

IBM OMEGAMON and Tivoli Management
Services on z/OS

IBM Z Monitoring Configuration Manager

1.2

SC28-3147



Contents

Products supported by Configuration Manager	8
Introduction to Configuration Manager	10
Comparison with PARMGEN	11
You must decide whether to use Monitoring Configuration Manager or PARMGEN	11
Overview	11
Details	12
Preparing to use Configuration Manager	15
Location of SMP/E installation target libraries	15
Naming your runtime environment	15
Access privileges for TSO user ID	15
Authorization for certain Configuration Manager functions	15
Issues to be aware of before using Configuration Manager	16
Defining the OMEGAMON® subsystem to z/OS®	17
Creating your first, minimal runtime environment	19
Creating or updating a runtime environment	26
Sharing runtime members with an SMP/E target installation library or creating a full, stand-alone set of runtime members	27
Completing the parameters for discovered Db2® subsystems	28
Converting a hub monitoring server to a remote monitoring server	28
Defining multiple runtime environments in an RTEDEF library	30
Batch interface	34
JCL	34
RTE_NAME and RTE_PLIB_HILEV parameters versus the values in KCIVARS	35
Contents of the KCIVARS input data set	35
Actions	35
Action options	36
CREATE	37
DISCOVER	40
GENERATE	47
DELETE	57
MIGRATE	60
PACKAGE	64
DEPLOY	67
KCIOMEGA workflows	73
Action job output	74
KCIPRINT sysout data set	75
KCITRACE sysout data set	75
Renamed SYSPRINT sysout data sets	75
Parameter validation report (\$VALRPT)	76
Products installed report (REPORT)	76
Parameters	78
Parameters that determine what products are configured	78
Parameters in the initial runtime environment configuration profile	79
GBL_DSN_CICS_CTG_DLL	79
GBL_DSN_CSF_SCSFMODE0	79
GBL_DSN_DB2_DSNEXT	80
GBL_DSN_DB2_LOADLIB_Vn	80
GBL_DSN_DB2_RUNLIB_Vn	81
GBL_DSN_IMS_RESLIB	81
GBL_DSN_IMS_SCEXLINK	81
GBL_DSN_IMS_SFUNLINK	82
GBL_DSN_WMQ_SCSQANLE	82
GBL_DSN_WMQ_SCSQAUTH	82
GBL_DSN_NETVIEW_CNMLINK	82
GBL_HFS_JAVA_DIRn	83
GBL_TARGET_HILEV	83
GBL_USS_TKANJAR_PATH	83
GBL_USS_TKAYHFS_PATH	84
RTE_COMM_PROTOCOLn	84
RTE_NAME	85
RTE_PLIB_HILEV	85

RTE_SECURITY_CLASS.....	86
RTE_SECURITY_FOLD_PASSWORD_FLAG.....	86
RTE_SECURITY_USER_LOGON.....	87
RTE_STC_PREFIX.....	87
RTE_TCP_HOST.....	88
RTE_TCP_PORT_NUM.....	88
RTE_TEMS_NAME_NODEID.....	89
RTE_TEMS_TRANSPORT_MODE.....	89
RTE_USS_RTEDIR.....	90
RTE_VTAM_APPLID_PREFIX.....	90
RTE_X_SECURITY_EXIT_LIB.....	90
Parameters with significant default values.....	91
RTE_TYPE.....	91
Parameters with different default values than PARMGEN.....	91
Parameters introduced by Monitoring Configuration Manager.....	93
Global (GBL) parameter introduced by Configuration Manager.....	93
Runtime environment (RTE) parameters introduced by Configuration Manager.....	94
Configuration Manager (KFJ) parameters.....	99
Target copy (TRG) parameters.....	116
Sparse parameter tables: The first row sets the default values for subsequent rows.....	118
Example: Db2® profile configuration.....	118
Example: Db2® subsystem configuration.....	119
Runtime environment definition (RTEDEF) library.....	120
Runtime environment definition library members.....	120
Concatenation order of runtime environment definition library members.....	122
Initial runtime environment library members.....	123
Characteristics of the runtime environment defined by the initial members.....	123
The initial members include only a few product-specific members.....	123
Special considerations for SYSPLEX rollout.....	124
Using an SMP/E target copy.....	124
RTEDEF(<i>rte_name</i>).....	124
RTEDEF(KDS\$PARM).....	125
RTEDEF(KGW\$PARM).....	125
RTEDEF(GBL\$PARM).....	126
RTEDEF(PCK\$PARM).....	126
RTEDEF(<i>trg_copy_name</i>).....	128
Runtime members.....	130
Runtime member locations.....	130
Parameters that specify runtime member locations.....	130
Communication between monitoring components.....	132
Typical topology.....	132
Required parameters.....	133
Other parameters.....	134
Communication between a remote monitoring server and the hub monitoring server.....	136
Communication between monitoring agents and a remote monitoring server.....	136
Choice of communication protocols.....	136
Variables in parameter values.....	137
Variables versus LPAR-specific RTEDEF members.....	137
Using variables.....	137
RTEDEF members that define variables.....	137
Unique variable names.....	138
Variables migrated from PARMGEN.....	138
Setting up security exits in your runtime environment.....	139
Setting up security exits using CREATE	139
Setting up security exits using MIGRATE	139
Rebuild and relink security exits using GENERATE	139
Using override embed members in Configuration Manager.....	140
Example.....	140
Enable override embed members when creating an RTE.....	140
Enable override embed members for an existing RTE.....	142
Update override embed members for an existing RTE after maintenance.....	143
Deploying remote runtime environments.....	145
Remote deployment scenario.....	145
Remote deployment example using local libraries.....	147
Overview.....	147
Diagram.....	148
1. CREATE or MIGRATE actions.....	148
2. DISCOVER action.....	149

3. Manual steps to transfer and merge your DISCOVER results	150
4. Manual steps to customize your runtime environment before generating runtime members	150
5. GENERATE action	151
6. PACKAGE action	152
7. Manual steps to transfer your package data sets to the remote system	153
8. DEPLOY on target system	153
9. Manual steps to adjust or copy your started task procedures	153
Parameters that cannot be customized for remote deployment	154
Using SMP/E target library copies	156
Define SMP/E target copy settings.....	156
Copy SMP/E target libraries.....	157
Create a target copy for an existing runtime environment	159
Maintain SMP/E target library copies.....	160
How-tos using Configuration Manager	161
How to: Migrate to Configuration Manager from PARMGEN.....	161
How to: Add a new agent or product to a runtime environment	161
How to: Add OMEGAMON® Data Provider to a runtime environment	162
How to: Use a shared procedure for multiple monitoring servers	164
How to: Create a high-availability hub monitoring server.....	165
How to: Configure passphrase and MFA support in the OMEGAMON 3270 Classic interface	166
Troubleshooting	171
Navigating Configuration Manager action output	171
Collecting diagnostic data using PDCOLLECT	171
Using PDCOLLECT (with OA68463)	172
Using PDCOLLECT (without OA68463)	177
Messages	179
Message format	179
KFJ messages.....	179
KFU messages	198
Index	221

Note

Before using this information and the product it supports, read the "Notices" topic at the end of this information.

Edition notice

Last updated: 2026-03-11

This edition applies to Version 1 Release 2 of IBM Z Monitoring Configuration Manager and to all subsequent releases and modifications until otherwise indicated in new editions.

About this information

IBM Z® Monitoring Configuration Manager (also referred to as Monitoring Configuration Manager) is a tool that configures an OMEGAMON runtime environment from a set of parameters that you specify. This process is easier and faster than using the legacy PARMGEN, with its many parameters, for configuration.

These topics provide instructions for using Monitoring Configuration Manager to perform the following tasks.

- Develop batch jobs to run the Create, Discover, Migrate, Generate, and Delete actions. See [“Batch interface” on page 34](#) for details.
- Select which parameters you want to use with Monitoring Configuration Manager. See [“Parameters” on page 78](#) for a description of valid parameters.
- Specify a library to contain the runtime environment definitions (RTEDEF). See [“Runtime environment definition \(RTEDEF\) library” on page 120](#) for more information.
- View troubleshooting material and error messages you might see when using Monitoring Configuration Manager. See [“Troubleshooting” on page 171](#) and [“Messages” on page 179](#) for more information.

Products supported by Configuration Manager

These products and components can be configured using Monitoring Configuration Manager, regardless of the product being part of a suite or pack offering or purchased as a stand-alone point product.

Each product or component is listed with its corresponding code.

<i>Table 1: Products supported by Configuration Manager</i>		
Product or component	Version	Code
Tivoli Enterprise Monitoring Server	6.3	DS
OMEGAMON® Enhanced 3270 User Interface	7.5	OB
IBM® OMEGAMON® Dashboard Edition on z/OS	5.5	WO
IBM Z® OMEGAMON® Integration Monitor	5.6	WO
IBM Z® OMEGAMON Data Provider	1.1	AY
IBM OMEGAMON® for CICS® on z/OS	5.5	C5
IBM Z OMEGAMON for CICS	5.6	CICS TS: C5, CICS TG: GW
IBM Z OMEGAMON AI for CICS	6.1	CICS TS: C5, CICS TG: GW
IBM Tivoli OMEGAMON® XE for Db2 Performance Expert on z/OS	5.4	D5
IBM Tivoli® OMEGAMON® XE for Db2® Performance Monitor on z/OS	5.4	D5
IBM OMEGAMON® for Db2 Performance Expert on z/OS	5.5	D5
IBM Z® OMEGAMON® AI for Db2	6.1	D5
IBM OMEGAMON® for IMS on z/OS	5.5	I5
IBM Z® OMEGAMON® for JVM	5.5	JJ
IBM Z® OMEGAMON® Runtime Edition for JVM	5.5	JJ
IBM Z® OMEGAMON® AI for JVM	6.1	JJ
IBM OMEGAMON® for Messaging on z/OS	7.5	MQ: MQ Integration Bus: QI
IBM OMEGAMON® for Networks on z/OS	5.5	N3
IBM Z® OMEGAMON® Network Monitor	5.6	N3
IBM Z® OMEGAMON® AI for Networks	6.1	N3
IBM OMEGAMON® for Storage on z/OS	5.5	S3
IBM Z OMEGAMON® AI for Storage	6.1	S3
IBM OMEGAMON® for z/OS	5.5	M5
IBM Z® OMEGAMON® Monitor for z/OS	5.6	M5
IBM Z® OMEGAMON® AI for z/OS®	6.1	M5
IBM Z® NetView Enterprise Management Agent	6.3	NA
IBM Z® NetView Enterprise Management Agent	6.4	NA
IBM Z® NetView Enterprise Management Agent	6.5	NA
IBM Tivoli Advanced Allocation Management for z/OS	3.3	RJ
IBM Tivoli Advanced Audit for DFSMSshm	2.6	RG
IBM Tivoli Advanced Backup and Recovery for z/OS	2.4	RV
IBM Tivoli Advanced Catalog Management for z/OS	2.6	RN
IBM Tivoli Advanced Reporting and Management for DFSMSshm	2.6	RH
IBM Tivoli Automated Tape Allocation Manager for z/OS	3.3	RK

Product or component	Version	Code
IBM Tivoli® Composite Application Manager (ITCAM) for Application Diagnostics Agent	7.1.0	YN
IBM Tivoli Composite Application Manager (ITCAM) for Application Diagnostics Agent	7.1.1	YN

Note: You can use Configuration Manager to produce a list of installed products at your site. For more information, see [“Products installed report \(REPORT\)”](#) on page 76.

For links to the documentation for these products, see [Where to find information](#).

Introduction to Configuration Manager

IBM Z® Monitoring Configuration Manager, also known as Monitoring Configuration Manager or Configuration Manager, is a tool that configures an OMEGAMON runtime environment from a set of parameters that you specify.

A runtime environment (RTE) consists of the started tasks and related members, including MVS™ data sets and z/OS® UNIX® System Services files, that are required to monitor subsystems on a z/OS® LPAR. These started tasks and related members are collectively known as *runtime members*.

To generate the runtime members for a runtime environment, you configure a set of parameters, and then you run a single Monitoring Configuration Manager job. Parameters are name-value pairs stored as plain text. The following figure illustrates this basic concept:

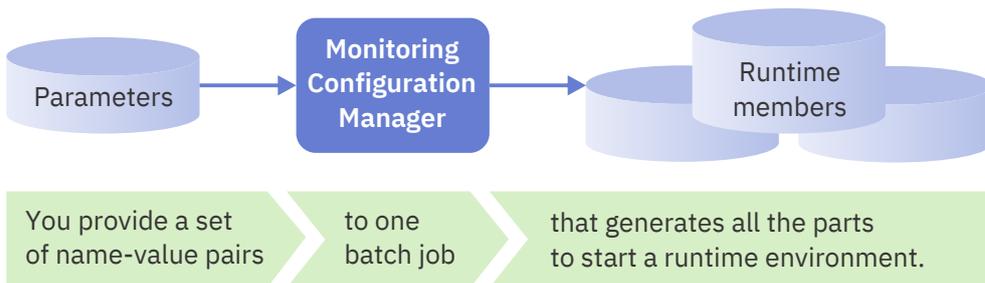


Figure 1: The basic concept: parameters in, one job, runtime members out

Generating runtime members from parameters is the main purpose of Monitoring Configuration Manager. This is known as the **GENERATE** action.

Monitoring Configuration Manager can also perform other actions:

- The **CREATE** action creates an initial set of parameters for a runtime environment.
- The **DISCOVER** action discovers subsystems on an LPAR, and then creates corresponding parameters to configure a runtime environment to monitor those subsystems.
- The **DELETE** action deletes the runtime members for a runtime environment.

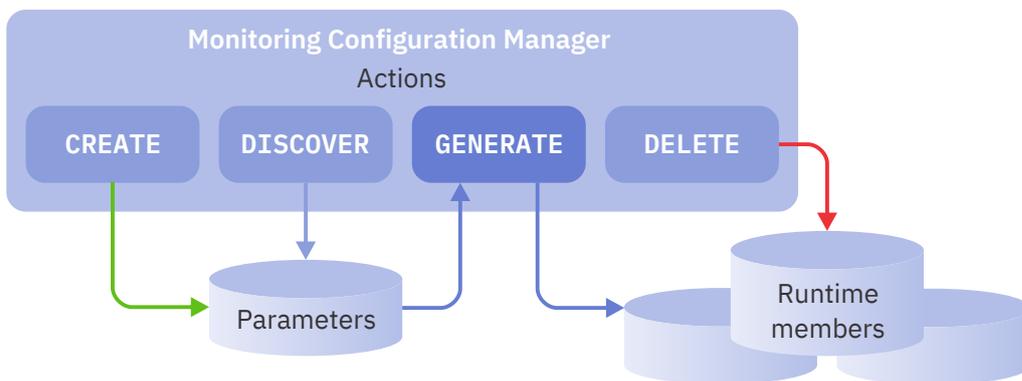


Figure 2: Actions

In addition to the actions shown in the diagram:

- The **MIGRATE** action imports existing PARMGEN RTE configuration settings from a specific WCONFIG member.
- The **PACKAGE** and **DEPLOY** actions support a convenient rollout from a single LPAR to other runtime environments.

Comparison with PARMGEN

IBM Z® Monitoring Configuration Manager evolved from PARMGEN. If you understand PARMGEN, then a comparison can help you to quickly understand Monitoring Configuration Manager.

You can use either Monitoring Configuration Manager or PARMGEN to configure runtime environments for the supported OMEGAMON agents. Monitoring Configuration Manager and PARMGEN use mainly the same parameters. However, Monitoring Configuration Manager radically simplifies the process of configuring runtime environments from those parameters.

You must decide whether to use Monitoring Configuration Manager or PARMGEN

You can use Monitoring Configuration Manager to generate runtime members for some runtime environments, PARMGEN for others, and run them in the same monitoring topology, communicating with the same hub monitoring server. In that sense, you can use Monitoring Configuration Manager and PARMGEN alongside each other.

However, you cannot use Monitoring Configuration Manager and PARMGEN interchangeably to generate runtime members for a runtime environment from the same set of parameters. For each runtime environment, you must decide whether to use Monitoring Configuration Manager or PARMGEN.

Note: You can move PARMGEN data to Configuration Manager using the **MIGRATE** action.

Overview

The following table provides an overview of the differences between Monitoring Configuration Manager and PARMGEN.

<i>Table 2: Monitoring Configuration Manager versus PARMGEN: overview</i>	
Monitoring Configuration Manager	PARMGEN
<u>Batch-only interface.</u>	Combination of ISPF user interface and batch. You navigate ISPF panels to select which job to submit.
The same simple, concise JCL for all actions.	Different jobs for different actions. Complex, long JCL. JCL members tailored for each runtime environment must be created, stored, and potentially recreated, depending on the situation.
To generate runtime members from parameters, you submit one job .	To generate runtime members from parameters, you use ISPF panels to submit a series of jobs . Some jobs submit other jobs. You need to check the output from each job, and then return to the ISPF panels to submit the next job in the series.
To generate runtime members, you submit the same job in all situations. Enhancements introduced by Monitoring Configuration Manager, including performance improvements and the streamlining of previously separate stages into a single job, removes the need for users to decide which stages to run in different situations.	You need to understand which job, or series of jobs, to submit in different situations. For example, you need to understand which jobs to run to update a runtime environment after applying SMP/E maintenance to your OMEGAMON agents.

Monitoring Configuration Manager	PARMGEN
<p>You only need to know about the input parameters and the output runtime members. Monitoring Configuration Manager insulates you from the underlying complexity.</p>	<p>In addition to understanding the inputs and outputs, you also need to understand the details of the process that generates runtime members. For example, you need to understand the difference between interim staging (IK*) libraries, work (WK*) libraries, and the final output runtime (RK*) libraries. You also need to understand which stages of the process, and which jobs, affect each of those libraries.</p>
<p>Sparse configuration profiles containing only the parameters you need. The initial configuration profile members contain only a few dozen parameters. This is all you need to use basic functions if you are content with default parameter values.</p>	<p>Comprehensive configuration profiles containing all parameters for all agents. You edit a configuration profile member containing hundreds of parameters interspersed with multiline comments.</p>
<p>Integrated subsystem discovery.</p>	<p>Requires IBM® Discovery Library Adapter for z/OS® (DLA).</p>
<p>Available for all agents of the IBM Z® Monitoring Suite and the IBM Z® NetView Enterprise Management Agent, as well as for IBM® OMEGAMON for z/OS version 5.5.0 or later, IBM® OMEGAMON for Networks on z/OS V5.5.0 or later, and IBM Z® OMEGAMON Integration Monitor V5.5.0 or later. In addition, support is provided for all agents that are part of the IBM Z Service Management Suite. This includes the same list of agents as above, except for Integration Monitor. Point product installations are also supported for the aforementioned agents/products.</p>	<p>Available with IBM Z® Monitoring Suite and other products.</p>

Details

The following table describes some differences in the implementation details between Monitoring Configuration Manager and PARMGEN.

For a comprehensive list of parameters that have different default values in Monitoring Configuration Manager and PARMGEN, see [“Parameters with different default values than PARMGEN” on page 91](#).

Text in *italics* represents a parameter value. For example, *rte_plib_hilev* represents the value of the **RTE_PLIB_HILEV** parameter.

Table 3: Monitoring Configuration Manager versus PARMGEN: details

Monitoring Configuration Manager	PARMGEN
<p>Stores <u>parameters</u> and <u>variables</u> in: <code>rte_plib_hilev.RTEDEF</code></p> <p>Each RTEDEF library can contain definitions for multiple runtime environments, customized to run on multiple LPARs.</p> <div style="border: 1px solid blue; padding: 10px; margin-top: 10px;"> <p>Note: The Configuration Manager GENERATE action creates an <code>rte_plib_hilev.rte_name.WCONFIG(rte_name)</code> member that is similar to the member created by PARMGEN, with one key difference: the member created by Configuration Manager contains default parameter values; it does not reflect the values in your RTEDEF library members. The WCONFIG data sets created by Configuration Manager, as a whole, should be considered a <i>black box</i>.</p> </div>	<p>Stores parameters in: <code>rte_plib_hilev.rte_name.WCONFIG</code></p> <p>Each WCONFIG library contains the definition for a single runtime environment, with limited flexibility to customize that definition to run on multiple LPARs.</p> <p>Stores variables in: <code>gbl_user_jcl</code></p>
<p>Organizes parameters into members according to their prefix and whether they apply to all LPARs or only to a specific LPAR.</p> <p>Provides a well-defined <u>member naming convention</u> so you know which parameters to store where and which parameters take precedence.</p> <p>Similarly, Monitoring Configuration Manager organizes variables into members that enable different values for each LPAR.</p> <p>You can configure a runtime environment using a combination of parameters that are common to all LPARs and parameters that apply only to a specific LPAR.</p>	<p>Mixes runtime environment (RTE_*) parameters and product-specific (Kpp_*) parameters in a single large member, <code>WCONFIG(rte_name)</code>.</p> <p>You can divide this monolithic member into smaller members, but you need to manage this yourself for each runtime environment. You need to define your own member naming convention, and then edit the <code>WCONFIG(\$SYSIN)</code> member to include the members in PARMGEN processing and define their order of precedence.</p> <p>No built-in support for LPAR-specific parameter values beyond using variables, such as <code>SYSNAME</code>, that have LPAR-specific values.</p>
<p>Enables you to define <u>LPAR-specific parameter values</u> without using <u>variables</u>.</p>	<p>Uses variables to customize configuration profiles for different LPARs.</p>
<p>Uses the parameters in the first row of a table as the defaults for subsequent rows.</p> <p>If you omit a parameter from a subsequent row, that row uses the value from the first row. This enables you to define more concise "<u>sparse</u>" <u>parameter tables</u> with less duplication of values.</p>	<p>You must specify parameter values for each row in a table of parameters.</p>
<p>By default, writes system library members to the same high-level qualifiers as other non-VSAM runtime members: <code>rte_hilev.SYS1.*</code></p> <p>where * is the system library low-level qualifier: PROCLIB, VTAMLIB, or VTAMLST</p>	<p>By default, writes system library members directly to: <code>SYS1.*</code></p>

Monitoring Configuration Manager	PARMGEN
<p>Writes concise started tasks with minimal comments.</p> <div style="border: 1px solid blue; padding: 5px; margin-top: 10px;"> <p>Tip: Monitoring Configuration Manager writes concise started tasks to: <code>rte_hilev.SYS1.PROCLIB</code> and versions with verbose comments to the same location used by PARMGEN: <code>rte_plib_hilev.rte_name.RKANSAMU</code> <code>rte_plib_hilev.rte_name.RKD2SAM</code> (for Db2®)</p> </div>	<p>Writes started tasks with verbose comments.</p>
<p>Sets the default value of the RTE_USS_DIR parameter to <code>/var/rtehome</code>.</p>	<p>Sets the default value of the RTE_USS_DIR parameter to <code>/rtehome</code>, a subdirectory of the root directory. Creating a new subdirectory of the root directory is bad practice.</p>
<p>In the z/OS® UNIX® System Services file system, by default writes only to <code>rte_uss_dir/rte_name</code></p>	<p>By default, writes to various z/OS® UNIX® directories.</p>
<p>Sets the default value of the RTE_TYPE parameter to SHARING and RTE_SHARE to SMP. By default, runtime environments refer to some runtime members, such as load modules, in the SMP/E installation target library, rather than creating a full copy of those members in the runtime environment's own runtime libraries.</p>	<p>Sets the default value of the RTE_TYPE parameter to FULL. By default, the runtime environment runtime members include a full copy of all members required from the SMP/E installation target library.</p>
<p>Provides the ability to create one or more copies of SMP/E target libraries from which you can create or update your runtime environments. Sharing with an SMP/E target in reality is sharing with an SMP/E target copy. The Configuration Manager <i>target copy</i> feature copies only the data sets needed for products that are selected for configuration into the target copy libraries.</p> <p>For RTE_SHARE, you can only specify SMP for a sharing runtime environment. Configuration Manager does not use base runtime environments (as in PARMGEN).</p>	<p>Provides the ability to use a set of static base libraries in a base runtime environment from which a sharing-with-base runtime environment obtains read-only runtime libraries.</p> <p>For RTE_SHARE, you can specify the name of base or full runtime environment from where a sharing runtime environment obtains its base library information, or you can specify SMP to share SMP/E target libraries.</p>

Preparing to use Configuration Manager

Review prerequisites and other considerations before you start using Configuration Manager.

