disconnection of all open connections:

    ```
    db2 "force application all"
    ```

8. Start the WebSphere Application Server Liberty using the conman startappserver command (see Starting and stopping the application server and appservman on page 431)

Oracle tablespace size

Oracle (RDBMS) is divided in tablespaces which are an allocation of space where datafiles are stored.

The number of transactions that can be managed by the Oracle log files depends on the way the Oracle database is configured, thus it is important to allocate an appropriate table size to avoid issues on performance.

For example, considering a workload of 500000 jobs per day, 80GB of .dbf file size is recommended.

Multiple Dynamic Workload Console production plan reports

From the Dynamic Workload Console you can launch production plan reports. These are heavy users of CPU time, and if they are requested for the entire plan, they can also take some considerable time to produce. If several are running at once, they can have a noticeable impact on the performance of the master domain manager.

If you notice a degradation of performance, you can determine if there are any reports running by checking for the report work files, as follows;

1. Navigate to the operating system's temporary directory
2. Look for files that have the following file name template:

    ```
    TWS-sequential_number-extr
    ```

Each report currently in progress has one of these work files open. The files are removed when the report is completed.

3. Check the dates of these files, and consider only recent files (if a report fails during production at any time, its file remains in the temporary directory until the next reboot of the master domain manager or you run an operating system cleanup process that discards all files in the temporary directory).

There is no direct action to take, as you must wait until the report completes for the performance to recover.

However, if you note that large numbers of reports are being issued, it might indicate the following scenario:

1. A user issues a report request, expecting it to be available immediately
2. When the report does not appear immediately, the user things it has hung, closes and reopens the browser, and reissues the report. The closing of the browser does not stop the report production.
3. The user might repeat this action several times.

In this case, you can take action to remind the user that the production of large reports can be time-consuming, and that it always better to wait.

Dynamic Workload Console - adjusting session timeout settings

About this task

The value assigned to the session timeout settings defines after how many minutes a user is automatically logged out from the WebSphere Application Server Liberty. If you plan to perform long running operations, or to have many users connected concurrently to the Dynamic Workload Console, or expect to have low performance on the system where the Dynamic Workload Console is installed, you might want to edit these values: `httpSession invalidationTimeout="5h"` and `ltpa expiration="1440"`.

Perform these steps to change the values assigned to the timeout settings:

1. Stop WebSphere Application Server Liberty:

   UNIX™

   ```
   ./stopAppServer.sh  [-direct]
   ```

   Windows™

   ```
   stopAppServer.bat [-direct]
   ```

   For more information about stopping WebSphere Application Server Liberty, see Application server - starting and stopping on page 428.

2. Create a .xml file with this content (i.e. timeout_config.xml):

   ```
   <server description="http_timeout_config">
    <httpSession invalidationTimeout="5h" invalidateOnUnauthorizedSessionRequestException="false"/>
    <ltpa expiration="1440"/>
    </server>
   ```

3. Save the file in the following path:`<DATA_DIR>/usr/dwcServer/configDropins/overrides`
4. Start WebSphere Application Server Liberty:

   UNIX™

   ```
   ./startAppServer.sh  [-direct]
   ```

   Windows™

   ```
   startAppServer.bat [-direct]
   ```

Note: The desired time must be indicated in minutes

Dynamic Workload Console - Increasing application server heap size

Follow this procedure to increase the Java™ heap size:

1. Log on to the computer where Dynamic Workload Console is installed as the following user:

   Windows™ operating systems:

   Any user in the *Administrators* group.

   UNIX™ operating systems:

   root
2. Stop the WebSphere Application Server Liberty by running:

   Windows™ operating systems:

   DWC_home`\appservertools\stopAppServer.bat`

   UNIX™ operating systems:

   DWC_home`/appservertools/stopAppServer.sh`
3. Open the following file:

   Windows operating systems:

   DWC_DATA_dir`\usr\servers\dwcServer\configDropins\overrides\jvm.options`

   UNIX operating systems:

   DWC_DATA_dir`/usr/servers/dwcServer/configDropins/overrides/jvm.options`
4. Here is an example:

   ```
   -Xms4096m
   -Xmx4096m
   -Xgcpolicy:gencon
   #nursery mem size
   -Xmn1024m
   ```

> **Note:** In case of high workload (more than 50 concurrent users) use 6144 as heap size and 1536 as nursery mem size. The above suggested settings must be applied when the RAM configuration value twice the value of the heap size.