This section provides information about the following items, which are prerequisites or should be considered before using Configuration Manager:

- You need to know where your SMP/E installation target library is located. See [Location of SMP/E installation target libraries](#).
- Consider the naming convention for your runtime environments. See [Naming your runtime environment](#).
- The TSO user ID that will be used to run Configuration Manager jobs needs some special access privileges. See [Access privileges for TSO user ID](#).
- Certain Configuration Manager functions require APF authorization or special RACF privileges. See [Authorization for certain Configuration Manager functions](#).
- Review other known issues. See [Issues to be aware of before using Configuration Manager](#).

Location of SMP/E installation target libraries

You need to know where your target libraries are installed:

- On MVS™: The high-level qualifiers of the SMP/E target libraries, such as TKANMOD.
- On z/OS® UNIX® System Services:
 - For products that require it, such as OMEGAMON for CICS and OMEGAMON for JVM, the path of the SMP/E target directory that is defined in the SMP/E installation jobs by ddname TKANJAR. The default directory path is /usr/lpp/kan/bin/IBM.
 - For OMEGAMON® Data Provider, the path of the SMP/E target directory that is defined in the SMP/E installation jobs by ddname TKAYHFS. The default directory path is /usr/lpp/omdp.

Typical best practice is to make a copy of the original SMP/E-managed locations and refer to the copies. This enables you to manage when to introduce changes in the original SMP/E-managed locations into your environment.

Naming your runtime environment

When you define your runtime environment, it is recommended to use a naming convention that will not interfere with your system libraries. Consider meaningful names that will easily distinguish and isolate your OMEGAMON runtime environments on your system.

Configuration Manager uses the values of **RTE_NAME** and **RTE_PLIB_HILEV** to set the name of the runtime environment and runtime environment definition library. It is recommended that the combined length of these parameters does not exceed 28 characters. For more information, see [“Creating your first, minimal runtime environment” on page 19](#).

Access privileges for TSO user ID

The TSO user ID that you plan to run Configuration Manager jobs (for example, your own user ID) must have the following access privileges:

- Read access to the target libraries and the z/OS® UNIX® directory defined by the TKANJAR ddname.
- Read access to the following z/OS® System Authorization Facility (SAF) resources in the FACILITY class:
BPX.FILEATTR.APF
BPX.FILEATTR.PROGCTL

You do not need z/OS® UNIX® superuser privileges to run Configuration Manager.

Authorization for certain Configuration Manager functions

Some Configuration Manager programs and actions require APF authorization or special RACF privileges, as follows:

- To use actions **DISCOVER** (for full discovery), **PACKAGE**, and **DEPLOY**, you must run program **KCIALPHA** from an APF-authorized load library. Make sure the necessary APF authorization of the TKANMOD library is made. For more information about APF authorization, see step [“7” on page 24](#) in [Creating your first, minimal runtime environment](#).
- The **PACKAGE** and **DEPLOY** actions use z/OS DFSMSdss commands to implement remote deployment. You might need authority to run **ADRSSU**, which is the program that is invoked when using DFSMSdss. For more information, see the authorization requirements in [“PACKAGE” on page 64](#).

Issues to be aware of before using Configuration Manager

When using Configuration Manager, especially the **GENERATE** action, make sure the job does not use a Batch Optimization tool. These tools are known to exhaust below-the-line storage and fail the job with ABEND878-10.

Defining the OMEGAMON® subsystem to z/OS®

Some OMEGAMON® monitoring agents depend on the OMEGAMON® subsystem. Before starting a runtime environment that contains any of these monitoring agents, you must define the OMEGAMON® subsystem to z/OS®.

Before you begin

Check whether the OMEGAMON® subsystem has already been defined to z/OS®.

For example, issue the following **DISPLAY MVS™** system command to list subsystems:

```
D SSI
```

The default name of the OMEGAMON® subsystem is CNDL.

About this task

IBM Z® Monitoring Configuration Manager does not depend on the OMEGAMON® subsystem. Defining the OMEGAMON® subsystem is not a prerequisite for using Monitoring Configuration Manager.

However, the OMEGAMON® subsystem *is* a prerequisite for starting some of the runtime environments that you create with Monitoring Configuration Manager.

The following OMEGAMON® monitoring agents depend on the OMEGAMON® subsystem:

- CICS® (optional)
- Db2®
- IMS
- Storage
- z/OS®

You must define the OMEGAMON® subsystem on each LPAR where you plan to start runtime environments that contain any of these monitoring agents.

When you create a runtime environment that contains one of these monitoring agents, the generated runtime members include a started task that starts the OMEGAMON® subsystem.

Optionally, you can configure your LPAR to start the OMEGAMON® subsystem whenever z/OS® restarts (IPL).

Procedure

1. Copy the OMEGAMON® subsystem initialization module KCNDLINT from the target library TKANMOD to a library in the linklist.
For example, SYS1.LINKLIB.
2. Refresh the library lookaside (LLA) library directory indexes.
Issue the following **MODIFY MVS™** system command:

```
F LLA,REFRESH
```

3. Dynamically define the OMEGAMON® subsystem to z/OS®.
Issue the following **SETSSI MVS™** system command:

```
SETSSI ADD,SUBNAME=CNDL,INITRTN=KCNDLINT,INITPARM='SSPROC=OMEGCN'
```

where:

- CNDL is the subsystem name.
CNDL is the default value of the runtime environment parameter **RTE_KCNSTR00_SSID**.

The values of **SUBNAME** and **RTE_KCNSTR00_SSID** must match.

- OMEGCN, the name of the procedure that initializes the subsystem, is the value of the runtime environment parameter **RTE_CANSCN_STC**.
The default value of **RTE_CANSCN_STC** consists of the value of the **RTE_STC_PREFIX** parameter followed by the suffix CN. The value of **RTE_STC_PREFIX** in the initial set of parameters created by Monitoring Configuration Manager is OMEG (an abbreviation of OMEGAMON®).

The **SSPROC** parameter of the **SETSSI** command and the runtime environment parameter **RTE_CANSCN_STC** must match.

If you do not want the subsystem address space to be started immediately, omit **INITPARM**.

4. Define the OMEGAMON® subsystem in the IEFSSNxx member of the SYS1.PARMLIB library. For example:

```
SUBSYS SUBNAME(CNDL) INITRTN(KCNDLINT) INITPARM('SSPROC=OMEGCN')
```

The **SETSSI** command in the previous step dynamically defines the subsystem so that it is available immediately. Defining the subsystem in the IEFSSNxx member defines the subsystem during z/OS® initialization (IPL), so that you do not have to reissue the **SETSSI** command after each IPL.

If you do not want the subsystem address space to be started at IPL, omit **INITPARM**.

Creating your first, minimal runtime environment

If you are a first-time user of IBM Z® Monitoring Configuration Manager, creating a minimal runtime environment is a good place to start. This example consists of a z/OS® agent, a monitoring server, and an enhanced 3270 user interface. You can logon to the enhanced 3270 user interface to view data from the z/OS® agent.

Before you begin

Read the [prerequisites](#) for using Monitoring Configuration Manager.

The [OMEGAMON® subsystem must be defined to z/OS®](#).

About this task

Here is a diagram of the minimal runtime environment created by the following procedure:

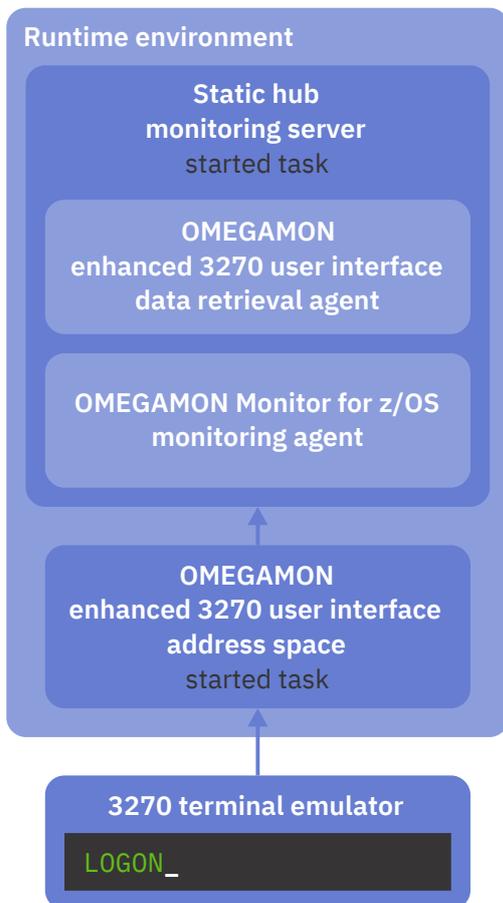


Figure 3: Overview of a minimal runtime environment

To create this minimal runtime environment, you follow the same procedure you would follow to create any runtime environment. The difference is that a minimal runtime environment involves setting fewer parameters, and involves fewer tasks to complete the configuration after running Monitoring Configuration Manager.

Procedure

1. Submit a job that performs the **CREATE** action of Monitoring Configuration Manager. The **CREATE** action creates a runtime environment definition library, `rte_plib_hilev.RTEDEF`, and populates it with an [initial set of parameters](#).

Example JCL:

Figure 4: Example JCL to perform the **CREATE** action

```

//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=<lpar>
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=<tlib_hlq>.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=<tlib_hlq>.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION CREATE
RTE_NAME <rte_name>
RTE_PLIB_HILEV <rte_plib_hilev>
/*

```

Similar JCL is supplied in the KFJJMCM member of the TKANSAM target library.

Note: Refer to this sample member, KFJJMCM, for any updates to the parameters. Any new or changed parameters will be listed in this member and can be customized according to the action that you want to run.

Edit the example job statement to match your site's standards. For example, for job name, class, and message class. Consider changing the example job name prefix, UID, to your TSO user ID.

Replace the placeholders in the example JCL with appropriate values:

<lpar>

Run Monitoring Configuration Manager actions on the LPAR where you will start the runtime environment.

For example, if your site uses JES2, insert a **SYSAFF** job parameter after the **JOB** statement to ensure that the job runs on that LPAR.

<tlib_hlq>

The high-level qualifiers of the target libraries.

<rte_name>

Runtime environment name, 1 - 8 characters.

Monitoring Configuration Manager uses this name for various purposes, including:

- MVS™ member names
- MVS™ data set name qualifiers
- z/OS® UNIX® System Services directory name, all uppercase

<rte_plib_hilev>

The high-level qualifiers of the runtime environment definition library:

rte_plib_hilev.RTEDEF

Monitoring Configuration Manager uses the values of **RTE_NAME** and **RTE_PLIB_HILEV** to set the default value of other parameters, such as **RTE_HILEV** and **RTE_VSAM_HILEV**, that are used for data set names. To avoid exceeding the z/OS 44-character limit for data set names, the combined length of **RTE_NAME** and **RTE_PLIB_HILEV** should not exceed 28 characters. For example, if **RTE_NAME** is 8 characters, then **RTE_PLIB_HILEV** should not exceed 20 characters.

Tip: After running a Monitoring Configuration Manager job, check the KCIPRINT sysout data set.

2. Edit the parameters in the *rte_plib_hilev*.RTEDEF library to configure a static hub monitoring server, a z/OS® agent, and an enhanced 3270 user interface. In the RTEDEF (*rte_name*) member, set the following **CONFIGURE_*** parameters to Y:

CONFIGURE_TEMS_KDS

Configures a monitoring server (Tivoli® Enterprise Monitoring Server, or TEMS)

CONFIGURE_ZOS_KM5

Configures a z/OS® monitoring agent

CONFIGURE_E3270UI_KOB

Configures an enhanced 3270 user interface address space

Either delete all other **CONFIGURE_*** parameters or set them to N.

The following diagram shows how these **CONFIGURE_*** parameters, and the default values of related parameters, configure the topology of the runtime environment:

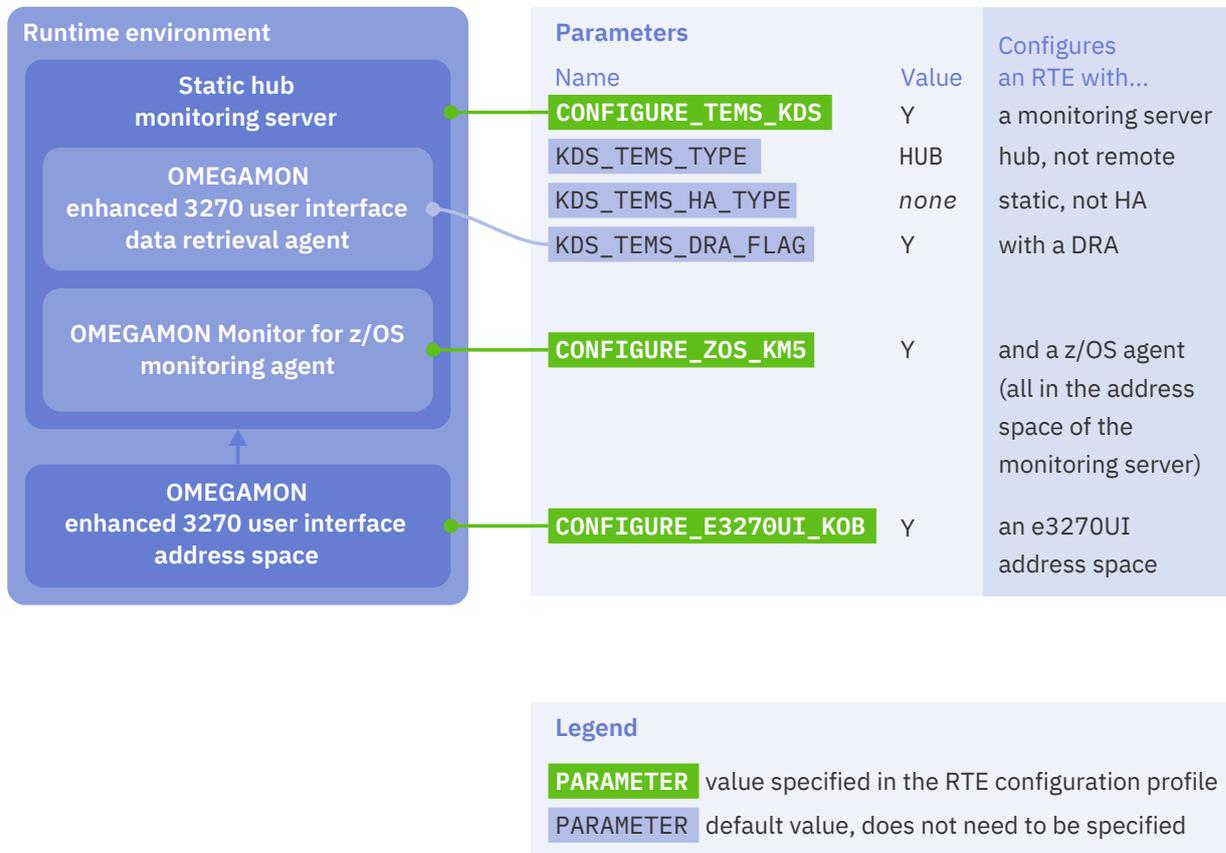


Figure 5: **CONFIGURE_*** parameters for a minimal runtime environment

Also in the RTEDEF (*rte_name*) member, set the following parameters to match your site-specific standards:

RTE_STC_PREFIX

1- to 4-character prefix of the started task names for this runtime environment. The value in the initial set of parameters is OMEG.

RTE_VTAM_APPLID_PREFIX

Prefix of the VTAM® applids in this runtime environment. For this minimal runtime environment, there is only one VTAM® application: the enhanced 3270 user interface.

Each VTAM® application in a runtime environment has a corresponding parameter for the VTAM® applid. The default values of these parameters are the value of **RTE_VTAM_APPLID_PREFIX** followed by an application-specific suffix.

In the initial set of parameters created by the **CREATE** action, the value of **RTE_VTAM_APPLID_PREFIX** is *OMxx*, where *xx* is the value of the z/OS® static system symbol **SYSCONE**. **SYSCONE** is a 1- or 2-character shorthand notation for the system (LPAR) name. This value is one example of why you need to run Monitoring Configuration Manager actions on the LPAR where you will start the runtime environment.

If you use these values, then the default VTAM® applid for the enhanced 3270 user interface is *OMxxOBAP*. For example, if the system (LPAR) name is ZOS1, then the VTAM® applid is OMS10BAP.

RTE_USS_RTEDIR

The path of the z/OS® UNIX® directory where you want Monitoring Configuration Manager to write runtime files required by the started tasks.

The TSO user ID that runs Monitoring Configuration Manager jobs must have permission to write to this directory, otherwise the **GENERATE** action will fail.

RTE_TCP_PORT_NUM

The TCP/IP port number on which the monitoring server will listen.

Tip: Later steps in this procedure describe how to activate VTAM® resources and APF-authorize libraries. If you insert the parameter **RTE_X_STC_INAPF_INCLUDE_FLAG** Y in the RTEDEF (*rte_name*) member, then the started tasks include a member that performs these steps for you.

In the RTEDEF (GBL\$PARM) member, set the following parameter:

GBL_HFS_JAVA_DIR1

The z/OS® UNIX® path of the Java™ home directory.

The following diagram shows parameters that configure identifiers and values used by the runtime environment. Notice how the **RTE** parameters determine the default values of other parameters. For example, the default value of the **KDS_TEMS_STC** parameter is the value of **RTE_STC_PREFIX** followed by the suffix DS.

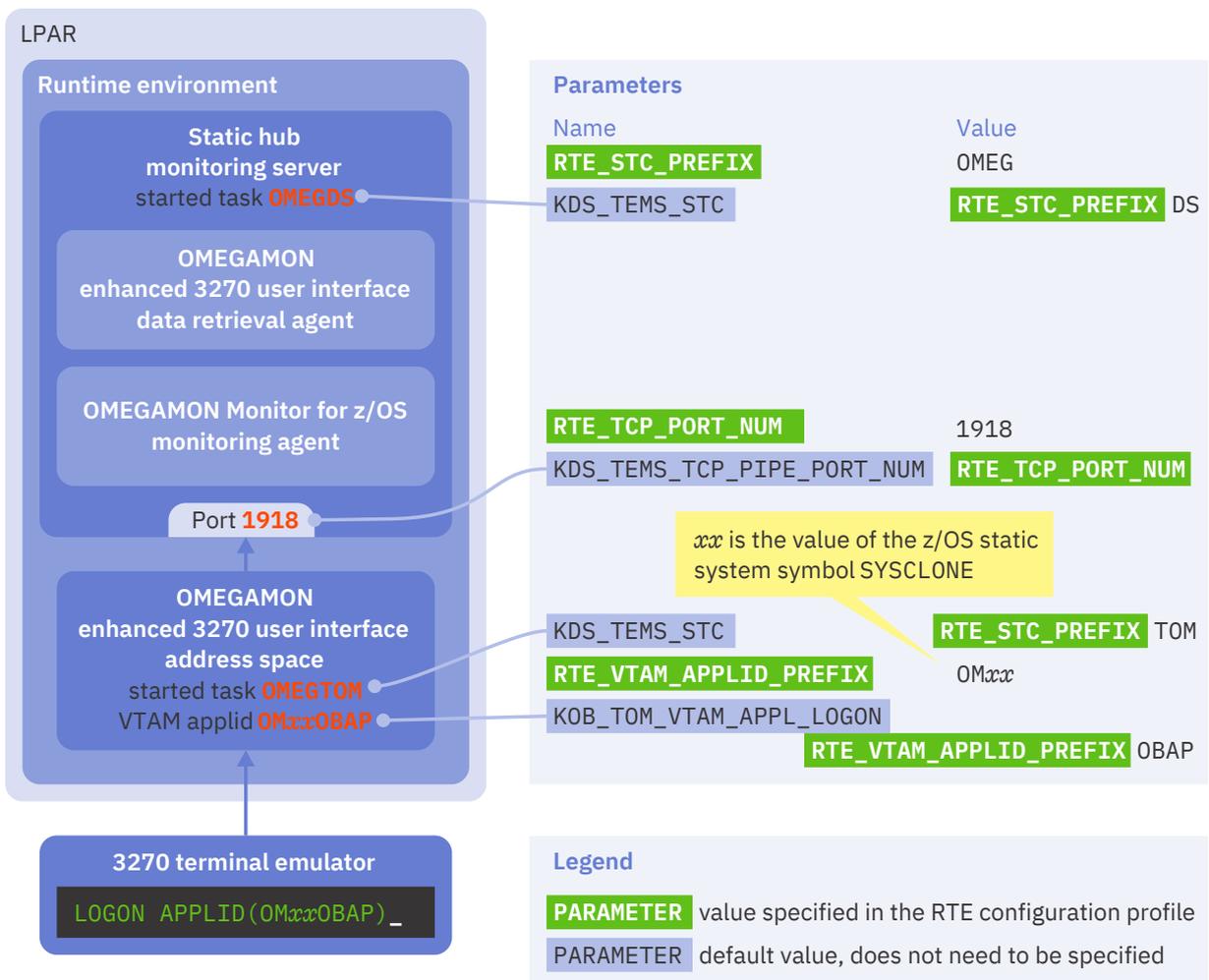


Figure 6: Parameters that specify z/OS®-related identifiers for a minimal runtime environment

- Submit a job that performs the **DISCOVER** action. The **DISCOVER** action discovers CICS® regions, Db2® subsystems, IMS control regions, MQ subsystems, TCP/IP stacks, and System Symbols, and then writes corresponding members to the RTEDEF library.

Strictly speaking, for this minimal runtime environment, you can skip this step, because this runtime environment will run only the z/OS® agent, which does not require any subsystem parameters. However,

performing the **DISCOVER** action is still a useful exercise, because it prepares the RTEDEF library for extending the runtime environment to run other agents.

Reuse the same JCL as before, with the following changes:

[1]

Optionally, change the program name in the JCL **EXEC** statement to KCIALPHA. KCIALPHA is an APF-authorized version of KCIOMEGA. APF-authorization enables the program to discover more subsystem details.

[2]

Change the action to **DISCOVER**.

Example JCL:

Figure 7: Example JCL to perform the **DISCOVER** action

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=<lpar>
//S1      EXEC PGM=KCIALPHA,REGION=0M,DYNAMNBR=256      [1]
//STEPLIB DD DISP=SHR,DSN=<tlib_hlq>.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=<tlib_hlq>.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION          DISCOVER                                [2]
RTE_NAME        <rte_name>
RTE_PLIB_HILEV  <rte_plib_hilev>
/*
```

4. Submit a job that performs the **GENERATE** action.

Reuse the same JCL as before, with the following changes:

[1]

If you changed the program name to KCIALPHA for the **DISCOVER** action, change it back to KCIOMEGA before performing the **GENERATE** action.

[2]

Change the action to **GENERATE**.

Example JCL:

Figure 8: Example JCL to perform the **GENERATE** action

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=<lpar>
//S1      EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256      [1]
//STEPLIB DD DISP=SHR,DSN=<tlib_hlq>.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=<tlib_hlq>.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION          GENERATE                                [2]
RTE_NAME        <rte_name>
RTE_PLIB_HILEV  <rte_plib_hilev>
/*
```

The **GENERATE** action generates the runtime members for the runtime environment, including the started tasks.

You have completed the steps that involve the configuration software, Monitoring Configuration Manager.

The remaining steps complete the configuration of the runtime environment *outside* of the configuration software.

These “complete the configuration” steps depend on your site-specific procedures and the requirements of the components, such as the monitoring agents, that you have chosen to configure in the runtime environment. The requirements of each component are described in the separate product documentation for each component.

Typically, at this point in the procedure for creating a runtime environment, you would need to refer to that separate documentation. However, to help make this “first runtime environment” procedure stand-alone, and because in this procedure we have selected a fixed set of specific components, the “complete the configuration” steps are presented here.

Depending on your user privileges, you might need to ask someone else to perform some or all of the following steps. For example, only z/OS® system administrators are typically allowed to write to system libraries.

5. Use your site-specific procedures to copy the runtime members for started tasks and VTAM® definitions to your system libraries.
Copy the members from the following libraries to your corresponding PROCLIB, VTAMLIB, and VTAMLST system libraries:

```
rte_hilev.SYS1.PROCLIB
rte_hilev.SYS1.VTAMLIB
rte_hilev.SYS1.VTAMLST
```

The default value of the RTE_HILEV parameter is the value of RTE_PLIB_HILEV.

If you followed the earlier tip to set the **RTE_X_STC_INAPF_INCLUDE_FLAG** parameter to Y, then you can skip the next two steps. However, you should still *read* these steps to understand the requirements of the runtime environment for VTAM® resources and APF-authorized libraries.

6. Activate the VTAM® resources defined by this runtime environment.
Issue the following **VARY ACT** MVS™ system command:

```
VARY NET,ACT,ID=rte_vtam_applid_prefixNODE,SCOPE=ALL
```

The **ID** parameter of the **VARY ACT** command must match the value of the runtime environment parameter **RTE_VTAM_GBL_MAJOR_NODE**.

The default value of **RTE_VTAM_GBL_MAJOR_NODE** is the value of **RTE_VTAM_APPLID_PREFIX** followed by the string **NODE**. If you use the **RTE_VTAM_APPLID_PREFIX** initial value of **OMxxx**, then the default value of **RTE_VTAM_GBL_MAJOR_NODE** is **OMxxxNODE**, where **xxx** is the value of the z/OS® static system symbol **SYSCLONE**. For example, if the system (LPAR) name is **ZOS1**, then specify **ID=OMS1NODE**.

7. APF-authorize libraries.
Add the following data sets to the authorized program facility (APF) list:

- The following runtime environment library:
`rte_hilev.rte_name.RKANMODU`
- The following target libraries, under the high-level qualifiers of the STEPLIB of the Monitoring Configuration Manager job:
TKANMOD
TKANMODL
TKANMODP
TKANMODR

The runtime member `rte_hilev.SYS1.PROCLIB(rte_stc_prefixAPF)` contains **VARY ACT** and **SETPROG APF** commands for this runtime environment. Different runtime environments require different VTAM® resources and APF-authorized libraries, depending on the configured products.

If you specify the parameter **RTE_X_STC_INAPF_INCLUDE_FLAG** Y in the RTEDEF (`rte_name`) member, and then perform the **GENERATE** action, some started tasks will contain an **INCLUDE** statement to include that member, so that you do not need to issue these commands separately. Whether started tasks are allowed to perform such commands depends on your local site practices.

Setting **RTE_X_STC_INAPF_INCLUDE_FLAG** can be expedient for initial testing. However, typical best practice is to activate VTAM® resources and APF-authorize libraries during system initialization rather than each time you start a task. Use the generated `SYS1.PROCLIB(rte_stc_prefixAPF)` member to identify which libraries you need to add to the APF list at system initialization.

8. Start at least the following tasks: `rte_stc_prefixCN`, `rte_stc_prefixDS`, and `rte_stc_prefixTOM`.
The user ID that you associate with these started tasks must have z/OS® UNIX® superuser privileges and access to the [runtime members](#).

rte_stc_prefixCN

OMEGAMON® subsystem.

The OMEGAMON® subsystem does not belong to a runtime environment. You only need one OMEGAMON® subsystem per LPAR.

If the OMEGAMON® subsystem has already been started, the job for this started task will fail. The JESMSG LG output data set for the failed job will contain the following messages:

CNDL018I OMEGAMON SUBSYSTEM ALREADY ACTIVE ...
CNDL002I OMEGAMON SUBSYSTEM *Vvrm* TERMINATED ...

This is not a problem: your runtime environment will use the already-active subsystem.

***rte_stc_prefix*DS**

Monitoring server.

***rte_stc_prefix*TOM**

Enhanced 3270 user interface.

What to do next

Logon to the enhanced 3270 user interface and view the monitoring data from the z/OS® agent.

Creating or updating a runtime environment

To create or update a runtime environment, you edit a set of parameters, and then you submit a job that performs the **GENERATE** action to generate runtime members from those parameters.

Before you begin

Read the [prerequisites](#) for using Monitoring Configuration Manager.

If you are updating a runtime environment after using SMP/E to apply maintenance to the target libraries, and your runtime environment refers to a copy of those libraries, then use your site-specific procedures to refresh that copy.

About this task

When creating a runtime environment, you can optionally perform the **CREATE** action to create an initial set of parameters.

When creating or updating a runtime environment, you can optionally perform the **DISCOVER** action to create or update subsystem parameters, rather than editing them yourself.

Updating a runtime environment encompasses many scenarios. For example:

- Applying maintenance, after using SMP/E to update target libraries
- Upgrading an agent to a new product release
- Adding or removing agents
- Changing parameter values; for example, to change a hub monitoring server to a remote monitoring server

The procedure for updating the runtime environment is the same for every scenario.

Procedure

1. Create a runtime environment definition. If you are updating a runtime environment, the definition already exists: skip this step.
A runtime environment definition consists of one or more plain-text members in a library. These members specify the parameters that define the runtime environment. For details, see [“Runtime environment definition \(RTEDEF\) library”](#) on page 120.

To create a runtime environment definition, submit a job that performs the **CREATE** action of Monitoring Configuration Manager.

Example JCL:

*Figure 9: Example JCL to perform the **CREATE** action*

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=<lpar>
//S1      EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=<tlib_hlq>.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=<tlib_hlq>.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION          CREATE
RTE_NAME        <rte_name>
RTE_PLIB_HILEV  <rte_plib_hilev>
/*
```

Similar JCL is supplied in the KFJJMCM member of the TKANSAM target library.

Edit the example job statement to match your site's standards. For example, for job name, class, and message class.

Replace the placeholders in the example JCL with appropriate values. For details, see [“Batch interface”](#) on page 34.

Tip: After running a Monitoring Configuration Manager job, check the KCIPRINT sysout data set.

2. Edit the parameters in the RTEDEF library to meet your requirements for the runtime environment.
3. Optionally, submit a job that performs the **DISCOVER** action.
The **DISCOVER** action discovers CICS®, Db2®, IMS, and MQ subsystems, TCP/IP stacks, and System symbols, and then writes corresponding members to the RTEDEF library.

Use the same JCL as the previous step but change the action to **DISCOVER**.

Optionally, change the program name in the JCL **EXEC** statement to KCIALPHA. KCIALPHA is an APF-authorized version of KCIOMEGA. APF-authorization enables the program to discover more subsystem details.

Check and, if necessary, edit the contents of the members created by the **DISCOVER** action.

For discovered Db2® subsystems, you need to complete some of the parameters: see “Completing the parameters for discovered Db2 subsystems” on page 28.

4. Submit a job that performs the **GENERATE** action.
Use the same JCL shown in the first step but change the action to **GENERATE**.

The **GENERATE** action generates the runtime members for the runtime environment, including the started tasks.

Note: You have completed the steps that involve the configuration software, Monitoring Configuration Manager.
The remaining steps complete the configuration of the runtime environment *outside* of the configuration software.

These “complete the configuration” steps depend on your site-specific procedures and the requirements of the components, such as the monitoring agents, that you have chosen to configure in the runtime environment. The requirements of each component are described in the separate product documentation for each component.

Depending on your user privileges, you might need to ask someone else to perform some or all of the following steps. For example, only z/OS® system administrators are typically allowed to write to system libraries.

5. Use your site-specific procedures to copy the runtime members for started tasks and VTAM® definitions to your system libraries.
Copy the members from the following libraries to your corresponding PROCLIB, VTAMLIB, and VTAMLST system libraries:

```
rte_hilev.SYS1.PROCLIB  
rte_hilev.SYS1.VTAMLIB  
rte_hilev.SYS1.VTAMLST
```

The default value of the RTE_HILEV parameter is the value of RTE_PLIB_HILEV.

6. Follow the instructions in the OMEGAMON® shared documentation to complete the configuration of the runtime environment.
7. For a newly created runtime environment: start the tasks. For an updated runtime environment: stop and then restart the tasks.

Sharing runtime members with an SMP/E target installation library or creating a full, stand-alone set of runtime members

Runtime members can be either a full stand-alone set or they can refer to some read-only members, such as load modules, in SMP/E target installation libraries.

Procedure

1. In the RTEDEF (*rte_name*) member, specify an **RTE_TYPE** value.
Valid values:

FULL

Stand-alone runtime members. Runtime members have no dependency on target libraries.

SHARING

Some runtime members refer to the target libraries.

The high-level qualifiers of the target libraries are specified by the **GBL_TARGET_HILEV** parameter.

SHARING reduces the storage requirement for each runtime environment.

If **RTE_TYPE** is SHARING, then the value of the RTE_SHARE parameter must be SMP.

If you omit **RTE_TYPE**, the default value is SHARING.

2. Submit a job that performs the **GENERATE** action.

Completing the parameters for discovered Db2® subsystems

The **DISCOVER** action discovers only some of the parameter values required to monitor Db2® subsystems. You must supply the remaining values.

Before you begin

You must have performed a **DISCOVER** action that created an RTEDEF (KD5@*lpar*) member containing a table of parameters: one row for each discovered Db2® subsystem.

About this task

You must supply values for the following parameters:

KD2_DBnn_DB2_RUNLIB

The Db2® RUNLIB library.

KD2_DBnn_DB2_PORT_NUM

The port number on which the OMEGAMON® for Db2® Collector (or “server”, default started task suffix 02) listens for requests.

The **DISCOVER** action sets a placeholder value for the port number. Typically, you will need to change this value to match your site-specific standards.

To specify the RUNLIB library for discovered Db2® subsystems, you must either ensure that the global parameters are correct or edit the parameters for each Db2® subsystem.

Procedure

1. Specify the correct port numbers.
In the RTEDEF (KD5@*lpar*) member, change the placeholder value of each **KD2_DBnn_DB2_PORT_NUM** parameter to the actual port number you want to use.
2. Specify Db2® RUNLIB libraries.
Select one of the following choices:
 - To use global parameter values for Db2® LOADLIB and RUNLIB libraries: in the RTEDEF (GBL\$PARM) member or LPAR-specific RTEDEF (GBL@*lpar*) member, define **GBL_DSN_DB2_LOADLIB_Vn** and **GBL_DSN_DB2_RUNLIB_Vn** parameters for the Db2® versions that your site uses.
 - To set LOADLIB and RUNLIB libraries for each Db2® subsystem: in the RTEDEF (KD5@*lpar*) member, specify a value for **KD2_DBnn_DB2_RUNLIB**.

Converting a hub monitoring server to a remote monitoring server

Initially, you might configure a new runtime environment to be stand-alone, with its own hub monitoring server. Later, you can integrate that runtime environment with the rest of your monitoring topology by converting its hub monitoring server to a remote monitoring server that communicates with a central hub.

Before you begin

The following procedure assumes that you have already created the following two runtime environments:

- A runtime environment with a hub monitoring server that you want to convert to a remote monitoring server.
- A runtime environment with the central hub monitoring server that you want the new remote monitoring server to communicate with. We'll call this the **central hub** runtime environment.

About this task

Converting a hub monitoring server to a remote monitoring server involves changing the **KDS_TEMS_TYPE** parameter value from HUB to REMOTE, and setting some other **KDS_*** parameters to refer to the central hub.

The following diagram shows a stand-alone runtime environment with a hub:



Figure 10: Before: A stand-alone runtime environment with a hub monitoring server

The following diagram shows the runtime environment after you have converted its hub to a remote monitoring server that communicates with a central hub:

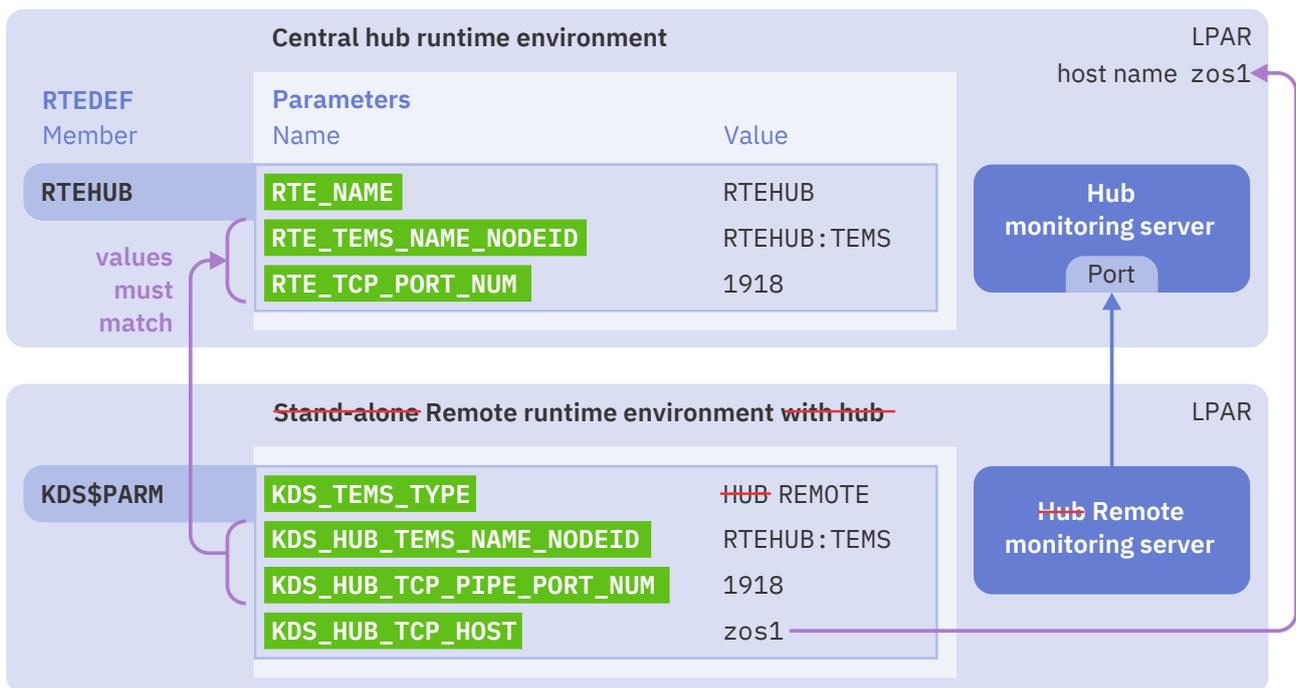


Figure 11: After: A runtime environment with a remote monitoring server

Procedure

1. Edit the RTEDEF (KDS\$PARM) or RTEDEF (KDS\$lpar) member for the runtime environment with the hub that you want to convert to a remote monitoring server. Set the following parameters:

KDS_TEMS_TYPE

Change from HUB to REMOTE.

KDS_HUB_TEMS_NAME_NODEID

Set to the value of **RTE_TEMS_NAME_NODEID** in the central hub runtime environment.

KDS_HUB_TCP_PIPE_PORT_NUM

Set to the value of RTE_TCP_PORT_NUM in the central hub runtime environment.

KDS_HUB_TCP_HOST

Set to the host name of the LPAR for the central hub runtime environment.

2. Submit a job that performs the **GENERATE** action for the runtime environment whose parameters you have just edited.

Defining multiple runtime environments in an RTEDEF library

You can define one runtime environment per RTEDEF library or, as described here, you can define multiple runtime environments in a single RTEDEF library.

About this task

For each runtime environment that you want to define, you create a corresponding RTEDEF (*rte_name*) member. To cater for LPAR-specific parameter differences between runtime environments, you create LPAR-specific RTEDEF members.

The simple example presented here defines two runtime environments on different LPARs:

- A runtime environment with a hub monitoring server and an enhanced 3270 user interface, but no monitoring agents.
- A runtime environment with a remote monitoring server and a z/OS® monitoring agent.

The following diagram shows the significant RTEDEF members and their parameters for this example:

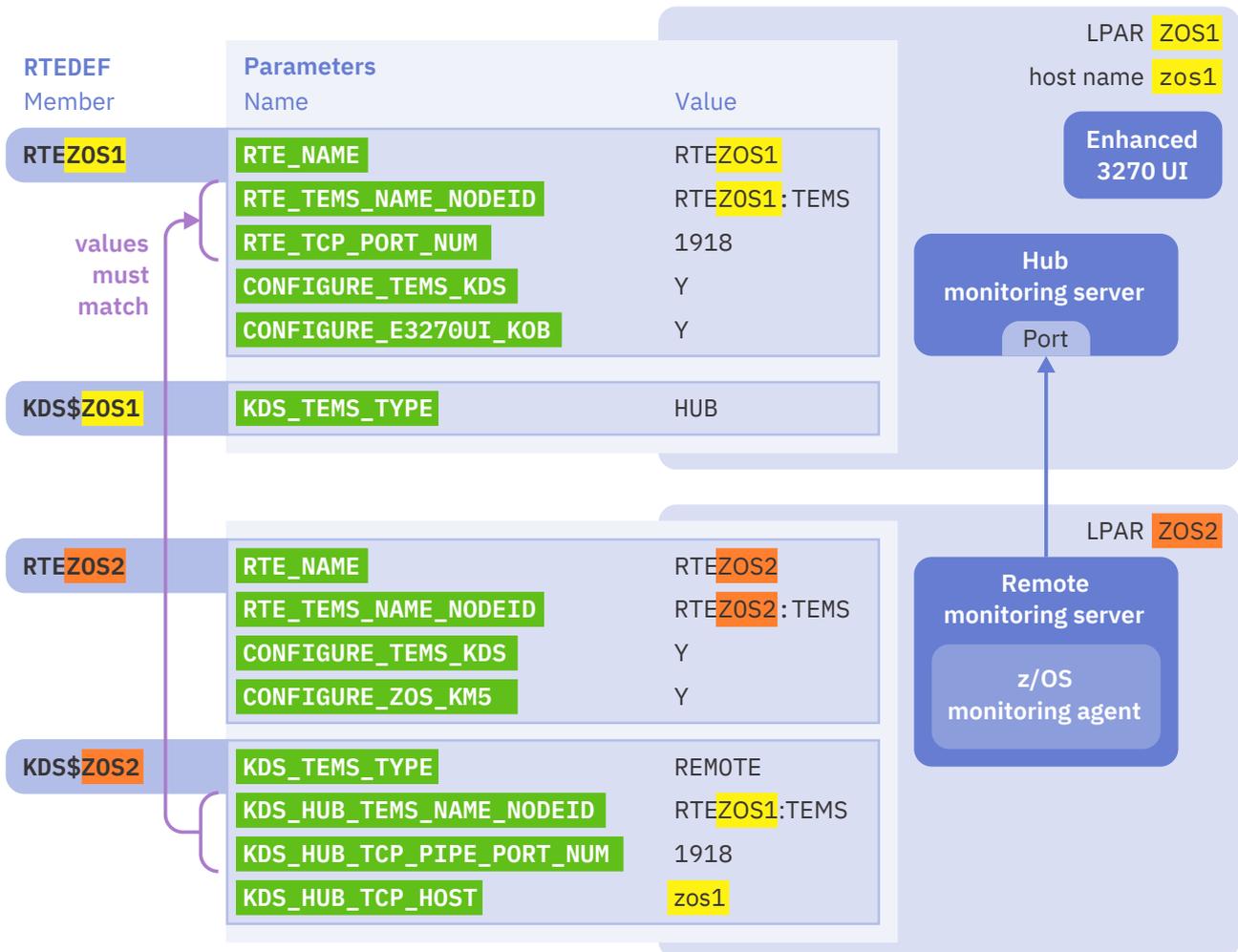


Figure 12: Defining two runtime environments in a single RTEDEF library

In the following procedure, replace the example names with names appropriate for your site:

- Replace the host name `zos1` with the host name of the LPAR where you will run the hub monitoring server.
- Replace the LPAR names `ZOS1` and `ZOS2` with the names of LPARs at your site.
- Use RTE names that match your site naming conventions.

Procedure

1. Use the **CREATE** action to create an initial set of parameters in a new RTEDEF library.

Example JCL:

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1 [1]
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=<tlib_hlq>.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=<tlib_hlq>.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION CREATE
RTE_NAME RTEZOS1 [2]
RTE_PLIB_HILEV TSOUID.MONSUITE
/*
```

[1]

Run this job on the LPAR where you will run the hub monitoring server, `ZOS1`.