5. Save the file `jvm.option`
6. Start the WebSphere Application Server Liberty, by running

   Windows operating systems:

   DWC_home`\appservertools\startAppServer.bat`

   UNIX operating systems:

   DWC_home`/appservertools/startAppServer.sh`

Dynamic Workload Console graphical views

When viewing graphical views in a supported web browser, especially when there are hundreds of objects, including dependencies, it is recommended that you use either Google Chrome or Mozilla Firefox to guarantee the best possible performance. This applies to the following graphical views:

- Job Stream Graphical View (both model and plan)
- Plan View
- Preproduction Plan View

Using Google Chrome might reduce the Dynamic Workload Console performance

Due to Google Chrome cache static resources restrictions, when using the Dynamic Workload Console with default certificates, the performance could be reduced.

To avoid impact on the performance, use custom certificates or a different browser.

Chapter 11. Availability

This section describes factors that might affect the availability of HCL Workload Automation on a workstation. It covers the following topics:

Resolving user ID account on Windows® operating systems

About this task

HCL Workload Automation needs to resolve the user ID account on Windows® operating systems to verify the security information.

Windows® users can be classified as domain users or local users. Domain users are defined in the domain controller, while local users are defined in the workstations of the network.

For a domain user, HCL Workload Automation requests the primary domain controller (or any domain controller for Windows® 2000 or 2003 Active Directory), to identify an available domain controller. It then uses this domain controller identity to type out the structure for the user.

For a local user, HCL Workload Automation makes a request to the local workstation. Generally, HCL Workload Automation specifies two cases: one for the HCL Workload Automation user and one for the streamlogon user.

The following is a list of steps that HCL Workload Automation performs to authenticate Windows® users, and the APIs involved:

1. HCL Workload Automation looks up the user in the reference domain. For the domain user, the reference domain is the name of the Windows® network. For the local user, it is the name of the local workstation.

   API: `LookupAccountName`.
2. If the user is a domain user, HCL Workload Automation asks the primary domain controller for any domain controller that is available to resolve the account for the user in the reference domain.

   API: `NetGetAnyDCName` for Windows® or `DsGetDcName` for Windows® 2000 or 2003.
3. HCL Workload Automation requests the domain controller (or the local workstation if the user is local) for information about the user.

API: `NetUserGetInfo`.

Note: On Windows® 2000 and 2003, the permissions for this API are contained in the `BUILTIN\"Pre-Windows 2000 compatible access"` group.

Using a temporary directory on UNIX™

When performing HCL Workload Automation operations on UNIX™, temporary files are written to the temporary directory on the local workstation. Ensure that the <*TWS_user*> running operations has *read* and *write* access to this directory.

Chapter 12. License computation model

About this task

Use the license computation model to keep track of your license usage and maintain compliance. You can buy a trial license, which lasts 30 days before it expires, or a standard license, which calculates the number of successful jobs you run.

You can choose between the following license types:

trial

This license lasts for 30 days starting from your subscription day, irrespective of the number of jobs you run. Trial licenses grant you access to the product exclusively for the duration of the trial period.

full

This license usage is calculated based on the number of successful jobs executed. Multiple successful runs of the same job in the same job stream instance are counted only once for any calendar day (UTC timezone), this includes reruns and EVERY definitions. Unsuccessful jobs, internetwork dependencies on standard agents, and remote jobs are not calculated. The license usage information is sent to the license server you specified in the **licenseServerUrl optman** option, where it is processed. If the license server cannot be contacted, the information is stored locally and an error message is displayed.

The number of successful jobs is calculated by the **FINALPOSTREPORTS** job stream, which is a successor of the **FINAL** job stream. The **FINAL** job stream is placed in production every day and runs **JnextPlan** before the start of a new day.

The **FINALPOSTREPORTS** job stream is responsible for printing post-production reports and providing statistics. To gather statistic data, the **FINALPOSTREPORTS** job stream runs the **UpdateStats** script. By default, the **UpdateStats** script gathers a number of statistical data, which might be irrelevant for your organization. You can therefore decide whether you want to run the whole **UpdateStats** script, or just the license usage statistics. You can use either the Dynamic Workload Console or the **composer modify** command line. See following example:

```
UPDATESTATS SCRIPTNAME "$(install_dir)/UpdateStats"
        STREAMLOGON $(tws_user)
        RECOVERY CONTINUE
        FOLLOWS CHECKSYNC
```

See the section about enabling product license management in *Planning and Installation Guide* for information about the high-level steps required to set up your license server and configure HCL Workload Automation to communicate with it.

Additional resources

- For a general overview about My HCLSoftware, see My HCLSoftware - an overview.
- For detailed information about My HCLSoftware, see What is My HCLSoftware? and How to register as a Customer on HCLSoftware portals.