[2]

Specify the **RTE_NAME** that you want to use for the runtime environment with the hub monitoring server. For example, `RTEZOS1`: the prefix `RTE`, followed by the LPAR name. You are free to use your own naming convention: for example, there is no requirement for the value to begin with `RTE` or to end with the LPAR name.

2. Edit the RTEDEF (`RTEZOS1`) member to configure a monitoring server and an enhanced 3270 user interface.

Set the following parameters to Y:

CONFIGURE_TEMS_KDS to configure a monitoring server

CONFIGURE_E3270UI_KOB to configure an enhanced 3270 user interface

Either delete all other **CONFIGURE_*** parameters or set them to N.

Review the following parameter values and, if necessary, change them to match your site requirements:

RTE_TEMS_NAME_NODEID

RTE_TCP_PORT_NUM

3. Create an RTEDEF (`KDS$ZOS1`) member.

Set a single parameter in this member:

KDS_TEMS_TYPE HUB

Strictly speaking, this member is unnecessary, because `HUB` is the default value of **KDS_TEMS_TYPE**.

However, creating this member serves as a reminder that the runtime environment on this LPAR contains a *hub* monitoring server. You will also need this member if you decide later to further configure the hub: for example, to make it a high-availability hub (**KDS_TEMS_HA_TYPE** HA).

You have completed the definition of the runtime environment for the hub.

4. Run another job that performs a **CREATE** action, this time for the LPAR where you will run the remote monitoring server.

Example JCL:

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS2 [1]
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=<tlib_hlq>.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=<tlib_hlq>.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION CREATE
RTE_NAME RTEZOS2 [2]
RTE_PLIB_HILEV TSOUID.MONSUITE
/*
```

[1]

Run this job on the LPAR where you will run the remote monitoring server, ZOS2.

[2]

Specify the **RTE_NAME** that you want to use for the runtime environment with the remote monitoring server. For example, RTEZOS2.

5. Edit the new RTEDEF (RTEZOS2) member to configure a monitoring server and a z/OS® monitoring agent. Set the following parameters to Y:

CONFIGURE_TEMS_KDS to configure a monitoring server
CONFIGURE_ZOS_KM5 to configure a z/OS® monitoring agent
Either delete other **CONFIGURE_*** parameters or set them to N.

6. Create an RTEDEF (KDS\$ZOS2) member.
Set the following parameters in the member:

KDS_TEMS_TYPE

Set to REMOTE.

KDS_HUB_TEMS_NAME_NODEID

Set to the value of **RTE_TEMS_NAME_NODEID** in the hub runtime environment.

KDS_HUB_TCP_PIPE_PORT_NUM

Set to the value of **RTE_TCP_PORT_NUM** in the hub runtime environment.

KDS_HUB_TCP_HOST

Set to the host name of the LPAR for the hub runtime environment.

7. Optionally, delete the RTEDEF (KDS\$PARM) member.
The KDS\$PARM member created by the **CREATE** action contains only one parameter, **KDS_TEMS_TYPE**. The value of **KDS_TEMS_TYPE** in the LPAR-specific members takes precedence over the value in KDS\$PARM. So the KDS\$PARM member is, effectively, redundant.

You have completed the definition of the runtime environment for the remote monitoring server.

8. Use the **GENERATE** action to create runtime members for the hub runtime environment.
Example JCL:

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID  
/*JOBPARM SYSAFF=ZOS1 [1]  
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256  
//STEPLIB DD DISP=SHR,DSN=<plib_hlq>.TKANMOD  
//KCIFLOW DD DISP=SHR,DSN=<plib_hlq>.TKANCUS(KFJOMEGA)  
//KCIVARS DD *  
ACTION GENERATE  
RTE_NAME RTEZOS1 [2]  
RTE_PLIB_HILEV TSOUID.MONSUITE  
/*
```

[1]

Run this job on the LPAR where you will run the hub, ZOS1.

[2]

Specify an **RTE_NAME** value to match the RTEDEF (RTEZOS1) member for the hub runtime environment.

9. Use the **GENERATE** action to create runtime members for the remote monitoring server runtime environment.
Reuse the JCL from the previous step, with the following changes:

[1]

Run the job on the LPAR where you will run the remote monitoring server, ZOS2.

[2]

Specify the corresponding **RTE_NAME** for that runtime environment, RTEZOS2.

What to do next

For details on completing the configuration of these runtime environments and then starting them, see the corresponding steps in the general procedure for [creating a runtime environment](#).

Extend this example with more LPARs and more monitoring agents. Use the **DISCOVER** action to discover subsystems on each LPAR.

This example assumed that the same global parameter values in RTEDEF (GBL\$PARM) apply to the runtime environments on both LPARs. To specify different values for different LPARs, create LPAR-specific RTEDEF (GBL\$*lpar*) members.

Batch interface

The JCL to run IBM Z® Monitoring Configuration Manager is simple and concise. You specify an action, the name of the runtime environment on which you want to perform that action, and the high-level qualifiers of the data sets for that runtime environment.

JCL

Figure 13: JCL to run Monitoring Configuration Manager

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=<lpar>
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=<tlib_hlq>.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=<tlib_hlq>.TKANCUS(KFJOMEGA)
//KCIVARS DD *
*
*
*<action> SELECT FROM:
* CREATE | DISCOVER | GENERATE | DELETE | MIGRATE | PACKAGE | DEPLOY
*
ACTION          <action>
OPTION           <option>
RTE_NAME        <rte_name>
RTE_PLIB_HILEV  <rte_plib_hilev>
/*
```

Similar JCL is supplied in the KFJJMCM member of the TKANSAM target library.

KCIOMEGA is the program that runs Monitoring Configuration Manager. The KCIFLOW data set provides input to KCIOMEGA. That is all you need to know about the KCIOMEGA program and the KCIFLOW data set to run Monitoring Configuration Manager. If you want to know more, see [“KCIOMEGA workflows” on page 73](#).

Replace the placeholders in the JCL with appropriate values:

<lpar>

Run Monitoring Configuration Manager actions on the LPAR where you will start the runtime environment. For example, if your site uses JES2, insert a **SYSAFF** job parameter after the **JOB** statement to ensure that the job runs on that LPAR.

<tlib_hlq>

The high-level qualifiers of the target libraries.

<action>

One of the Monitoring Configuration Manager [actions](#). You can abbreviate actions to their first three characters: **CRE**, **DIS**, **GEN**, **DEL**, **MIG**, **PAC**, and **DEP**.

<option>

One or more compatible Monitoring Configuration Manager [options](#). Multiple options must be specified by a comma with no spaces.

<rte_name>

Runtime environment name, 1 - 8 characters. Monitoring Configuration Manager uses this name for various purposes, including:

- MVS™ member names
- MVS™ data set name qualifiers
- z/OS® UNIX® System Services directory name, all uppercase

<rte_plib_hilev>

The high-level qualifiers of the [runtime environment definition library](#): *rte_plib_hilev*.RTEDEF

Monitoring Configuration Manager uses the values of **RTE_NAME** and **RTE_PLIB_HILEV** to set the default value of other parameters, such as **RTE_HILEV** and **RTE_VSAM_HILEV**, that are used for data set names. To avoid exceeding the z/OS 44-character limit for data set names, the combined length of **RTE_NAME** and **RTE_PLIB_HILEV** should not exceed 28 characters. For example, if **RTE_NAME** is 8 characters, then **RTE_PLIB_HILEV** should not exceed 20 characters.

RTE_NAME and RTE_PLIB_HILEV parameters versus the values in KCIVARS

The **CREATE** action uses the **RTE_NAME** and **RTE_PLIB_HILEV** values that you specify in the KCIVARS input data set as the initial values of the **RTE_NAME** and **RTE_PLIB_HILEV** parameters in the `rte_plib_hilev.RTEDEF(rte_name)` member. At that point in time, the values of the **RTE_NAME** and **RTE_PLIB_HILEV** parameters in the `RTEDEF(rte_name)` member match the values that you specified in KCIVARS.

However, you might edit the values of the **RTE_NAME** and **RTE_PLIB_HILEV** parameters in the `RTEDEF(rte_name)` member so they no longer match the values that you specified in KCIVARS.

For subsequent actions, the **RTE_NAME** and **RTE_PLIB_HILEV** values that you specify in KCIVARS are used only to locate the `rte_plib_hilev.RTEDEF(rte_name)` member. The action uses the **RTE_NAME** and **RTE_PLIB_HILEV** parameters in that `RTEDEF` library.

Contents of the KCIVARS input data set

The contents of the KCIVARS data set are **case-sensitive**: you must specify the variable names and their values exactly as described.

KCIVARS can contain comment lines and inline comments:

- Comment lines begin with an asterisk (*) in column 1.

```
* Comment line
```

- Inline comments begin with an asterisk after a variable value.

```
RTE_NAME MYRTE * Inline comment
```

Actions

IBM Z® Monitoring Configuration Manager can perform several actions. The main action, **GENERATE**, generates runtime members, which is the main purpose of Monitoring Configuration Manager. The other actions are optional, for your convenience.

You typically perform the actions in the following order:

1. **CREATE**
2. **DISCOVER**
3. **GENERATE**
4. **DELETE**

In addition, a **MIGRATE** action can be used to copy the required configuration parameters from PARMGEN for use with Monitoring Configuration Manager. When using **MIGRATE**, you typically perform the actions in the following order:

1. **MIGRATE**
2. **GENERATE**
3. **DELETE**

Finally, to support cross-sysplex rollout scenarios, the **PACKAGE** and **DEPLOY** actions can be used after the **GENERATE** action has completed successfully.

Action options

The **OPTION** parameter is available for some actions and provides more granular control over certain processing. Use of this parameter is optional.

This topic summarizes the available options for each of the respective actions, as follows. For sample JCL, see [“JCL” on page 34](#).

CREATE action

The following options are available for the **CREATE** action. For more information, see [“CREATE” on page 37](#).

MULTIPLE

Use a single Configuration Manager RTEDEF data set for multiple runtime environments. You can abbreviate this keyword to **MULTI**.

TRGCOPY

Create a target copy member in the RTEDEF library. You can abbreviate this keyword to **TRG**.

GENERATE action

Some options require a complete run of the **GENERATE** action prior to use. Additionally, some options are not compatible to run with other options during the same job.

To specify more than one option, separate the values with a comma and no spaces. For example: **OPTION USS, SECEXITS**

The following options are available for the **GENERATE** action. For details about using these options, see [“GENERATE” on page 47](#) and [“GENERATE options” on page 49](#).

USS | NOUSS

Provides control of the **GENERATE** workflow stage that deploys the parts related to z/OS® UNIX® System Services. You must perform a complete run of the **GENERATE** action before you can use **OPTION USS**.

SECEXITS | NOSECEXITS

Provides control of configuration processing for security exits. You must perform a complete run of the **GENERATE** action before you can use **OPTION SECEXITS**. You can abbreviate these keywords to **SEC** and **NS**, respectively.

VALIDATE

Performs initial validation of RTEDEF parameters. You can abbreviate this keyword to **VAL**.

QUICKLOAD

Loads the read-only configuration members to the RK* data sets. The read-only members are those members that are not impacted by customization during configuration. You can abbreviate this keyword to **QL**.

QUICKCONFIG

Updates the configurable members for the runtime environment (for example, in the RKANPARU, RKANSAMU, and RKANCMDU libraries) without refreshing data from SMP/E target libraries. You can abbreviate this keyword to **QC**.

RELINK | NORELINK

Provides control of when to assemble and link edit modules for OMEGAMON for Networks (KN3) and OMEGAMON enhanced 3270 user interface (KOB). You must perform a complete run of the **GENERATE** action before you can use **OPTION RELINK**. You can abbreviate these keywords to **LINK** and **NL**, respectively.

TRGCOPY

Copies SMP/E target libraries into target copy data sets. You can abbreviate this keyword to **TRG**.

PREPARE

Prepares intermediate work libraries for your runtime environment separately from other **GENERATE** action stages. You can use this option to build the work data sets before other **GENERATE** action stages so that other required tasks dependent on the data sets can be performed ahead of time or in parallel. You can abbreviate this keyword to **PREP**.

MIGRATE action

The following option is available for the **MIGRATE** action. For more information, see [“MIGRATE” on page 60](#).

MULTIPLE

Use a single Configuration Manager RTEDEF data set for multiple PARMGEN runtime environments. You can abbreviate this keyword to **MULTI**.

PACKAGE action

The following option is available for the **PACKAGE** action. For more information, see [“PACKAGE” on page 64](#).

NOUSS

Do not include the files and directories related to z/OS® UNIX® in the **PACKAGE** output.

DEPLOY action

The following options are available for the **DEPLOY** action. For more information, see [“DEPLOY” on page 67](#).

USS

Run only the **DEPLOY** workflow stage that deploys the parts related to z/OS® UNIX®.

NOUSS

Do not run the z/OS® UNIX® deploy stage in the **DEPLOY** action.

All actions

The following option is available for all actions.

DEBUG

This option provides output for troubleshooting purposes and should only be used under the guidance of IBM Software Support. The **DEBUG** keyword can be used with all other **OPTION** keywords. You can abbreviate this keyword to **DBG**.

CREATE

The **CREATE** action creates an initial runtime environment definition that you can customize to match your requirements.

Before you begin

Review the following information before you use the **CREATE** action:

- For an overview of how the **CREATE** action fits into the process of creating a runtime environment, see [“Creating your first, minimal runtime environment” on page 19](#).
- For information about the members that the **CREATE** action allocates and populates, see [“Initial runtime environment library members” on page 123](#).
- The **CREATE** action supports creating one or more runtime environments in a single RTEDEF configuration. It is recommended that you decide prior to the creation of your first runtime environment whether you plan to create one or multiple runtime environments in a single RTEDEF.

Note: If you are going to set up a High Availability TEMS (HA TEMS), make sure only one runtime environment is defined in the RTEDEF (that is, the one used for the HA TEMS).

Note: You can also use the **CREATE** action as part of the process to create a copy of your SMP/E target libraries. For more information, see [“Using SMP/E target library copies” on page 156](#).

About this task

The **CREATE** action creates an initial runtime environment definition by allocating and populating the necessary data sets, members, and configuration settings.

Note: Using the **CREATE** action to create your initial runtime environment definition is optional and provided for convenience; you can perform the same steps manually. Experienced users can skip **CREATE** and copy an existing RTEDEF library. You can allocate the RTEDEF library yourself using a record format of fixed-length, blocked (FB) and a record length of 80. You can also create the members; the only required member is *rte_name*.

The following list provides details about the **CREATE** action:

- The **CREATE** action allocates the runtime environment definition library, *rte_plib_hilev*. RTEDEF, if it does not already exist, and populates it with initial configuration settings. The **CREATE** action does not overwrite members. If the RTEDEF library already exists, the **CREATE** action only writes members that do not yet exist.
- You can create one or more runtime environments in a single RTEDEF configuration. The default behavior of the **CREATE** action is to create only one runtime environment in the RTEDEF data set. Using the **OPTION MULTIPLE** parameter, you can create multiple runtime environments in a single RTEDEF data set. Each runtime environment creation requires a separate **CREATE** action job. If you plan to configure multiple runtime environments in a single RTEDEF data set, make sure to include the **OPTION MULTIPLE** parameter on every **CREATE** action job, including the first one.

Note: You can abbreviate **OPTION MULTIPLE** to **OPTION MULTI**. **OPTION MULTIPLE** is not compatible with the other available **CREATE** action option, **TRGCOPY**.

- The **CREATE** action creates the necessary members in the RTEDEF data set, as follows:
 - When using the default behavior of the **CREATE** action to create one runtime environment in a single RTEDEF data set (omitting the **OPTION MULTIPLE** parameter), the **CREATE** action will create members of type *Kpp\$PARM* in the respective created RTEDEF data set, along with the *rte_name* member for the runtime environment-specific parameters.
 - When creating a runtime environment in a configuration where a single RTEDEF contains multiple runtime environments, use parameter **OPTION MULTIPLE** and **KFJ_SYSNAME lpar** in the KCIVARS DD. The **CREATE** action will create members of type *Kpp\$lpar* in the RTEDEF data set, along with the *rte_name* member for the runtime environment-specific parameters.

Note: When using **OPTION MULTIPLE**, the **CREATE** action creates the *Kpp\$lpar* members automatically. It is recommended that you create the *Kpp\$lpar* members as well and add the parameter values that are the same for all runtime environments in the given RTEDEF library.

On subsequent runs of the **CREATE** action, reuse the same **RTE_PLIB_HILEV** parameter value, but update the values for parameters **RTE_NAME** and **KFJ_SYSNAME** to create a new set of runtime environment parameter members. There is no limit on how many runtime environments can be configured in a single RTEDEF data set.

- The **CREATE** action allocates the security exits library with the default name *rte_plib_hilev.rte_name*. SECEXITS (or, optionally, the name specified in the **KFJ_SECURITY_EXITS_LIB** parameter). The **CREATE** action also populates the security exits library with default security exits members and defines the library to the runtime environment using the **RTE_X_SECURITY_EXIT_LIB** parameter. For more information, see [“Setting up security exits in your runtime environment” on page 139](#).
- If the use of override embed members is enabled by specifying parameter **KFJ_USE_EMBEDS** set to Y, the **CREATE** action allocates the embeds data set with the default name *rte_plib_hilev.rte_name*. EMBEDS (or, optionally, the name specified in the **KFJ_EMBEDS_LIB** parameter). The **CREATE** action sets up the embeds data set, populates it with supported override embed parameters (if applicable), and defines it to the runtime environment using the **RTE_X_OVERRIDE_EMBEDS_LIB** parameter. For more information, see [“Using override embed members in Configuration Manager” on page 140](#).
- The **CREATE** action works with the **KFJ_LOCAL_PLIB_HILEV** parameter to allow for local generation of runtime environments for remote systems using different high-level qualifiers.

When the **KFJ_LOCAL_PLIB_HILEV** parameter is specified, the generated *kfj_local_plib_hilev*.RTEDEF data set will contain an additional member: PCK\$PARM for a default (single) **CREATE** action, or member PCK\$*lpar* in a multiple **CREATE** action. This member allows locally generated runtime environments using a different data set high-level qualifier than the one intended to be used on the deployment target (for example, the production system).

For more information about remote deployments, see [“Special considerations for SYSPLEX rollout” on page 124](#), [“RTEDEF\(PCK\\$PARM\)” on page 126](#), and [“Deploying remote runtime environments” on page 145](#).

To create a runtime environment definition using the **CREATE** action, use the following procedure.

Procedure

1. Modify the KFJJMCM sample job in TKANSAM (see example below) to select a **CREATE** action.
2. Specify values for the required parameters **RTE_NAME** and **RTE_PLIB_HILEV**.
3. (Optional) Specify additional parameters as needed, for example:
 - To create this runtime environment in a RTEDEF data set that does or will contain multiple runtime environment configurations, add the **OPTION MULTIPLE** and **KFJ_SYSNAME lpar** parameters. If this is a subsequent run of the **CREATE** action, reuse the same **RTE_PLIB_HILEV** parameter value, but update the values for parameters **RTE_NAME** and **KFJ_SYSNAME**.
 - To specify a different name for the security exits library, add the **KFJ_SECURITY_EXITS_LIB** parameter and value.
 - To enable the use of override embed members, add the **KFJ_USE_EMBEDS** parameter set to Y and the **KFJ_EMBEDS_LIB** parameter and value.
4. Run the KFJJMCM job to generate and populate the RTEDEF data set and other required data sets. Job messages for the **CREATE** action are written to the KCIPRINT SYSOUT data set.

Creating runtime environment definition for one LPAR

The following JCL jobs create the runtime environment definition library TSOUID.MONSUITE.RTEDEF and populate it with various members, including the runtime environment configuration profile member RTE1. The first example is for a single runtime environment RTEDEF, and the second example is for a multiple runtime environment RTEDEF. These examples also specify that override embed members are enabled and provide custom data set names for the security exits and embeds libraries.

Figure 14: Example JCL to perform the **CREATE** action (one LPAR) for a single runtime environment RTEDEF

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUIITE.TKANMOD
//KCIFFLOW DD DISP=SHR,DSN=MONSUIITE.TKANCUS(KFJOMEGA)
//KCIIVARS DD *
*
ACTION                CREATE
RTE_NAME              RTE1
RTE_PLIB_HILEV        TSOUID.MONSUITE

KFJ_SECURITY_EXITS_LIB TEST1.TST.DEMO.MYEXITS
KFJ_USE_EMBEDS        Y
KFJ_EMBEDS_LIB        TEST1.TST.DEMO.MYEMBEDS
/*
```

Figure 15: Example JCL to perform the **CREATE** action (one LPAR) for a multiple runtime environment RTEDEF

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUIITE.TKANMOD
//KCIFFLOW DD DISP=SHR,DSN=MONSUIITE.TKANCUS(KFJOMEGA)
//KCIIVARS DD *
*
ACTION                CREATE
OPTION                MULTIPLE
RTE_NAME              RTE1
RTE_PLIB_HILEV        TSOUID.MONSUITE

KFJ_SECURITY_EXITS_LIB TEST1.TST.DEMO.MYEXITS
```

```

KFJ_USE_EMBEDS          Y
KFJ_EMBEDS_LIB          TEST1.TST.DEMO.MYEMBEDS

KFJ_SYSNAME             lpar
/*

```

Creating a runtime environment definition for remote deployment

The following JCL creates the runtime environment definition library TSOUID.DEV.RTEDEF. In addition to the standard members, it will contain member PCK\$PARM, where you will find all available **KFJ_LOCAL_*** parameters.

Figure 16: Example JCL to perform the **CREATE** action (remote deployment)

```

//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUITE.TKANMOD
//KCIFFLOW DD DISP=SHR,DSN=MONSUITE.TKANCUS(KFJOMEGA)
//KCIIVARS DD *
*
ACTION                CREATE
RTE_NAME              RTE1
RTE_PLIB_HILEV        TSOUID.PROD
KFJ_LOCAL_PLIB_HILEV TSOUID.DEV
/*

```

Note: If **OPTION MULTIPLE** is used, the **CREATE** action creates member PCK\$lpar instead of PCK\$PARM.

DISCOVER

The **DISCOVER** action discovers CICS® regions, Db2® subsystems, IMS control regions, MQ subsystems, TCP/IP stacks and system symbols on an LPAR, and then creates corresponding members in the runtime environment definition library.

Before you begin

This documentation uses *subsystem* as an informal collective term for CICS® regions, Db2® subsystems, IMS control regions, MQ subsystems, and TCP/IP stacks.

Discovery is not limited to the products that you have configured using **CONFIGURE_*** parameters. The **DISCOVER** action always discovers all the subsystems it can.

It is recommended that you review the following topics, which describe the results of the **DISCOVER** action:

- [“Parameters created by the DISCOVER action” on page 42](#)
- [“Members created by the DISCOVER action” on page 43](#)

Tip: Bring up all the subsystems you intend to monitor before you run the **DISCOVER** action. You can re-run the **DISCOVER** action if any subsystems are added or started on your LPAR after your initial discovery run. In this case, you need to consolidate the respective discovery output in your RTEDEF data set (that is, [merge the generated members](#) accordingly).

About this task

The **DISCOVER** action discovers subsystems and system symbols on an LPAR, and then creates corresponding members in the runtime environment definition library (*rte_plib_hilev*.RTEDEF) and populates the members with the discovered data.

The following list provides details about the **DISCOVER** action:

- Run the **DISCOVER** action on the LPAR whose subsystems you want to discover. For example, insert a JES2 **SYSAFF** job parameter after the **JOB** statement to ensure that the job runs on the correct LPAR:

```
/*JOBPARM SYSAFF=<lpar>
```

- Use the KCIALPHA program to discover more details. Optionally, for the **DISCOVER** action only, change the program name in the JCL **EXEC** statement from KCIOMEGA to KCIALPHA. KCIALPHA is an APF-authorized version of KCIOMEGA. APF authorization enables KCIALPHA to discover more subsystem details, which would otherwise need to be entered manually. If KCIALPHA is run from a load library that is not APF authorized, partial discovery will still be completed, but the job will end with return code 8.
- The **DISCOVER** action can run stand-alone, that is, without requiring an existing RTEDEF data set as pointed to by the **RTE_PLIB_HILEV** parameter. The resulting RTEDEF will only contain the members produced by discovery and will use the default prefixes for VTAM applids and started task definitions. For more information about this use case, see the example [“Stand-alone discovery using defaults” on page 41](#).
- For some subsystem data sets, if an alias is defined for the data set, the **DISCOVER** action will use the alias name instead of the original data set name. The following parameters will be populated with alias names if they exist: **KD2_DBnn_DB2_LOADLIB**, **KI2_I1nn_CLASSIC_IMS_RESLIB**, **KN3_TCPXnn_TCPIP_PROFILES_DSN**.
- After performing a **DISCOVER** action, review the messages about discovery in the KCIPRINT SYSOUT data set.
- The output of the **DISCOVER** action depends on the subsystem. For more information, see the following topics:
 - [“Parameters created by the DISCOVER action” on page 42](#)
 - [“Members created by the DISCOVER action” on page 43](#)
- On the first run, the **DISCOVER** action creates members for each type of subsystem it discovers. On subsequent runs, if the discovery member for a subsystem exists, it will not be overwritten, but instead a comment member (*Kpp#lpar*) will be created. You must then review the members and manually apply the updates that you want to keep. For more information, see [“First-time discovery versus rediscovery” on page 44](#).

To run a discovery using the **DISCOVER** action, use the following procedure.

Procedure

1. Modify the KFJJMCM sample job in TKANSAM (see example below) to select a **DISCOVER** action.
2. Specify values for the required parameters **RTE_NAME** and **RTE_PLIB_HILEV**.
3. Run the KFJJMCM job to discover the available subsystems.
The **DISCOVER** action creates the discovery members in the RTEDEF data set and populates the parameters with discovered values. Job messages for the **DISCOVER** action are written to the KCIPRINT SYSOUT data set.
4. Review the content in the discovery members and update as necessary.

Discovering subsystems

The following JCL discovers subsystems on the LPAR ZOS1 and creates corresponding members in TSQUID.MONSUITE.RTEDEF. For example, if the **DISCOVER** action discovers Db2® subsystems, then it creates the member KD5@ZOS1; or, if that member already exists, KD5#ZOS1.

Figure 17: Example JCL to perform the **DISCOVER** action

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIALPHA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUITE.TKANMOD
//KCIFFLOW DD DISP=SHR,DSN=MONSUITE.TKANCUS(KFJOMEGA)
//KCIIVARS DD *
ACTION DISCOVER
RTE_NAME RTE1
RTE_PLIB_HILEV TSQUID.MONSUITE
/*
```

The JES2 **SYSAFF** job parameter ensures that the job runs, and the **DISCOVER** action discovers subsystems, on LPAR ZOS1.

Stand-alone discovery using defaults

The following JCL is used to perform a stand-alone discovery that uses default values.

If you want to discover IMS systems using the stand-alone run of the **DISCOVER** action and specify different prefixes for started task and VTAM applids, specify the prefixes of your choice in the **RTE_VTAM_APPLID_PREFIX** and **RTE_STC_PREFIX** parameters. This way the VTAM and started task prefixes will be honored accordingly in member RTEDEF (*KI5@lpar*).

Figure 18: Example JCL to perform a stand-alone discovery using the **DISCOVER** action

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIALPHA,REGION=OM,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUITE.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=MONSUITE.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION                DISCOVER
RTE_NAME              RTE1
RTE_PLIB_HILEV        TSUID.MONSUITE

RTE_STC_PREFIX        OMEG
RTE_VTAM_APPLID_PREFIX OM&SYSCLONE.
/*
```

If you do not have any SMP/E target libraries on the system on which you want to run a stand-alone **DISCOVER** action, you can use the utility, TKANSAM (KFJMAINT), with action **BLDREMDS** to build the necessary minimum data sets (TKANSAM, TKANMOD, and TKANCUS libraries) needed to run the action. Make sure you transfer the created data sets to your remote system where the stand-alone **DISCOVER** should run and where the necessary APF authorization of the TKANMOD library is made.

Parameters created by the DISCOVER action

The **DISCOVER** action creates parameters for each subsystem it discovers.

Overview of output

The output of the **DISCOVER** action depends on the subsystem:

Table 4: Output of the DISCOVER action	
Subsystem	Output
Db2®, IMS, TCP/IP	Parameters that configure the runtime environment to monitor these subsystems. For Db2®, you need to complete the discovered details .
CICS®	Parameters that specify an historical datastore allocation table for task history file disposition.
MQ	Statements using embed overrides that you can place in the KMQ\$CUSR embed override member: <pre>SET MANAGER NAME(queue-manager)</pre>

In addition to the subsystem discovery, a member is created that contains the System symbols that can be used by the **GENERATE** action along with the respective **KCIPARSE** extracted variables. For more information, see “[RTEDEF\(SYS@lpar\)](#)” on page 46.

Parameters

The following table lists the parameters created by the **DISCOVER** action.

The parameters created depend on which program performs the action: KCIOMEGA or KCIALPHA.

In general, KCIOMEGA discovers only the subsystem identifier. For detailed discovery, use KCIALPHA.

Table 5: Parameters created by the DISCOVER action			
Subsystem	Parameter (<i>nn</i> is the 2-digit table row number)	KCIOMEGA	KCIALPHA
CICS®	KC2_HSnn_CLASSIC_CICS_REGION	No	Yes

Subsystem	Parameter (<i>nn</i> is the 2-digit table row number)	KCIOMEGA	KCIALPHA
Db2®	KD2_DBnn_DB2_SSID	Yes	Yes
Db2®	KD2_DBnn_DB2_VER	No	Yes
Db2®	KD2_DBnn_DB2_DS_GROUP	No	Yes
Db2®	KD2_DBnn_DB2_LOADLIB	No	Yes
IMS	KI2_I1nn_CLASSIC_IMSID	Yes	Yes
IMS	KI2_I1nn_CLASSIC_IMS_RESLIB	No	Yes
MQ	For MQ subsystems, DISCOVER generates SET MANAGER statements, not parameters.	Yes	Yes
TCP/IP	KN3_TCPXnn_TCP_STC	Yes	Yes
TCP/IP	KN3_TCPXnn_TCPIP_PROFILES_DSN	No	Yes
TCP/IP	KN3_TCPXnn_TCPIP_PROFILES_MBR	No	Yes

Members created by the DISCOVER action

The **DISCOVER** action creates members for each type of subsystem it discovers.

Table 6: Members created by the DISCOVER action	
Subsystem	Member name
CICS®	KC5@ <i>lpar</i> for first-time discovery KC5# <i>lpar</i> for rediscovery
Db2®	KD5@ <i>lpar</i> for first-time discovery KD5# <i>lpar</i> for rediscovery
IMS	KI5@ <i>lpar</i> for first-time discovery KI5# <i>lpar</i> for rediscovery
MQ	KMQ# <i>lpar</i> Note: For MQ subsystems, only the KMQ# <i>lpar</i> member is created for both first-time discovery and rediscovery. For more information, see “RTEDEF(KMQ# <i>lpar</i>)” on page 45.
TCP/IP	KN3@ <i>lpar</i> for first-time discovery KN3# <i>lpar</i> for rediscovery
Symbols	SYS@ <i>lpar</i> for first-time discovery SYS# <i>lpar</i> for rediscovery

Member naming convention

For subsystem and symbols discovery, the **DISCOVER** action creates RTEDEF library members with the following naming convention:

Prefix

For subsystem type: *Kpp* from the corresponding **CONFIGURE_*** parameter.
For symbols: SYS.

Separator

An at sign (@) or a hash (#).
A hash indicates that the member is a comment: the **GENERATE** action ignores these members.

Note: The at sign (@) appears in only some code pages. For example, when using CCSID 1141, it appears as "§" (0x7C).

Suffix

Identifies the LPAR.

First-time discovery versus rediscovery

The **DISCOVER** action does not overwrite *Kpp@lpar* or *SYS@lpar* members (“@” members).

If a *Kpp@lpar* or *SYS@lpar* member already exists, the **DISCOVER** action writes a comment member, *Kpp#lpar* or *SYS#lpar*, respectively, and then continues, eventually completing with return code 4. Review the comments about discovery in the KCIPRINT sysout data set, and then review the “#” members. Edit the existing “@” members to apply any preferred updates.

The **DISCOVER** action overwrites comment members.

RTEDEF (KC5@lpar)

If the **DISCOVER** action discovers CICS® regions, it creates the RTEDEF (KC5@lpar) member. This member contains parameters that configure the CICS® monitoring agent.

Figure 19: RTEDEF (KC5@lpar) member created by the DISCOVER action

```
* CICS regions discovered by KCIOMEGA
* SYSPLEX=<sysplex> LPAR=<lpar> DATE=<date>

KC2_HS                BEGIN                * Table begin *

KC2_HS01_ROW          01
KC2_HS01_CLASSIC_CICS_REGION "<region_name>"
KC2_HS01_CLASSIC_VSAM_CYL 1                [1]

* More rows (one for each region discovered)...

KC2_HS                END                * Table end *
```

[1] KC2_HS01_CLASSIC_VSAM_CYL

If necessary, replace the initial value of 1 with an appropriate value for your site.

RTEDEF (KD5@lpar)

If the **DISCOVER** action discovers Db2® subsystems, it creates the RTEDEF (KD5@lpar) member. This member contains parameters that configure the Db2® monitoring agent.

Figure 20: RTEDEF (KD5@lpar) member created by the DISCOVER action

```
* Db2 subsystems discovered by KCIOMEGA
* SYSPLEX=<sysplex> LPAR=<lpar> DATE=<date>

KD2_DB                BEGIN                * Table begin *

KD2_DB01_ROW          01
KD2_DB01_DB2_SSID     "<ssid>"
KD2_DB01_DB2_DESCRIPTION "<ssid> Db2 subsystem"
KD2_DB01_DB2_VER      "<version>"
KD2_DB01_DB2_SYSNAME  "<lpar>"
KD2_DB01_DB2_PROFID   "P001"
KD2_DB01_DB2_DS_GROUP "N"
KD2_DB01_DB2_MONITOR_START "N"
KD2_DB01_DB2_PORT_NUM "2000"
KD2_DB01_DB2_LOADLIB  "<dsname>"                [1]
KD2_DB01_DB2_DSNTIAD  "DSNTIAD"
KD2_DB01_DB2_RUNLIB   ""

* More rows (one for each subsystem discovered)...
```

```
KD2_DB                                END                                * Table end *
```

The **DISCOVER** action only discovers the parameter values shown in the previous figure in *italics*. You must complete or edit the other parameters.

[1] KD2_DBnn_DB2_LOADLIB

If an alias is defined for the data set, the alias name will be used for the parameter value instead of the original data set name.

RTEDEF (KI5@lpar)

If the **DISCOVER** action discovers IMS control regions, it creates the RTEDEF (KI5@lpar) member. This member contains parameters that configure the IMS monitoring agent.

Figure 21: RTEDEF (KI5@lpar) member created by the **DISCOVER** action

```
* IMS control regions discovered by KCIOMEGA
* SYSPLEX=<sysplex> LPAR=<lpar> DATE=<date>

KI2_I1                                BEGIN                                * Table begin *

KI2_I101_ROW                          01
KI2_I101_CLASSIC IMSID                 "<imsid>"
KI2_I101_CLASSIC MPREFIX               "M<n>"
KI2_I101_CLASSIC GLOBAL                ""
KI2_I101_CLASSIC STC                   "<rte_stc_prefix>0I<n>"
KI2_I101_CLASSIC VTAM_NODE             "<rte_vtam_applid_prefix>0I<n>N"
KI2_I101_CLASSIC VTAM_APPL LOGON      "<rte_vtam_applid_prefix>0I<n>"
KI2_I101_CLASSIC IMS_RESLIB            "<dsname>"                                [1]
KI2_I101_CLASSIC LROWS                 "255"
KI2_I101_CLASSIC USER_PROFILE          "/C"
KI2_I101_CLASSIC CTRL_UNIT_ADDR        "XXXX"

* More rows (one for each control region discovered)...

KI2_I1                                END                                * Table end *
```

The **DISCOVER** action only discovers the values of the **KI2_I101_CLASSIC_IMSID** and **KI2_I101_CLASSIC_IMS_RESLIB** parameters.

For other parameters, **DISCOVER** generates placeholder values. Review these values and, if necessary, edit them to match your site requirements.

[1] KI2_I1nn_CLASSIC_IMS_RESLIB

If an alias is defined for the data set, the alias name will be used for the parameter value instead of the original data set name.

RTEDEF (KMQ#lpar)

If the **DISCOVER** action discovers MQ subsystems, it creates the RTEDEF (KMQ#lpar) member. This member contains **SET MANAGER** statements that you can use in the KMQ\$CUSR embed override member.

Figure 22: RTEDEF (KMQ#lpar) member created by the **DISCOVER** action

```
* MQ subsystems discovered by KCIOMEGA
* SYSPLEX=<sysplex> LPAR=<lpar> DATE=<date>

SET MANAGER NAME(*)

*SET MANAGER NAME(queue-manager)

* More SET MANAGER statements (one for each queue manager discovered)...
```

When discovering MQ subsystems, the **DISCOVER** action creates only a comment member, as indicated by the hash (#) in the member name. This is unlike the discovery of other types of subsystems, where the **DISCOVER** action creates members that contain configuration parameters and have the at sign (@) in the member name. Discovery of MQ subsystems is for informational purposes only.

The IBM OMEGAMON® for Messaging on z/OS, IBM MQ Monitoring agent performs its own discovery and, by default, will monitor any queue manager on the LPAR. The default setting of SET MANAGER NAME(*) is recommended because it allows you to change your queue manager configuration without having to change agent parameters.

If you want to name particular queue managers, you can specify multiple statements of SET MANAGER NAME(*queue-manager*).

Because it is a comment member, the information in KMQ#*lpar* is not used for agent configuration directly. However, you can use the information that was discovered and is provided in the KMQ#*lpar* member in your KMQ\$CUSR embed override member.

RTEDEF (KN3@*lpar*)

If the **DISCOVER** action discovers TCP/IP stacks, it creates the RTEDEF (KN3@*lpar*) member. This member contains parameters that configure the networks monitoring agent.

Figure 23: RTEDEF (KN3@*lpar*) member created by the **DISCOVER** action

```
* TCP/IP stacks discovered by KCIOMEGA
* SYSPLEX=<sysplex> LPAR=<lpar> DATE=<date>

KN3_TCPX                BEGIN                * Table begin *

KN3_TCPX01_ROW          01
KN3_TCPX01_TCP_STC     "<task-name>"
KN3_TCPX01_SYS_NAME    "<lpar>"
KN3_TCPX01_TCPIP_PROFILES_DSN  "<dsname>"                [1]

KN3_TCPX01_TCPIP_PROFILES_MBR  "<member-name>"

* More rows (one for each stack discovered)...

KN3_TCPX                END                * Table end *
```

[1] KN3_TCPXnn_TCPIP_PROFILES_DSN

If an alias is defined for the data set, the alias name will be used for the parameter value instead of the original data set name.

RTEDEF (SYS@*lpar*)

The **DISCOVER** action creates the RTEDEF (SYS@*lpar*) member. This member contains system symbols and KCIPARSE extracted variables needed by the **GENERATE** action.

The primary use is, in conjunction with parameter **KFJ_SYSNAME**, to allow you to pre-generate RTEs on a single local system without having to run the **GENERATE** step on the remote system.

Figure 24: RTEDEF (SYS@*lpar*) member created by the **DISCOVER** action

```
* System Symbols discovered by KCIOMEGA
* SYSPLEX=sysplex LPAR=lpar DATE=date

HLQPLEX                "hlqplex"                'S'
HSMHOST                "2"                'S'
HSMPRI                 "YES"                'S'
NUMSYS                 "2"                'S'
PLEXNAME               "plexname"          'S'
RTHLQ                  "RSRTE"            'S'
SMFID                  "smfid"            'S'
SUFFIX                 "RS"                'S'
SYSALVL                "2"                'S'
SYSCclone              "B2"                'S'
SYSLVL                 "205"              'S'
SYSNAME                "name"              'S'
SYS0SLVL               "Z1020500"          'S'
SYSPLEX                "sysplex"          'S'
SYSR1                  "RZ205A"           'S'
UNIXVER                "V2R05"            'S'
ZDDD                   "SAT."              'S'
...
SYSSMFID               "sysmfid"           'K'
```

SYSSMS	"Y"	'K'
SYSVTAMNETID	"TESTNET1"	'K'
SYSVTAMSSCP	"RSB2SSCP"	'K'
SYSIP	"Y"	'K'
SYSIPADDRESS	"123.456.78"	'K'
SYSIPADDRESS_F	"123.456.789.012"	'K'
SYSIPHOSTNAME	"hostname"	'K'

GENERATE

The **GENERATE** action generates runtime members for a runtime environment from a set of configured parameters.

Before you begin

Before performing a **GENERATE** action for an existing runtime environment, stop the started tasks for that runtime environment. Started tasks can lock runtime members, such as persistent data store data sets. Locked runtime members can cause the **GENERATE** action to fail.

Note: You can also use the **GENERATE** action to make a copy of your SMP/E target libraries. For more information, see [“Using SMP/E target library copies” on page 156](#).

If you are familiar with PARMGEN: PARMGEN uses several scenarios to perform maintenance actions. With Configuration Manager, numerous maintenance scenarios are not needed. The **GENERATE** action performs in one job what previously required multiple steps for the various maintenance scenarios. All of the PARMGEN scenarios are replaced by the **GENERATE** action.

About this task

With the **GENERATE** action, you can generate the runtime members and started tasks for your runtime environment.

The following list provides details about the **GENERATE** action:

- The **GENERATE** action generates runtime members from the parameters in the runtime environment definition library, *rte_plib_hilev*. RTEDEF. The **GENERATE** action builds the set of parameters that is used by concatenating the corresponding **RTEDEF** library members.
- Run the **GENERATE** action on the LPAR where you will start the runtime environment. For example, if your site uses JES2, insert a **SYSAFF** job parameter after the **JOB** statement to ensure that the job runs on that LPAR.
- If the RTEDEF library contains LPAR-specific members, then the **GENERATE** action uses the LPAR-specific members for the LPAR where the **GENERATE** action is running. For example, consider the RTEDEF library that contains the following members:
 - KDS\$PARM
 - KDS\$ZOS1
 - KDS\$ZOS2
 If you run the **GENERATE** action on LPAR ZOS1, then the **GENERATE** action uses the non-LPAR-specific member KDS\$PARM and the LPAR-specific member KDS\$ZOS1, but not the LPAR-specific member KDS\$ZOS2.
- Specify **KFJ_LOCAL_PLIB_HILEV** in the KCIIVARS DD, along with **KFJ_SYSNAME**, if you are generating a runtime environment that will be deployed using a different local high-level qualifier. For more information about this remote deployment scenario, see [“Remote deployment example using local libraries” on page 147](#).

Important: If you are using different settings for the local generation of the runtime environment, there are some limitations regarding the parameters that can be customized. Certain parameters are

not allowed in the respective RTEDEF members. If any of these parameters are explicitly specified in the RTEDEF members, message **KFJ00213E** is returned in KCIPRINT and the workflow stops. To continue, remove these parameters from the RTEDEF data set members and re-run the **GENERATE** action. For the list of parameters, see [“Parameters that cannot be customized for remote deployment” on page 154](#).

- You can use the **OPTION** parameter for the **GENERATE** action to control certain processing. For information about the available options, see [“GENERATE options” on page 49](#).
- After you run the **GENERATE** action, you can review a list of the members that were not replaced when the work (WK*) libraries were copied to the runtime (RK*) libraries due to having been defined in an exclude list. The list of members is provided in the exclude report, which is identified by the EXCLRPT DD statement. The exclude report is generated only for the **GENERATE** action without any options specified, or with only **OPTION NOUSS** or **OPTION QUICKCONFIG** specified. For more information, see [“Members excluded during GENERATE action” on page 55](#).

To generate runtime members for a runtime environment using the **GENERATE** action, use the following procedure.

Procedure

1. Modify the KFJJMCM sample job in TKANSAM (see example below) to select a **GENERATE** action.
2. Specify values for the required parameters **RTE_NAME** and **RTE_PLIB_HILEV**.
3. (Optional) To generate a runtime environment that will be deployed using a different local high-level qualifier, specify **KFJ_LOCAL_PLIB_HILEV** and **KFJ_SYSNAME**.
4. (Optional) To validate your RTEDEF data set before running the **GENERATE** action, add **OPTION VALIDATE**, then run the KFJJMCM job. Review the job output and correct any identified problem.

Note: You cannot use the **VALIDATE** option with any other **OPTION** value.

5. (Optional) To prepare intermediate work libraries, specify **OPTION PREPARE**, then run the KFJJMCM job.

Note: You cannot use the **PREPARE** option with any other **OPTION** value.

6. (Optional) To bypass select configuration steps, specify the following options using the **OPTION** parameter, as needed. To specify multiple options, separate the values with a comma and no spaces.
 - To bypass configuration processing for z/OS® UNIX® System Services, specify option **NOUSS**.
 - To bypass configuration processing for security exits, specify option **NOSECXITS**.
 - To bypass assembling and linking IBM Z OMEGAMON AI for Networks (KN3) or OMEGAMON enhanced 3270 user interface (KOB) modules, specify option **NORELINK**.
7. To run the **GENERATE** action to create the runtime members, remove **OPTION VALIDATE** (if present), and then run the KFJJMCM job. Job messages for the **GENERATE** action are written to the KCIPRINT SYSOUT data set.
8. (Optional) On subsequent runs of the **GENERATE** action, you can perform select configuration steps by specifying the following options using the **OPTION** parameter, as needed. To specify multiple options, separate the values with a comma and no spaces.
 - To run only the **GENERATE** workflow stage that deploys the parts related to z/OS® UNIX®, specify option **USS**.
 - To perform configuration processing for security exits only, specify option **SECXITS**.
 - To perform only the step that loads the read-only configuration members to the RK* data sets, specify option **QUICKLOAD**.

Note: You can use the **QUICKLOAD** option with the **USS** and **SECEXITS** options, but you cannot use it with the **NOUSS** option.

- To update the configurable members for the runtime environment (for example, in the RKANPARU, RKANSAMU, and RKANCMDU libraries) without refreshing data from SMP/E target libraries, specify option **QUICKCONFIG**.

Note: You cannot use the **QUICKCONFIG** option with the **VALIDATE** option or the **NOUSS** option.

- To only assemble and link elements, specify option **RELINK**.

The following JCL generates runtime members for the runtime environment that is defined by members of the TSOUID.MONSUITE.RTEDEF library, including RTE1 and LPAR-specific configuration profile members such as Kpp\$ZOS1.

Figure 25: Example JCL to perform the **GENERATE** action

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUITE.TKANMOD
//KCIFFLOW DD DISP=SHR,DSN=MONSUITE.TKANCUS(KFJOMEGA)
//KCIIVARS DD *
ACTION GENERATE
RTE_NAME RTE1
RTE_PLIB_HILEV TSOUID.MONSUITE
/*
```