Per Job license model

About this task

To generate a report that summarizes your monthly per-job license usage, you can generate a license metric tag file (SLMTag). The SLM tag that is generated applies the **100 monthly jobs** pricing method where, the job count increments by 1 for every 100 successfully executed jobs you run and 1 job is counted when you run anywhere from 1 to 100 jobs. For example, if you run 340 jobs, 4 licenses are counted.

You can optionally retrieve consumption information for a subset of workstations. To obtain this information, remove the comment before the lines:

```
-- “((Actual_workstation_name_in_run ='WKS_NAME1' AND
-- ACTUAL_WKS_FOLDER_NAME_IN_RUN = '/FOL_WKS1/') OR
-- (Actual_workstation_name_in_run = 'WKS_NAME2' AND
-- ACTUAL_WKS_FOLDER_NAME_IN_RUN = '/FOL_WKS2/')) and "
```

remove the double dashes (—), and replace the '*/FOL_WKS1/*', '*WKS_NAME1*', '*/FOL_WKS2/*', '*WKS_NAME2*' strings with your folder and workstation couples.

The master domain manager centrally maintains the history of the jobs that you run in your environment. By using the **optman** global option, **statsHistory**, you can set the number of days for which you maintain the history of the jobs. To track your monthly per-job license usage, set the value of **statsHistory** to 400 (which is the default value). For more information about **statsHistory**, see Global options - detailed description on page 27.

Queries to verify the number of jobs you run every month

About this task

An SQL query is provided that accesses the job history in the database to verify the number of jobs that you run every month in your environment. The job runs calculated with this query are not grouped in groups of 100 as with the previous queries, but are instead, the total number of jobs that ran.

You can run the SQL query either from the command-line interface of your database, or by creating your custom SQL report tasks from the Dynamic Workload Console, as described in the related section in *Dynamic Workload Console User's Guide*.

- For **DB2** database type:

```
SELECT Year, Month, count(*) AS JobNbr from
(SELECT unique year(Job_run_date_time) AS Year,
month(Job_run_date_time) AS Month, day(Job_run_date_time) AS day,
JOB_STREAM_WKS_FOL_NAME, JOB_STREAM_WKS_NAME_IN_RUN,
JOB_STREAM_FOLDER_NAME, JOB_STREAM_NAME_IN_RUN,
JOB_NAME_IN_RUN
FROM MDL.JOB_HISTORY_V
WHERE Job_status='S' and
-- ((Actual_workstation_name_in_run = 'WKS_NAME1' AND
-- ACTUAL_WKS_FOLDER_NAME_IN_RUN = '/FOL_WKS1/') OR
-- (Actual_workstation_name_in_run = 'WKS_NAME2' AND
-- ACTUAL_WKS_FOLDER_NAME_IN_RUN = '/FOL_WKS2/')) and
 (Actual_WKS_FOLDER_NAME_IN_RUN, Actual_workstation_name_in_run) not in
(select FOL_PATH, WKS_NAME from MDL.WKS_WORKSTATIONS W JOIN MDL.FOL_FOLDERS
F ON W.FOL_ID=f.FOL_ID where W.WKS_AGENT_TYPE='E'))
GROUP BY Year, Month
```

- For **ORACLE** database type:

```
SELECT Year, Month, cast (count(*) AS INT) AS JobNbr from
(SELECT unique EXTRACT(year FROM Job_run_date_time) AS Year,
  EXTRACT(month FROM Job_run_date_time) AS Month,
  EXTRACT(day FROM Job_run_date_time) AS Day,
  JOB_STREAM_WKS_FOL_NAME,
  JOB_STREAM_WKS_NAME_IN_RUN,
  JOB_STREAM_FOLDER_NAME,
  JOB_STREAM_NAME_IN_RUN,
  JOB_NAME_IN_RUN
  FROM JOB_HISTORY_V
  WHERE Job_status='S' and
-- ((Actual_workstation_name_in_run = 'WKS_NAME1' AND
-- ACTUAL_WKS_FOLDER_NAME_IN_RUN = '/FOL_WKS1/') OR
-- (Actual_workstation_name_in_run = 'WKS_NAME2' AND
-- ACTUAL_WKS_FOLDER_NAME_IN_RUN = '/FOL_WKS2/')) and
 (Actual_WKS_FOLDER_NAME_IN_RUN, Actual_workstation_name_in_run) not in
  (select FOL_PATH, WKS_NAME from WKS_WORKSTATIONS W JOIN FOL_FOLDERS F ON
W.FOL_ID=f.FOL_ID where W.WKS_AGENT_TYPE='E'))
GROUP BY Year, Month
```