Note the JES2 **SYSAFF** parameter that causes the job to run on LPAR ZOS1.

GENERATE options

You can use the **OPTION** parameter for the **GENERATE** action to control certain processing. This topic describes the available options for use with the **GENERATE** action.

Overview of GENERATE options

“[Overview of GENERATE options](#)” on page 49 provides an overview of the **GENERATE** options that are available. Each of the available options is described in more detail in “[GENERATE OPTION keywords](#)” on page 50.

OPTION	Abbreviation	Description
<u>USS</u>	–	Run only the GENERATE workflow stage that deploys the parts related to z/OS® UNIX® System Services. This option requires a complete run of the GENERATE action prior to use.
<u>NOUSS</u>	–	Do not run the z/OS® UNIX® deploy stage in the GENERATE action.
<u>SECEXITS</u>	SEC	Perform configuration processing for security exits only. This option requires a complete run of the GENERATE action prior to use.
<u>NOSECEXITS</u>	NS	Do not perform configuration processing for security exits.
<u>VALIDATE</u>	VAL	Perform validation of RTEDEF parameters.
<u>QUICKLOAD</u>	QL	Load the read-only configuration members to the RK* data sets. The read-only members are those members that are not impacted by customization during configuration. This option requires a complete run of the GENERATE action prior to use.

OPTION	Abbreviation	Description
<u>QUICKCONFIG</u>	QC	Update the configurable members for the runtime environment (for example, in the RKANPARU, RKANSAMU, and RKANCMU libraries) without refreshing data from SMP/E target libraries. This option requires a complete run of the GENERATE action prior to use.
<u>RELINK</u>	LINK	Assemble and link edit modules for OMEGAMON for Networks (KN3) and OMEGAMON enhanced 3270 user interface (KOB). This option requires a complete run of the GENERATE action prior to use.
<u>NORELINK</u>	NL	Do not assemble and link edit modules for OMEGAMON for Networks (KN3) and OMEGAMON enhanced 3270 user interface (KOB).
<u>TRGCOPY</u>	TRG	Make a copy of your SMP/E target libraries, from which you can create or update your runtime environments.
<u>PREPARE</u>	PREP	Prepare work data sets for your runtime environment. You can use this option to build the work data sets before other GENERATE action stages, in preparation of OMEGAMON installation or maintenance.

To specify more than one option, separate the values with a comma and no spaces. For example: **OPTION USS, SECEXITS**

Compatibility of GENERATE options

Some options are not compatible to run with other options during the same job. “[Compatibility of GENERATE options](#)” on page 50 indicates which options are compatible. **Y** in the table cell indicates that you can specify the two options together in the same job. **N** in the table cell indicates that you cannot specify the two options together in the same job.

Table 8: Compatibility of GENERATE options

	USS	NOUSS	SECEXITS	NOSECEXITS	VALIDATE	QUICKLOAD	QUICKCONFIG	RELINK	NORELINK	TRGCOPY	PREPARE
USS	—	N	Y	N	N	Y	Y	Y	N	N	N
NOUSS	N	—	N	Y	N	N	N	N	Y	Y	N
SECEXITS	Y	N	—	N	N	Y	Y	Y	N	N	N
NOSECEXITS	N	Y	N	—	N	N	N	N	Y	N	N
VALIDATE	N	N	N	N	—	N	N	N	N	N	N
QUICKLOAD	Y	N	Y	N	N	—	Y	Y	N	N	N
QUICKCONFIG	Y	N	Y	N	N	Y	—	Y	N	N	N
RELINK	Y	N	Y	N	N	Y	Y	—	N	N	N
NORELINK	N	Y	N	Y	N	N	N	N	—	N	N
TRGCOPY	N	Y	N	N	N	N	N	N	N	—	N
PREPARE	N	N	N	N	N	N	N	N	N	N	—

GENERATE OPTION keywords

USS | NOUSS

You can optionally control when to run the **GENERATE** workflow stage that deploys the parts related to z/OS® UNIX®. By default, the **GENERATE** action automatically performs any required z/OS® UNIX® configuration. However, there might be cases where you would want to skip this step (for reasons such as authorization issues), and perform this step at a later time. You can use the **OPTION** parameter to specify the **USS** or **NOUSS** value to control this processing, as follows:

USS

When **OPTION USS** is specified, the **GENERATE** action runs only the stage that deploys the parts related to z/OS® UNIX®; no other processing is performed (unless another **OPTION** value is also specified, such as **SECEXITS**). You cannot use this option on the initial run of the **GENERATE** action for a new runtime environment; it can only be used on subsequent runs.

Important:

- You must perform a complete run of the **GENERATE** action before you can use **OPTION USS**.

- If you have changed any parameters inside your RTEDEF members that impact z/OS® UNIX® configuration, do not use **OPTION USS**, because it will not capture your parameter changes. Instead, run action **GENERATE** with **OPTION NOUSS** (or without the option) to reconfigure the members related to z/OS® UNIX®. Then, you can use **OPTION USS** to move the updated members to z/OS® UNIX®.

NOUSS

When **OPTION NOUSS** is specified, the **GENERATE** action does not run the stage that deploys the parts related to z/OS® UNIX®. You can use this option on the initial run of the **GENERATE** action for a new runtime environment, as well as on subsequent runs.

SECEXITS | NOSECEXITS

You can optionally control when to perform configuration processing for security exits. By default, the **GENERATE** action automatically performs the required configuration tasks for security exits, which includes rebuilding and relinking the security exits into the RKANMODU load library. By performing these tasks separately from other stages, you can save valuable CPU cycles and isolate when additional authorization is needed, which might be the case when updating the load library.

You can use the **OPTION** parameter to specify the **SECEXITS** or **NOSECEXITS** value to control this processing, as follows:

SECEXITS

When **OPTION SECEXITS** is specified, the **GENERATE** action performs configuration for security exits only; no other processing is performed (unless another **OPTION** value is also specified, such as **USS**). You cannot use this option on the initial run of the **GENERATE** action for a new runtime environment; it can only be used on subsequent runs.

Important: You must perform a complete run of the **GENERATE** action before you can use the **SECEXITS** option.

NOSECEXITS

When **OPTION NOSECEXITS** is specified, the **GENERATE** action does not perform configuration processing for security exits. You can use this option on the initial run of the **GENERATE** action for a new runtime environment, as well as on subsequent runs.

Tip: This option does not update the RKANMODU load library, a data set that might require more restrictive authorization for updating. If you use this option together with option **NORELINK**, the **GENERATE** action does not update the RKANMODU library, and you can perform the other **GENERATE** configuration steps that do not require additional access.

For more information about creating your security exits library, see [“Setting up security exits in your runtime environment” on page 139](#).

VALIDATE

OPTION VALIDATE performs a validation of the parameters specified in the RTEDEF data set. Using this option helps you to prepare for running the full **GENERATE** action. You cannot use the **VALIDATE** option with any other **OPTION** value.

To ensure that your configuration is correct, **OPTION VALIDATE** performs the following steps:

- Verifies that all parameters have correct values defined and that no required parameters are missing. If any problems are found, the sysout data sets include the parameter validation report (**\$VALRPT**). If this report is generated, review the report and correct the identified parameters.
- Checks that any symbol used in a parameter value is properly defined. If an undefined symbol is encountered, message **KFJ00253E** is issued for each undefined symbol.
- Re-creates interim and work libraries.

Attention: If you previously used PARMGEN for configuration, be aware that using the **GENERATE** action with **OPTION VALIDATE** will re-create the WCONFIG data set, which will invalidate the PARMGEN configuration. This means that if you migrate a runtime environment from PARMGEN to Configuration Manager using the **MIGRATE** action, and then perform the **GENERATE** action with **OPTION VALIDATE**, you can no longer use the PARMGEN configuration.

QUICKLOAD

Using **OPTION QUICKLOAD**, you can load the read-only configuration members to the RK* data sets. The read-only members are those members that are not impacted by customization during configuration.

OPTION QUICKLOAD is compatible with the **USS** and **SECEXITS** options, but is not compatible with the **VALIDATE** and **NOUSS** options. This option requires a complete run of the **GENERATE** action prior to use.

QUICKCONFIG

With this option, you can quickly reconfigure your products without reloading the read-only runtime members or refreshing data from SMP/E target libraries. Using this option can improve performance, with the most impact experienced when using a full runtime environment (**RTE_TYPE** parameter is set to FULL).

When **OPTION QUICKCONFIG** is specified, the **GENERATE** action updates the configurable members for the runtime environment (for example, in the RKANPARU, RKANSAMU, and RKANCMU libraries) without refreshing data from SMP/E target libraries. The read-only members, which are members that are not impacted by customization during configuration, are not loaded to the RK* data sets.

More specifically, running the **GENERATE** action with **OPTION QUICKCONFIG** performs the following actions:

- Validates parameters
- Allocates missing data sets
- Updates configurable members for the runtime environment (for example, in the RKANPARU, RKANSAMU, and RKANCMU libraries)
- Updates started tasks, VTAM list, and VTAM node
- Updates z/OS® UNIX®-related members, but does not deploy them to the z/OS® UNIX® directories or files. To deploy prepared members to z/OS® UNIX® files or directories, you must combine the **QUICKCONFIG** option with the **USS** option.

Running the **GENERATE** action with **OPTION QUICKCONFIG** does not perform the following actions:

- Refresh data from SMP/E target libraries
- Update read-only modules from SMP/E target libraries
- Update z/OS® UNIX® files or directories
- Perform any assembly or linking operations related to security exits or other load modules

You cannot use the **QUICKCONFIG** option with the **VALIDATE** option or the **NOUSS** option. The **QUICKCONFIG** option requires a complete run of the **GENERATE** action prior to use.

RELINK | NORELINK

You can optionally control when to assemble and link elements into the RKANMODU load library when a relink of IBM Z OMEGAMON AI for Networks (KN3) or OMEGAMON enhanced 3270 user interface (KOB) modules is required, such as when applying maintenance. By default, the **GENERATE** action automatically performs these steps. By performing these tasks separately, you can isolate when additional authorization is needed, which might be the case when updating the load library.

You can use the **OPTION** parameter to specify the **RELINK** or **NORELINK** value to control this processing, as follows:

RELINK

Use **OPTION RELINK** to assemble and link elements into the RKANMODU load library when a relink of IBM Z OMEGAMON AI for Networks (KN3) or OMEGAMON Enhanced 3270 User Interface (KOB) modules is required, such as when applying maintenance. You cannot use this option on the initial run of the **GENERATE** action for a new runtime environment; it can only be used on subsequent runs.

Important: You must perform a complete run of the **GENERATE** action before you can use the **RELINK** option.

Note: If you are familiar with PARMGEN, the **RELINK** option performs the same function as the PARMGEN KCIJPLNK job (**Run post-SMP/E RKANMODU ASM/LINK steps**).

Tip: You can customize the binder program that is used by specifying the program name in parameter “**GBL_UTIL_BINDER**” on page 93.

When using the **GENERATE** action with **OPTION RELINK**, the details differ depending on the environment where the action runs, as follows:

On a local system (the configuration system in a remote deployment scenario)

You can run the **GENERATE** action with **OPTION RELINK** on a local system to assemble and link elements.

On a remote environment with a different z/OS level

You can run the **GENERATE** action with **OPTION RELINK** on a target system that is at a different z/OS level than the configuration system. You do not need to run all the steps in the remote deployment scenario (**GENERATE** action, **PACKAGE** action, transfer data sets, **DEPLOY** action).

Using the **RELINK** option requires the SMP/E library TKANMODS to be present on the target system. The TKANMODS library is not part of the full runtime environment, and it is not created by the **BLDREMDS** action. In other words, you must have an SMP/E environment present to ensure consistency. As a result, you must run Configuration Manager from the same SMP/E libraries.

You can rerun the **GENERATE** action with **OPTION RELINK** as needed.

When you use the **RELINK** option to relink modules on a remote environment at a different z/OS level, you might have different settings in the libraries set by parameters **GBL_DSN_SYS1_MACLIB** and **GBL_DSN_CEE_SCEELKED** than what you would have on your local system. If there are significant differences in the contents of those libraries, which control how modules are being assembled and linked, then you should run the **GENERATE** action with **OPTION RELINK** on the target system.

On a remote environment with the same z/OS level

You can run the **GENERATE** action with **OPTION RELINK** on a target system that is at the same z/OS level as the configuration system. In this environment, there is not an SMP/E environment on the target system, and only the minimal set of libraries is transferred to perform the **DEPLOY** action.

You can run the following series of steps:

1. **GENERATE** action with **OPTION RELINK**
2. **PACKAGE** action
3. Transfer of data sets
4. **DEPLOY** action

This implementation saves CPU cycles on the **GENERATE** step.

For a full runtime environment

In a full runtime environment, you must run the **GENERATE** action with **OPTION QUICKLOAD, RELINK** to achieve the correct relinking results. (This requirement is due to a dependency on the RKANSAM data set).

NORELINK

When **OPTION NORELINK** is specified, the **GENERATE** action does not assemble and link the IBM Z OMEGAMON AI for Networks (KN3) or OMEGAMON enhanced 3270 user interface (KOB) modules, even

if a relink is required. You can use this option on the initial run of the **GENERATE** action for a new runtime environment, as well as on subsequent runs.

Tip: This option does not update the RKANMODU load library, a data set that might require more restrictive authorization for updating. If you use this option together with option **NOSECEXITS**, the **GENERATE** action does not update the RKANMODU library, and you can perform the other **GENERATE** configuration steps that do not require additional access.

TRGCOPY

Use **OPTION TRGCOPY** to make a copy of your SMP/E target libraries, from which you can create or update your runtime environments. You can abbreviate this keyword to **TRG**. For more information, see [“Using SMP/E target library copies” on page 156](#).

PREPARE

Use **OPTION PREPARE** to prepare intermediate work libraries for your runtime environment. You can use this option to build the work data sets before other **GENERATE** action workflow stages, in preparation of OMEGAMON installation or maintenance. You might find this option useful if you have to perform other related tasks that require the work data sets ahead of time or in parallel, such as preparing members for Db2 binds or copying started task members to PROCLIB libraries.

With this option, you can run the **GENERATE** action with **OPTION PREPARE** initially to create the work data sets, and then, when you are ready, run the **GENERATE** option again (without the **PREPARE** option) to complete the additional workflow stages. You cannot use the **PREPARE** option with any other **OPTION** value.

Important: If you use the **PREPARE** option to create the work data sets, you must be at the same Configuration Manager maintenance level when you run subsequent **GENERATE** action jobs without the **PREPARE** option.

More specifically, running the **GENERATE** action with **OPTION PREPARE** performs the following actions:

- Validates parameter values
- Performs parsing steps
- Allocates and refreshes the interim staging (IK*) and work (WK*) libraries. These libraries will be copied to the runtime (RK*) libraries later.
- Allocates and refreshes PROCLIB, VTAMLST, and VTAMLIB libraries. These libraries contain started tasks, VTAM lists, and VTAM nodes.

Important: Data sets specified in parameters **GBL_DSN_SYS1_PROCLIB**, **GBL_DSN_SYS1_VTAMLIB**, and **GBL_DSN_SYS1_VTAMLST** will be impacted.

- Updates the global user library specified in parameter **GBL_USER_JCL**, which contains global parameters.

Important: Any changes to the following RTEDEF members will impact your existing runtime environment: **VAR\$GLOB**, **VAR\$lpar**, and **SYS@lpar**

- Creates a checkpoint. At the end of the job, the **GENERATE** action with **OPTION PREPARE** creates member \$PREPARE in the WKANPARU library and populates it with the checkpoint creation date and time, as follows:

```
***** Top of Data *****
* Checkpoint created by userID on yyyy-mm-dd hh:mm:ss
***** Bottom of Data *****
```

If the checkpoint is detected, subsequent **GENERATE** action jobs (without the **PREPARE** option) bypass the workflow stages performed by the **PREPARE** option and resume processing with the remaining workflow stages. The **GENERATE** action then deletes the \$PREPARE member at the end of the job.

Note: The checkpoint is not removed if any of the following options are used: **USS, SECEXITS, RELINK, QUICKLOAD**

Members excluded during GENERATE action

The Configuration Manager **GENERATE** action intentionally excludes some members when creating or refreshing the runtime (RK*) libraries. The output from the **GENERATE** action job includes a report that lists the members that were not replaced in the runtime libraries.

Exclude lists

To retain customized members in the runtime (RK*) user libraries during maintenance, you can use exclude lists to define the members not to replace.

IBM-provided exclude list definitions are stored in member KCI\$IW2R. User-defined exclude list definitions are stored in the embeds data set (EMBEDS) in member KCI\$XW2R. You must add any member that you want saved to your EMBEDS (KCI\$XW2R) member.

Note: For more information about using override embed members and the embeds data set, see [“Using override embed members in Configuration Manager” on page 140](#).

When you run the **GENERATE** action, members that are defined in the exclude lists are not replaced when the work (WK*) libraries are copied to the runtime (RK*) libraries.

After you run the **GENERATE** action, you can review a list of the members that were not replaced in the runtime (RK*) libraries. The list of members is provided in the exclude report, which is identified by the EXCLRPT DD statement.

Exclude report (EXCLRPT)

The exclude report (identified by the EXCLRPT DD statement) lists the members that were not replaced in the runtime (RK*) libraries by the **GENERATE** action job.

The exclude report is included in the job output for the **GENERATE** action only. If the **OPTION** parameter is used, the exclude report is generated only for options **NOUSS** and **QUICKCONFIG**.

Note: Both the **NOUSS** and **QUICKCONFIG** options update the configurable members for the runtime environment.

The following figure shows an example of the exclude report:

```

=====
EXCLUDE members not replaced in RK* libraries:
=====
Library: TSUSER.RTE.RKANPARU
-----
KAYDEL                                duplicate*
KAYOPEN                                duplicate*
KJJLSLST                               default
KC2GLB                                  default
KIPGLB99                               default
KDSENV                                  user

* consider removing exclude statements from KCI$XW2R user
exclude member
-----
Library: TSUSER.RTE.RKANSAMU
-----
CTKM000                                default
KC2GLB                                  default**
KC2GLBCP                                default
KC2GLBCR                                default
KC2GLBVR                                default
KC2GLBCR                                default
KC2GLBCV                                default
KC2GLBVR                                default
KC2GLB99                                default

** global definitions members are regenerated by the
configuration tool
-----
Library: TSUSER.RTE.RKD2PRF
-----
DB2PROF                                 user
-----
Library: TSUSER.RTE.RKD2SAM
-----
ALLOCD5                                 user

```

Figure 26: Example exclude report (EXCLRPT)

The following details describe the contents of the exclude report:

- The exclude report provides the status of excluded members for the following libraries:
 - RKANCMDU
 - RKANPARU
 - RKANSAMU
 - RKD2PAR
 - RKD2PRF
 - RKD2SAM

At least one excluded member must be found in the RK* data set for the respective section to be included in the report. If there are not any members excluded across all runtime libraries, the exclude report is not written.

- The exclude report provides the status for each excluded member as follows:

default

The member name is defined in the IBM-provided exclude list member KCI\$IW2R.

Note: Separate agent exclude rules are concatenated into the KCI\$IW2R member.

user

The member name is defined in the user-defined exclude list member EMBEDS (KCI\$XW2R).

duplicate

The member name is defined in both the IBM-provided exclude list member KCI\$IW2R and the user-defined exclude list member EMBEDS (KCI\$XW2R). The report uses an asterisk (*) to emphasize duplicate members and provide guidance to remove the duplicate definitions.

Tip: Because the member is defined in the default IBM-provided exclude list member KCI\$IW2R, you can remove the member name from your user-defined exclude list member EMBEDS (KCI\$XW2R).

- The exclude report includes the global definitions members for the following products:
 - IBM OMEGAMON for CICS ([Overview of the Global Data Area](#))
 - IBM OMEGAMON for IMS ([KIPGLB global definitions members](#))

The report uses two asterisks (**) to identify the global definitions members that are regenerated by Configuration Manager; these members are not affected by the exclude rules.

DELETE

Use the **DELETE** action to delete the runtime libraries for your runtime environment.

Before you begin

Review the following information before you use the **DELETE** action:

- The **DELETE** action requires the **RTE_NAME** and **RTE_PLIB_HILEV** parameters and values.
- If you are implementing a [remote deployment scenario](#) and used the **KFJ_LOCAL_PLIB_HILEV** parameter when creating the runtime environment, you can also specify this parameter for the **DELETE** action, depending on your situation. For more information, see [“Deleting libraries used for remote deployment” on page 59](#).
- Review details about the data sets that are deleted by the **DELETE** action and those that are not affected, as follows:

Data sets that are deleted by the DELETE action

The **DELETE** action deletes the runtime libraries created by the **GENERATE** action. This includes the PARMGEN libraries that the **GENERATE** action creates: WCONFIG, interim staging, work, and global user JCL.

The following list describes the name patterns of the data sets that are affected by the **DELETE** action:

- Data sets that match the following name patterns:
 - rte_plib_hilev.rte_name.**
 - rte_hilev.rte_name.**
 - rte_vsam_hilev.rte_name.**By default, these are all the same pattern, because the default value for *rte_hilev* and *rte_vsam_hilev* is *rte_plib_hilev*.
- For the following data sets, the default values for these parameters, when not specified explicitly, include *rte_name*, but custom values are allowed:
 - rte_pds_hilev.**
 - rte_pds2_hilev.**
- Data sets that match the following product-specific name patterns, if the respective products have been configured:
 - IBM Z® OMEGAMON® AI for Db2**
 - KD2_OMPE_DSHLQ.**
 - KD2_OMPE_VSAM_DSHLQ.**
 - IBM OMEGAMON for IMS on z/OS**
 - KI2_LOGR_EHLQ.KI2_LOGR_LS_PREFIX.**
- For remote deployment scenarios, if you used the **KFJ_LOCAL_PLIB_HILEV** parameter for deploying remote environments, data sets that match the following name patterns:
 - kfj_local_plib_hilev.rte_name.**
 - kfj_local_hilev.rte_name.**
 - kfj_local_vsam_hilev.rte_name.**
 - kfj_local_pds_hilev.**

Note: When deploying remote environments, the *rte_** parameters are for the remote runtime environments and the *kfj_local_** parameters are for the local runtime environments. On a single run of the **DELETE** action, either *rte_** parameter values or *kfj_local_** parameter values are used for the high-level qualifier.

Data sets not affected by the DELETE action

The **DELETE** action does not affect the following data sets:

- The runtime environment definition library: *rte_plib_hilev*.RTEDEF
- The security exits library, defined by **RTE_X_SECURITY_EXIT_LIB**
- The embed overrides library, defined by **RTE_X_OVERRIDE_EMBEDS_LIB**
- Any other data sets, such as persistent data store data sets, that have been allocated outside the *rte_name*-based data set name patterns described previously
- z/OS® UNIX® System Services directories

About this task

The following list provides details about the **DELETE** action:

- The **DELETE** action uses the specified information and values in the configuration members to derive the names of the data sets to be deleted.
- You must perform the **DELETE** action on the system where the files to be deleted exist, which is significant in remote deployment scenarios.
- The **DELETE** action requires the **CONFIRM** workflow variable to delete the data sets. You can also use the **CONFIRM** parameter to preview the data sets that will be deleted. Specify the **CONFIRM** parameter in the **KCIVARS DD** statement with one of the following values:

N

(Default) List all data sets that will be deleted. The list appears in **KCIPRINT**.

Important: It is recommended that you review the list of data sets that will be deleted before performing the delete.

Y

Delete the data sets. The **DELETE** action deletes your runtime libraries, as described in [“Data sets that are deleted by the DELETE action”](#) on page 57.

Procedure

Perform the following steps on the system where the files to be deleted exist:

1. Modify the **KFJJMCM** sample job in **TKANSAM** (see example below) to select a **DELETE** action.
2. Add the **CONFIRM** workflow variable and set to **N**, which will allow you to review the list of data sets that will be deleted.
3. Specify values for the required parameters **RTE_NAME** and **RTE_PLIB_HILEV**.
4. For remote deployment scenarios: If you are using remote deployment and want to delete data sets on the configuration system, specify the **KFJ_LOCAL_PLIB_HILEV** parameter and value.
5. Run the **KFJJMCM** job to display the data sets that will be deleted. The list appears in the **KCIPRINT SYSOUT** data set.
6. Review the generated list of data sets that will be deleted.
7. If you are satisfied with the list of data sets that will be deleted, change the **CONFIRM** workflow variable to **Y** and run the **KFJJMCM** job. The **DELETE** action deletes your runtime libraries.

The following JCL deletes the runtime libraries for the runtime environment that is defined in the member TSOUID.MONSUITE.RTEDEF (RTE1).

Figure 27: Example JCL to perform the **DELETE** action

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUITE.TKANMOD
//KCIFFLOW DD DISP=SHR,DSN=MONSUITE.TKANCUS(KFJOMEGA)
//KCIIVARS DD *
ACTION          DELETE
CONFIRM         Y
RTE_NAME       RTE1
RTE_PLIB_HILEV TSOUID.MONSUITE
/*
```

Note the JES2 **SYSAFF** that causes the job to run on LPAR ZOS1. This is required only if the **RTE_NAME** parameter in the TSOUID.MONSUITE.RTEDEF (RTE1) member refers to a variable whose value is LPAR-specific.

Deleting libraries used for remote deployment

You can use the **DELETE** action to delete libraries used for remote deployment.

The process of deploying a remote environment using the Configuration Manager **PACKAGE** and **DEPLOY** actions results in multiple sets of data sets on multiple systems. In a remote deployment scenario, you create a runtime environment on a specific LPAR (the configuration system), package the runtime environment data sets into transferable dump data sets on the configuration system, transfer the packaged data sets to the remote target system (target system), and deploy the packaged runtime environment data sets on the target system. As mentioned previously, many data sets are created; and these data sets can also have different high-level qualifiers. Deleting these data sets manually would be a complex task. Instead, you can delete these data sets easily and accurately using the **DELETE** action.

Using the **DELETE** action, you can remove these data sets on the configuration system or the target system, and you must perform the **DELETE** action on the system where the files to be deleted exist. If you used the **KFJ_LOCAL_PLIB_HILEV** parameter when creating the runtime environment, the **DELETE** action identifies and deletes the runtime environment that was generated for a remote system using different high-level qualifiers.

Important: It is recommended that you review the list of data sets that will be deleted by using the **CONFIRM** workflow variable set to N before performing the delete. After you are satisfied with the generated list, run the **DELETE** action using the **CONFIRM** workflow variable set to Y to delete the data sets.

The following examples describe which parameters to use with the **DELETE** action depending on your situation and assumes that you used **KFJ_LOCAL_PLIB_HILEV** when creating your runtime environment.

Note: For information about the data sets that are affected, see [“Data sets that are deleted by the DELETE action” on page 57](#).

Deleting data sets on the configuration system

To delete runtime environment data sets on the configuration system, run the **DELETE** action on the configuration system using the parameters in the following example. To indicate the high-level qualifier, note that both the **RTE_PLIB_HILEV** and **KFJ_LOCAL_PLIB_HILEV** parameters are specified.

```
//KCIIVARS DD *
ACTION          DELETE
CONFIRM         Y
RTE_NAME       RTE1
RTE_PLIB_HILEV TSOUID.TARG1
KFJ_LOCAL_PLIB_HILEV TSOUID.LOCL1
```

Figure 28: Example to delete data sets on the configuration system

This example will use the TSOUID.LOCL1.RTEDEF (PCK\$*) members to generate the list of data sets to delete on the configuration system.

Important: Make sure to specify the correct values for the RTE_PLIB_HILEV and KFJ_LOCAL_PLIB_HILEV parameters; otherwise, unexpected results might occur. It is important that you review the generated list of data sets to be deleted using the CONFIRM workflow variable set to N.

Deleting data sets on the remote system (target system)

To delete runtime environment data sets on the target system, run the **DELETE** action on the target system using the parameters in the following example. To indicate the high-level qualifier, note that only the RTE_PLIB_HILEV parameter is specified.

```
//KCIVARS DD *  
ACTION          DELETE  
CONFIRM         Y  
RTE_NAME       RTE1  
RTE_PLIB_HILEV TSOUID.TARG1
```

Figure 29: Example to delete data sets on the remote system (target system)

This example will delete the data sets for the runtime environment on the target system with high-level qualifier TSOUID.TARG1, or as defined in TSOUID.TARG1.RTEDEF (RTE1).

Important: Make sure to specify the correct value for the RTE_PLIB_HILEV parameter. If you inadvertently specify the value that was used for parameter KFJ_LOCAL_PLIB_HILEV for parameter RTE_PLIB_HILEV, unexpected results might occur. It is important that you review the generated list of data sets that will be deleted using the CONFIRM workflow variable set to N.

MIGRATE

The **MIGRATE** action imports configuration settings from a runtime environment that is configured with PARMGEN to one that is configured with Configuration Manager.

Before you begin

Review the following information before you use the **MIGRATE** action:

- If you migrate a runtime environment that is configured with PARMGEN to one that is configured with Configuration Manager, you can no longer use PARMGEN to configure the runtime environment. For more information, see [“Comparison with PARMGEN” on page 11](#).
- In this task, *source* refers to the runtime environment that is configured with PARMGEN, and *target* refers to the runtime environment that is configured with Configuration Manager.
- Migration works only for OMEGAMON products that are supported by Configuration Manager. If the migration source contains other products configured by PARMGEN that are not supported by Configuration Manager, error message [KFJ00001E](#) is issued for the **MIGRATE** action in the KCIPRINT output, and the job ends. For the list of supported products, see [“Products supported by Configuration Manager” on page 8](#).
- Consider the naming convention that you will use for your target runtime environment. Source and target runtime environments can share the same high-level qualifier, which is referred to as an *in-place migrate*.

Note: Whereas PARMGEN stores parameters in *rte_plib_hilev.rte_name.WCONFIG* for each runtime environment, Configuration Manager stores parameters and variables in *rte_plib_hilev.RTEDEF*, which can contain definitions for multiple runtime environments. For more information about the differences between PARMGEN and Configuration Manager, see [“Comparison with PARMGEN” on page 11](#).

- The **MIGRATE** action supports migrating one or multiple PARMGEN runtime environments into a single Configuration Manager RTEDEF configuration. It is recommended that you decide prior to the migration of your first runtime environment whether you plan to migrate one or multiple runtime environments into a single RTEDEF.

Note: If you are going to set up a High Availability TEMS (HA TEMS), make sure only one runtime environment is defined in the RTEDEF (that is, the one used for the HA TEMS).

- For parameters that describe data set qualifiers, the **MIGRATE** action does not migrate parameters that have been customized with hardcoded values that partially match the PARMGEN **RTE_HILEV** parameter value. For example, if in your PARMGEN configuration, you have set parameter **KD2_OMPE_DSHLQ** to use value **TEST.RTE1.HLQ1**, and **RTE_HILEV** is set to "TEST.RTE1", then **KD2_OMPE_DSHLQ** will not be migrated.
Before migrating from PARMGEN to Configuration Manager, review your customized parameters that describe data set qualifiers. If you have customized data set names that partially match your PARMGEN configuration **RTE_HILEV** value, but use a hardcoded value instead of parameter reference **%RTE_HILEV%**, you must update the value to use **%RTE_HILEV%** in order for the parameter to migrate successfully.
- Before performing a **MIGRATE** action, make sure you have a backup of your source PARMGEN runtime environment. The next step in Configuration Manager after a migration is to generate runtime members using the **GENERATE** action. If you perform an in-place migration, the subsequent **GENERATE** action will overwrite the runtime environment data sets that were used by PARMGEN.

About this task

With the **MIGRATE** action, you can import existing PARMGEN runtime environment configuration settings from a specific WCONFIG member into the Configuration Manager *rte_plib_hilev*.RTEDEF. The **MIGRATE** action reads the WCONFIG and other data sets from a PARMGEN installation, from which it creates the sparse descriptors containing the parameters, hiding every parameter setting that is considered a default or has not been changed. It also copies other files for system variables support, embed overrides, and security exits that are required to support the migration.

The following list provides details about the **MIGRATE** action:

- The **MIGRATE** action supports migrating one PARMGEN runtime environment at a time.
- You can migrate one or more PARMGEN runtime environments into a single Configuration Manager RTEDEF configuration. The default behavior of the **MIGRATE** action is to migrate only one runtime environment into the RTEDEF data set. Using the **OPTION MULTIPLE** parameter, you can migrate multiple runtime environments into a single RTEDEF data set. Each runtime environment migration requires a separate **MIGRATE** action job. If you plan to migrate multiple runtime environments into a single RTEDEF data set, make sure to include the **OPTION MULTIPLE** parameter on every **MIGRATE** action job, including the first one.

Note: You can abbreviate **OPTION MULTIPLE** to **OPTION MULTI**.

- The **MIGRATE** action creates the necessary members in the RTEDEF data set, as follows:
 - When using the default behavior of the **MIGRATE** action to migrate one runtime environment into a single RTEDEF data set (omitting the **OPTION MULTIPLE** parameter), the **MIGRATE** action will create members of type *Kpp\$PARM* in the respective created RTEDEF data set, along with the *rte_name* member for the runtime environment-specific parameters.
 - When migrating multiple runtime environments into a single RTEDEF configuration, use parameter **OPTION MULTIPLE** and **KFJ_SYSNAME 1par** in **KCIVARS DD**. The **MIGRATE** action will create members of type *Kpp\$lpar* in the RTEDEF data set, along with the *rte_name* member for the runtime environment-specific parameters.
On subsequent runs of the **MIGRATE** action, reuse the same **RTE_PLIB_HILEV** parameter value, but update the values for parameters **RTE_NAME**, **KFJ_MIGRATE_WCONFIG**, and **KFJ_SYSNAME** to create a new set of runtime environment parameter members. There is no limit on how many runtime environments can be migrated into a single RTEDEF data set.

- If the **MIGRATE** action detects that a specified target RTEDEF already contains *Kpp\$lpar* and *rte_name* members, **MIGRATE** issues an error message and stops. Note that **MIGRATE** will detect *Kpp\$PARM* and *VAR\$GLOB* members in RTEDEF as well. Because these members are considered to have a sysplex scope, they can only exist in RTEDEF during the migration process if no additional *Kpp\$lpar* members are intended to be migrated into the same RTEDEF. Depending on the case, any of the following messages might appear: [KFJ00218E](#), [KFJ00219E](#), [KFJ00220E](#)
- The **MIGRATE** action accepts PARMGEN runtime environments with system variables. However, system variables are not copied unless you have chosen to override them in your PARMGEN configuration. Variables are copied to the RTEDEF member *VAR\$GLOB* for a default (single) **MIGRATE** action or member *VAR\$lpar* in a multiple **MIGRATE** action.
- For parameters that use variables as their values, unlike PARMGEN, Configuration Manager does not allow the parameter and the variable to have the same name. If such settings exist in your PARMGEN configuration, the **MIGRATE** action renames the variables automatically. In most cases, the suffix *_R* is added to the variable name. For more information, see [“Variables migrated from PARMGEN” on page 138](#).
- The **MIGRATE** action allocates the security exits library with the default name *rte_plib_hilev.rte_name*.SECEXITS (or, optionally, the name specified in the **KFJ_SECURITY_EXITS_LIB** parameter). The **MIGRATE** action also copies the security exits used by the PARMGEN environment to the specified security exits library, and defines the source security exits library to the runtime environment using the **RTE_X_SECURITY_EXIT_LIB** parameter. For more information, see [“Setting up security exits in your runtime environment” on page 139](#).

Important: The **RTE_X_SECURITY_EXIT_LIB** parameter will contain the name of the security exits library used by the source PARMGEN environment; you must review this setting and update it if necessary before running the **GENERATE** action.

- If the use of override embed members is enabled by specifying parameter **KFJ_USE_EMBEDS** set to Y, the **MIGRATE** action allocates the embeds data set with the default name *rte_plib_hilev.rte_name*.EMBEDS (or, optionally, the name specified in the **KFJ_EMBEDS_LIB** parameter). The **MIGRATE** action sets up the embeds data set, populates it with supported override embed parameters (if applicable), and defines it to the runtime environment using the **RTE_X_OVERRIDE_EMBEDS_LIB** parameter. For more information, see [“Using override embed members in Configuration Manager” on page 140](#).
- The **MIGRATE** action works with the **KFJ_LOCAL_PLIB_HILEV** parameter to allow for local generation of runtime environments for remote systems using different high-level qualifiers. When the **KFJ_LOCAL_PLIB_HILEV** parameter is specified, the generated *kfj_local_plib_hilev*.RTEDEF data set will contain an additional member: *PCK\$PARM* for a default (single) **MIGRATE** action, or member *PCK\$lpar* in a multiple **MIGRATE** action. This member allows locally generated runtime environments using a different data set high-level qualifier than the one intended to be used on the deployment target (for example, the production system).

For more information about remote deployments, see [“Special considerations for SYSPLEX rollout” on page 124](#), [“RTEDEF\(PCK\\$PARM\)” on page 126](#), and [“Deploying remote runtime environments” on page 145](#).

After you run the MIGRATE action

After you run the **MIGRATE** action, you must carefully review the generated RTEDEF data set members to verify that the parameters have the expected values. You can use the report provided in the **MIGRATE** job output identified by the MIGRPT DD statement to review details about the parameters. This report presents parameters in the following groups:

- Parameters that are migrated to the RTEDEF data set because their values are different from the default values
- Parameters that are always migrated, regardless of values being default or not
- Parameters that are not migrated because of having default values
- Parameters that are not migrated because they match the PARMGEN **RTE_HILEV** parameter value

This report provides each parameter with its resolved value. Note that some parameters use system variables or are dependent on other parameters, and some parameters inherit values from other parameters and might be regarded as having default values. Additionally, some parameters have different

default values than PARMGEN, as outlined in “Parameters with different default values than PARMGEN” on page 91.

Important: Verify that the data set high-level qualifiers, data set names, and z/OS® UNIX® System Services paths are correct, as a subsequent **GENERATE** action might overwrite existing files.

To migrate a PARMGEN runtime environment to a Configuration Manager runtime environment, use the following procedure.

Note: The **CONFIRM** workflow variable is not supported for the **MIGRATE** action.

Procedure

1. Modify the KFJJMCM sample job in TKANSAM (see example below) to select a **MIGRATE** action.
2. Specify the required parameter values, as follows:
 - a. Specify values for the target environment that is to be configured with Configuration Manager in parameters **RTE_NAME** and **RTE_PLIB_HILEV**.
 - b. Specify the name of the source WCONFIG data set in the **KFJ_MIGRATE_WCONFIG** parameter. This is the WCONFIG data set of the PARMGEN-configured runtime environment from which configuration settings are to be imported.
3. (Optional) Specify additional parameters as needed, for example:
 - To migrate this runtime environment into a RTEDEF data set containing multiple runtime environment configurations, add the **OPTION MULTIPLE** and **KFJ_SYSNAME** *lpar* parameters. If this is a subsequent run of the **MIGRATE** action, reuse the same **RTE_PLIB_HILEV** parameter value, but update the values for parameters **RTE_NAME**, **KFJ_MIGRATE_WCONFIG**, and **KFJ_SYSNAME**.
 - To specify a different name for the security exits library, add the **KFJ_SECURITY_EXITS_LIB** parameter and value.
 - To enable the use of override embed members, add the **KFJ_USE_EMBEDS** parameter set to Y and the **KFJ_EMBEDS_LIB** parameter and value.
4. Run the KFJJMCM job to perform the migration and generate the new RTEDEF data set. Job messages for the **MIGRATE** action are written to the KCIPRINT SYSOUT data set.
5. Review the generated RTEDEF data set members to verify that the parameters have the expected values. Verify that the data set high-level qualifiers, data set names, and z/OS® UNIX® paths are correct. You can also use the report provided in the **MIGRATE** job output identified by the MIGRPT DD to review details about the parameters.

The following JCL jobs migrate an existing PARMGEN configuration pointed to by *highlevel.WCONFIG* into an RTEDEF library TSOUID.MONSUITE.RTEDEF. The first example is for a single runtime environment RTEDEF, and the second example is for a multiple runtime environment RTEDEF. These examples also specify that override embed members are enabled and provide custom data set names for the security exits and embeds libraries.

Figure 30: Example JCL to perform the **MIGRATE** action for a single runtime environment RTEDEF

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUITE.TKANMOD
//KCIFFLOW DD DISP=SHR,DSN=MONSUITE.TKANCUS(KFJOMEGA)
//KCIIVARS DD *
*
ACTION MIGRATE
RTE_NAME RTE1
RTE_PLIB_HILEV TSOUID.MONSUITE

KFJ_MIGRATE_WCONFIG highlevel.WCONFIG

KFJ_SECURITY_EXITS_LIB TEST1.TST.DEMO.MYEXITS
KFJ_USE_EMBEDS Y
```

```
KFJ_EMBEDS_LIB          TEST1.TST.DEMO.MYEMBEDS
/*
```

Figure 31: Example JCL to perform the **MIGRATE** action for a multiple runtime environment RTEDEF

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUIITE.TKANMOD
//KCIFFLOW DD DISP=SHR,DSN=MONSUIITE.TKANCUS(KFJOMEGA)
//KCIIVARS DD *
*
ACTION                MIGRATE
OPTION                MULTIPLE
RTE_NAME              RTE1
RTE_PLIB_HILEV       TSOUID.MONSUIITE