- For **MSSQL**, **Azure SQL**, and **Google Cloud SQL for SQL server** database types:

```
SELECT Year, Month, count(*) AS JobNbr from
(SELECT distinct year(Job_run_date_time) AS Year, month(Job_run_date_time) AS Month,
day(Job_run_date_time) AS day, Job_stream_wks_fol_name,
Job_stream_wks_name_in_run, Job_stream_folder_name,
Job_stream_name_in_run, Job_name_in_run
FROM MDL.JOB_HISTORY_V
WHERE Job_status='S' and
--((Actual_workstation_name_in_run = 'WKS_NAME1' AND
-- Actual_wks_folder_name_in_run = '/FOL_WKS1/') OR
-- (Actual_workstation_name_in_run = 'WKS_NAME2' AND
-- Actual_wks_folder_name_in_run = '/FOL_WKS2/')) AND
not exists (select 1 from MDL.WKS_WORKSTATIONS W JOIN MDL.FOL_FOLDERS F ON
W.FOL_ID=F.FOL_ID where W.WKS_AGENT_TYPE='E' AND
Actual_wks_folder_name_in_run = F.FOL_PATH AND
Actual_workstation_name_in_run = W.WKS_NAME)) r
GROUP BY Year, Month
```

Note:

- All jobs processed or managed by HCL Workload Automation are counted, but the same job counts once if repeated more than once during the same day. To meet this requirement and be considered as the same job, jobs must contain the same *jobstream_workstation_name*, *jobstream_name* and *job_name* strings and not run on a remote engine.
- The SQL queries select only jobs that run successfully. The SQL queries do not count shadow jobs, jobs that run on agent for z/OS, and rerun jobs.

Notices

This document provides information about copyright, trademarks, terms and conditions for product documentation.

This information was developed for products and services offered in the US. This material might be available from HCL in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

HCL may not offer the products, services, or features discussed in this document in other countries. Consult your local HCL representative for information on the products and services currently available in your area. Any reference to an HCL product, program, or service is not intended to state or imply that only that HCL product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any HCL intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-HCL product, program, or service.

HCL may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

HCL
330 Potrero Ave.
Sunnyvale, CA 94085
USA
Attention: Office of the General Counsel

For license inquiries regarding double-byte character set (DBCS) information, contact the HCL Intellectual Property Department in your country or send inquiries, in writing, to:

HCL
330 Potrero Ave.
Sunnyvale, CA 94085
USA
Attention: Office of the General Counsel

HCL TECHNOLOGIES LTD. PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. HCL may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-HCL websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this HCL product and use of those websites is at your own risk.

HCL may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

HCL
330 Potrero Ave.
Sunnyvale, CA 94085
USA
Attention: Office of the General Counsel

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by HCL under terms of the HCL Customer Agreement, HCL International Program License Agreement or any equivalent agreement between us.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

Information concerning non-HCL products was obtained from the suppliers of those products, their published announcements or other publicly available sources. HCL has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-HCL products. Questions on the capabilities of non-HCL products should be addressed to the suppliers of those products.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to HCL, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. HCL, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. HCL shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

HCL®, and other HCL graphics, logos, and service names including "hcltech.com" are trademarks of HCL. Except as specifically permitted herein, these Trademarks may not be used without the prior written permission from HCL. All other trademarks not owned by HCL that appear on this website are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by HCL.

Adobe™, the Adobe™ logo, PostScript™, and the PostScript™ logo are either registered trademarks or trademarks of Adobe™ Systems Incorporated in the United States, and/or other countries.

IT Infrastructure Library™ is a Registered Trade Mark of AXELOS Limited.

Linear Tape-Open™, LTO™, the LTO™ Logo, Ultrium™, and the Ultrium™ logo are trademarks of HP, IBM® Corp. and Quantum in the U.S. and other countries.

Intel™, Intel™ logo, Intel Inside™, Intel Inside™ logo, Intel Centrino™, Intel Centrino™ logo, Celeron™, Intel Xeon™, Intel SpeedStep™, Itanium™, and Pentium™ are trademarks or registered trademarks of Intel™ Corporation or its subsidiaries in the United States and other countries.

Linux™ is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft™, Windows™, Windows NT™, and the Windows™ logo are trademarks of Microsoft™ Corporation in the United States, other countries, or both.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Cell Broadband Engine™ is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

ITIL™ is a Registered Trade Mark of AXELOS Limited.

UNIX™ is a registered trademark of The Open Group in the United States and other countries.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the HCL website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of HCL.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of HCL.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

HCL reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by HCL, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

HCL MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Index