KFJ_MIGRATE_WCONFIG   highlevel.WCONFIG

KFJ_SECURITY_EXITS_LIB TEST1.TST.DEMO.MYEXITS
KFJ_USE_EMBEDS        Y
KFJ_EMBEDS_LIB        TEST1.TST.DEMO.MYEMBEDS

KFJ_SYSNAME           lpar
/*
```

What to do next

After you have finished migrating your PARMGEN runtime environments into the Configuration Manager RTEDEF library, use the **GENERATE** action to generate runtime members using the configured parameters. See [GENERATE](#).

PACKAGE

The **PACKAGE** action packages a runtime environment that can then be deployed to a remote system.

Before you begin

Review the following information before you use the **PACKAGE** action:

- You can run the **PACKAGE** action only after you have successfully generated a runtime environment using the **GENERATE** action. For more information about the complete remote deployment process, see [“Remote deployment scenario”](#) on page 145.
- The **PACKAGE** action uses the z/OS DFSMSdss **DUMP** command. You might need authority to run **ADDRDSSU**, which is the program that is invoked when using DFSMSdss. Specifically, the **PACKAGE** action uses the **DUMP** command keywords **CONCURRENT** and **TOLERATE(ENQFAILURE)**. The use of these keywords with the **DUMP** command is controlled by the following RACF FACILITY class profiles:

Table 9: RACF FACILITY class profile names for DFSMSdss keywords

Keyword	Profile Name
CONCURRENT with DUMP	STGADMIN.ADR.DUMP.CNCURRNT
TOLERATE(ENQF) with DUMP	STGADMIN.ADR.DUMP.TOLERATE.ENQF

Make sure that you have the proper RACF authority to use DFSMSdss and these keywords. If authorization is required, DFSMSdss issues error message [ADR707E](#).

For more information, see [z/OS DFSMSdss Storage Administration Guide: Protecting DFSMSdss functions with RACF FACILITY class profiles](#).

- Before you run the **PACKAGE** action, perform the following important steps:
 - For an existing runtime environment, stop the started tasks for that runtime environment. Started tasks can lock runtime members, which can cause the **PACKAGE** action to fail.
 - For the SMS-managed data sets for the runtime environment to be packaged, make sure that any migrated data sets are recalled.

Related topics

- [“Remote deployment scenario” on page 145](#)
- [“Remote deployment example using local libraries” on page 147](#)
- [“KFJ_LOCAL_PLIB_HILEV” on page 102](#)
- [“KFJ_PACK_HILEV” on page 107](#)
- [“DEPLOY” on page 67](#)

About this task

The following list provides details about the **PACKAGE** action:

- The **PACKAGE** action uses the z/OS DFSMSdss **DUMP** command to create a series of data sets that contain the runtime environment. These dump files can be transferred across SYSPLEX boundaries, using methods like FTPS. The **PACKAGE** action is used in conjunction with the **DEPLOY** action.
- The **PACKAGE** action requires the use of the KCIALPHA program. KCIALPHA is an APF-authorized version of KCIOMEGA.
- You must specify high-level qualifier values for locating existing data sets and for allocating the output package data sets. The following parameters are used for specifying high-level qualifiers:

RTE_PLIB_HILEV

(Required) Specifies the high-level qualifiers for the runtime environment definition (RTEDEF) library, as follows:

- If local libraries are not used (that is, parameter **KFJ_LOCAL_PLIB_HILEV** is not specified), **RTE_PLIB_HILEV** specifies the high-level qualifier of the RTEDEF library on the configuration system.
- If local libraries are used (that is, parameter **KFJ_LOCAL_PLIB_HILEV** is also specified), **RTE_PLIB_HILEV** specifies the high-level qualifier of the RTEDEF library on the remote (target) system.

You must specify the **RTE_PLIB_HILEV** parameter and value for the **PACKAGE** action.

KFJ_LOCAL_PLIB_HILEV

(Optional) Specifies the high-level qualifier for the local libraries. You must specify this parameter and value for the **PACKAGE** action if the parameter was used with the **CREATE** and **GENERATE** actions when building the runtime environment to be packaged. Otherwise, do not specify this parameter for the **PACKAGE** action. For more information on using local libraries for remote deployment, [“Remote deployment example using local libraries” on page 147](#).

Note: Parameter **KFJ_LOCAL_PLIB_HILEV** is used when packaging the runtime environment; it is not used when deploying the package.

KFJ_PACK_HILEV

(Optional) Specifies a custom high-level qualifier for the output package data sets, which can be useful to distinguish the package data sets from the many runtime environment data sets in the local libraries. If this parameter is not specified, the value for parameter **KFJ_LOCAL_PLIB_HILEV** (if specified) or **RTE_PLIB_HILEV** is used as the high-level qualifier for the output package data sets.

- The **PACKAGE** action creates the following package data sets:
 - A set of dump data sets using the following naming convention:

```
package_hlq.rte_name.PACKxx.DMP
```

where:

- *package_hlq* is one of the following values:
 - **KFJ_PACK_HILEV** parameter value, if specified
 - **KFJ_LOCAL_PLIB_HILEV** parameter value, if **KFJ_PACK_HILEV** is not specified

- **RTE_PLIB_HILEV** parameter value, if neither **KFJ_PACK_HILEV** or **KFJ_LOCAL_PLIB_HILEV** is specified

To avoid exceeding the z/OS 44-character limit for data set names, the combined length of *package_hlq* and *rte_name* should not exceed 28 characters. For example, if *rte_name* is 8 characters, then *package_hlq* should not exceed 20 characters.

- *xx* is one of the following package codes:

MN

Main non-VSAM package. Contains data sets from the **RTE_HILEV** libraries and some from the **RTE_PLIB_HILEV** libraries.

MV

Main VSAM package. Contains non-history related VSAM data sets from the **RTE_VSAM_HILEV** libraries.

HN

History non-VSAM package. Contains history-related data sets from the **RTE_PDS_HILEV** libraries.

HV

History VSAM package. Contains history-related data sets from the **RTE_VSAM_HILEV** libraries.

- A metadata file named *package_hlq.rte_name.PACKMD*. This flat file is not tersed or dumped and is required for the **DEPLOY** action if local libraries are used for the deployment.
- You can specify the following optional parameters in the **KCIVARS DD** statement when running the **PACKAGE** action:

KFJ_ADRDSSU_ADMIN

Specifies whether to use the **ADMINISTRATOR** keyword with z/OS DFSMSdss commands. Valid values are Y and N. The default is N. This option might be required depending on the security settings at your site.

KFJ_PACK_TERSE

Specifies if the dump data sets should also be tersed. Valid values are Y and N. The default is N. Specify Y if you want to transfer the packages using FTP. In this case, the dump files will be tersed and only the tersed copy will be retained.

Note: If you are using virtual tapes to transfer your data sets, tersing the dump files is not necessary.

If specified, the following parameters will be used for both the DMP and DMP . TRS data sets:

KFJ_PACK_DATACLAS

Specifies the data class for the package. This setting is needed for large package files or tersed files as it allows you to specify multiple volumes.

KFJ_PACK_MGMTCLAS

Specifies the management class for the package.

KFJ_PACK_STORCLAS

Specifies the storage class for the package.

KFJ_PACK_UNIT

Specifies the unit for the package if non-SMS or virtual tape is used.

KFJ_PACK_VOLUME

Specifies the volume for the package if non-SMS or virtual tape is used.

- You can use the **OPTION NOUSS** parameter to bypass the z/OS® UNIX® System Services workflow stage and not include any related files or directories in the produced output data sets.

To run the **PACKAGE** action, complete the following steps.

Procedure

1. Update the KFJJMCM sample job in TKANSAM as follows (see example below):
 - a. Select a **PACKAGE** action.
 - b. Change the program name in the JCL **EXEC** statement from KCIOMEGA to KCIALPHA.
 - c. Specify values for the required parameters **RTE_NAME** and **RTE_PLIB_HILEV**.
 - d. (Optional) Specify any additional parameters as needed.
2. For an existing runtime environment, stop the started tasks for that runtime environment.
3. Make sure that any migrated SMS-managed data sets to be packaged have been recalled.
4. Run the KFJJMCM job to perform the packaging process and generate the related package files. Job messages for the **PACKAGE** action are written to the KCIPRINT SYSOUT data set.

Result

The packaging dump data sets and the metadata file are created.

Example

The following JCL creates DUMP data sets for the runtime environment that is defined by members of the TSOUID.MONSUITE.RTEDEF library.

Figure 32: Example JCL to perform the PACKAGE action

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIALPHA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUITE.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=MONSUITE.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION                PACKAGE
RTE_NAME              RTE1
RTE_PLIB_HILEV       TSOUID.MONSUITE
/*
```

What to do next

Transfer the files to the target system using a method like FTPS, and run the **DEPLOY** action.

DEPLOY

The **DEPLOY** action deploys a packaged runtime environment to a remote system.

Before you begin

Review the following information before you use the **DEPLOY** action:

- You can run the **DEPLOY** action only after you have successfully packaged a runtime environment using the **PACKAGE** action.
- The **DEPLOY** action uses z/OS DFSMSdss commands. You might need authority to run **ADDRSSU**, which is the program that is invoked when using DFSMSdss. If authorization is required, DFSMSdss issues error message **ADR707E**. For more information, see *z/OS DFSMSdss Storage Administration Guide: Protecting DFSMSdss functions with RACF FACILITY class profiles*.
- If you do not have any SMP/E target libraries on the system on which you want to run a **DEPLOY** action, you can use the utility TKANSAM(KFJMAINT) with the **BLDREMDS** action to build the necessary minimum data sets (TKANSAM, TKANMOD, and TKANCUS libraries) needed to run the action. Make sure you transfer the created data sets to your remote system where **DEPLOY** should run and where the necessary APF authorization of the TKANMOD library is made. These SMP/E target libraries are the minimum required to allow Configuration Manager to run for a full runtime environment. However, for a runtime environment that you are sharing with SMP/E, you will need to copy your entire set of SMP/E target libraries to the system where you run the **DEPLOY** action.

Important: When you use the **DEPLOY** action, the target runtime environment data sets will be updated. Verify that these data sets are not in use before you use the **DEPLOY** action.

Important: The **PACKAGE** action creates a metadata file named *package_hlq.rte_name.PACKMD*. The **DEPLOY** action requires the PACKMD metadata file if a data set is renamed, which can occur if the **KFJ_LOCAL_*** parameters (in the KCIIVARS DD and in the PCK\$PARM member) are used when creating the package. If the PACKMD file is not available, the **DEPLOY** action restores packages as is, retaining all high-level qualifiers and SMS properties, and you might experience unpredictable results in your deployment. It is recommended that you use the PACKMD metadata file when using the **DEPLOY** action.

Related topics

- [“Remote deployment scenario” on page 145](#)
- [“Remote deployment example using local libraries” on page 147](#)
- [“PACKAGE” on page 64](#)
- [“KFJ_LOCAL_PLIB_HILEV” on page 102](#)
- [“KFJ_PACK_HILEV” on page 107](#)

About this task

The following list provides details about the **DEPLOY** action:

- The **DEPLOY** action deploys a runtime environment by restoring package data sets generated by the **PACKAGE** action on a target system. The **DEPLOY** action uses DFSMSdss to restore the data sets.
- The **DEPLOY** action requires the use of the KCIALPHA program. KCIALPHA is an APF-authorized version of KCIOMEGA.
- You must specify high-level qualifier values that are used for locating existing data sets. The following parameters are used for specifying high-level qualifiers:

RTE_PLIB_HILEV

(Required) Specifies the high-level qualifiers for locating the runtime environment definition (RTEDEF) library. You must specify this parameter and value for the **DEPLOY** action.

KFJ_PACK_HILEV

(Optional) Specifies the high-level qualifier value for the package data sets. If this parameter is not specified, the value for parameter **RTE_PLIB_HILEV** is used.

Important: You must use parameter **KFJ_PACK_HILEV** with the **DEPLOY** action if you transferred your package data sets with a high-level qualifier that is different from **RTE_PLIB_HILEV**.

Note: It is recommended that you specify the **KFJ_PACK_HILEV** parameter when deploying your runtime environment.

Note: Parameter **KFJ_LOCAL_PLIB_HILEV** is not supported on the **DEPLOY** action.

- You can optionally control when to run the **DEPLOY** workflow stage that deploys the parts related to z/OS® UNIX® System Services. By default, the **DEPLOY** action automatically deploys files and directories related to z/OS® UNIX®, if they are present in the packaged runtime environment data sets. However, there might be cases where you want to skip this step (for reasons such as authorization issues), and perform this step

at a later time. You can use the **OPTION** parameter to specify the **NOUSS** or **USS** value to control this processing, as follows:

NOUSS

When **OPTION NOUSS** is specified, the **DEPLOY** action does not run the stage that deploys the parts related to z/OS® UNIX®.

With this option, only z/OS data sets are deployed; files and directories related to z/OS® UNIX® are bypassed.

USS

When **OPTION USS** is specified, the **DEPLOY** action runs only the stage that deploys the parts related to z/OS® UNIX®; no other processing is performed.

This option is useful when you want to refresh files and directories related to z/OS® UNIX® only.

- The **DEPLOY** action performs an unconditional restore for the main VSAM and non-VSAM packages (fully replaces the data sets) and a conditional restore for the history packages (does not replace existing data sets). Note the following behaviors:
 - If the history dump data sets (**.PACKHN or **.PACKHV) are not found or failed to restore, a return code of 4 will be generated.
 - If the main dump data sets (**.PACKMN or **.PACKMV) are not found or failed to restore, a return code of 8 will be generated.

Note: The **DEPLOY** action replaces some main package VSAM and non-VSAM files, but does not replace any history-related files. Therefore, it is normal for PACKHN and PACKHV deploy flows to end with RC=8. If you want to avoid this, for example if you roll out maintenance, do not transfer these packages to the target system. While normally a return code of 8 would cause Configuration Manager to stop, in this particular situation (history files), a return code of 8 is considered acceptable and will not prevent Configuration Manager from continuing to function.

- When restoring (and potentially unterming) the packaged runtime environment, the **DEPLOY** action reuses the following parameters that were used with the **PACKAGE** action:
 - **KFJ_PACK_HILEV**
 - **KFJ_ADRDSSU_ADMIN**
 - **KFJ_PACK_TERSE**
 - **KFJ_PACK_DATACLAS**
 - **KFJ_PACK_UNIT**
 - **KFJ_PACK_VOLUME**

See “[PACKAGE](#)” on page 64 for information about these parameters.

- If **KFJ_PACK_TERSE** is set to Y, the **DEPLOY** action first unterms the package. The untermed DMP file high-level qualifier and SMS parameters are used as specified in the following parameters:
 - **RTE_PLIB_HILEV**
 - **RTE_SMS_MGMTCLAS**
 - **RTE_SMS_STORCLAS**
 - **RTE_SMS_UNIT**
 - **RTE_SMS_VOLUME**
- For large packages being untermed, you should use **KFJ_PACK_DATACLAS** accordingly to allow multi-volume data set allocation for the extracted package files.
- If **KFJ_PACK_UNIT** or **KFJ_PACK_VOLUME** is specified, it applies to all of the packages being untermed.
- If **RTE_SMS_VOLUME** is specified but **RTE_SMS_MGMTCLAS** is not, **RTE_SMS_MGMTCLAS** defaults to NULLMGMTCLAS. Similarly, **RTE_SMS_STORCLAS** defaults to NULLSTORCLAS.

To run the **DEPLOY** action, complete the following steps.

Procedure

1. Locate the dump (.DMP) data sets and the metadata file (.PACKMD) that were created by the **PACKAGE** action.
The naming convention of the package data sets is *package_hlq.rte_name.PACKMD*, where *package_hlq* is the value of parameter **KFJ_PACK_HILEV** or **RTE_PLIB_HILEV**.

Note: It is possible that the high-level qualifier value was modified when transferring the package data sets to the remote system.

2. Modify the KFJJMCM sample job in TKANSAM (see example below) to select a **DEPLOY** action.
3. Change the program name in the JCL **EXEC** statement from KCIOMEGA to KCIALPHA.
4. Specify values for the required parameters **RTE_NAME** and **RTE_PLIB_HILEV**.
5. (Optional) Specify any additional parameters as needed.

Important: You must use parameter **KFJ_PACK_HILEV** if you transferred your package data sets to the remote system with a high-level qualifier that is different from the **RTE_PLIB_HILEV** parameter value.

6. Run the KFJJMCM job to perform the deploy process and restore the related package files.
Job messages for the **DEPLOY** action are written to the KCIPRINT SYSOUT data set and to the \$REPORT DD. If return code 4 or 8 is received, review the \$REPORT DD statement in the JCL job output to ensure that the restore process completed successfully. For more information, see [“DEPLOY action output” on page 71](#).

The following examples restore (*deploy*) the data sets for a runtime environment that has been packaged by the **PACKAGE** action.

Using default high-level qualifier with parameter **RTE_PLIB_HILEV**

In this example, the package data sets were named using the default high-level qualifier of the runtime definition library and were transferred to the remote system using the same file names.

The packaged data sets contain the runtime environment named RTE1 and the runtime environment definition library TSOUID.MONSUITE.RTEDEF.

The packaged data sets are named TSOUID.MONSUITE.RTE1.PACK*.

The following **DEPLOY** action job, run on the remote system, deploys (restores) the package data sets on the target LPAR to data sets with the high-level qualifier TSOUID.MONSUITE.

Figure 33: Example JCL to perform the DEPLOY action using default high-level qualifier

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIALPHA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUITE.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=MONSUITE.TKANCUS(KFJOMEGA)
//KCIIVARS DD *
ACTION                DEPLOY
RTE_NAME              RTE1
RTE_PLIB_HILEV       TSOUID.MONSUITE
/*
```

Using custom high-level qualifier with parameter **KFJ_PACK_HILEV**

You must use parameter **KFJ_PACK_HILEV** with the **DEPLOY** action if you transferred your package data sets to the remote system with a high-level qualifier that is different from the **RTE_PLIB_HILEV** parameter value.

In this example, the package data sets contain the runtime environment named RTE1 and the runtime environment definition library TSOUID.MONSUITE.RTEDEF. However, the files were transferred to the remote system using the high-level qualifier TESTSYS.

As a result, the package data sets on the remote system are named TESTSYS.MONSUITE.RTE1.PACK*, requiring the use of parameter **KFJ_PACK_HILEV**.

Figure 34: Example JCL to perform the DEPLOY action using custom high-level qualifier

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIALPHA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUITE.TKANMOD
//KCIFFLOW DD DISP=SHR,DSN=MONSUITE.TKANCUS(KFJOMEGA)
//KCIIVARS DD *
ACTION          DEPLOY
RTE_NAME        RTE1
RTE_PLIB_HILEV  TSQUID.MONSUITE
KFJ_PACK_HILEV  TESTSYS.MONSUITE
/*
```

DEPLOY action output

The **DEPLOY** action produces job output that provides information about the deployment of the data sets as a result of the job.

When running in default mode, the **DEPLOY** action writes summarized output and the job return code to the **KCIPRINT** SYSOUT data set and produces a more detailed report in the **\$REPORT** DD. The following sections provide information about these output data sets and their contents, which includes [data set deployment states](#) and [DEPLOY action return codes](#).

KCIPRINT

KCIPRINT contains a count of the data sets that the **DEPLOY** action processes, summarized by [deployment state](#) (*deployed, skipped, failed*), for each package data set. Looking at KCIPRINT, you can see if any data sets failed, but you will not see the exact data set names. This content is a condensed preview of what is included in the **\$REPORT** output. KCIPRINT also contains the [return code](#) for the job.

The following figures show example KCIPRINT output for the **DEPLOY** action.

The example in “[Figure: DEPLOY action output in KCIPRINT – data set deployment summary](#)” on [page 71](#) shows sample summary counts of the data sets processed by the **DEPLOY** action for each package (dump) data set; the actual number of data sets reported will be unique to the job:

Figure 35: DEPLOY action output in KCIPRINT – data set deployment summary

```
IMB Z Monitoring Configuration manager is about to perform DEPLOY action

Processing: PACKMN
- Deployed: .. 30
- Skipped: ... 21
- Failed: .... 0

Processing: PACKHN
- Deployed: .. 0
- Skipped: ... 23
- Failed: .... 0

Processing: PACKMV
- Deployed: .. 38
- Skipped: ... 41
- Failed: .... 0

Processing: PACKHV
- Deployed: .. 0
- Skipped: ... 14
- Failed: .... 0

For more detailed information see $REPORT DD statement
```

For more information about the deployment states, see “[Data set deployment states](#)” on [page 72](#).

The example in “[Figure: DEPLOY action output in KCIPRINT – return code](#)” on [page 71](#) shows a possible return code for the **DEPLOY** action; the actual return code will be unique to the job:

Figure 36: DEPLOY action output in KCIPRINT – return code

```
KFU00004I KCIALPHA is ending; RC=4 SYSPLEX=sysplex LPAR=1par DATE=...
```

For more information about return codes for the **DEPLOY** action, see “[DEPLOY action return codes](#)” on page 73.

For more information about KCIPRINT, see “[Navigating Configuration Manager action output](#)” on page 171 and “[Action job output](#)” on page 74.

\$REPORT

\$REPORT provides details about which data sets were deployed, which were intentionally skipped, and which failed. In addition, the report includes the deployment status of the z/OS UNIX System Services (z/OS UNIX) data sets as a result of the job.

Note: If you encounter issues with the **DEPLOY** action and the normal run does not explain what failed, use the **DEPLOY** action with **OPTION DEBUG** to print the complete output.

The following figures shows example \$REPORT output.

The \$REPORT content lists the individual data sets that were skipped, deployed, and failed (if any) for each package (dump) data set. The report sample in “[Figure: DEPLOY action output in \\$REPORT – data set deployment state](#)” on page 72 shows a portion of the report for the PACKMN package data set only; the actual number of data sets reported will be unique to the job:

```
Processing data set: PACKMN
Will NOT REPLACE existing data sets
-----
Total deployed data sets: ..... 0
Total skipped data sets: ..... 21
Total failed to deploy data sets: .. 0

Details:
|- Skipped data sets:
||- rte_plib_hilev.rte_name.ims_id.RKOIPCSV
||- rte_plib_hilev.rte_name.ims_id.RKOIPFSV
||- rte_plib_hilev.rte_name.ims_id.RKOIPCSV
...
*****

Processing data set: PACKMN
Will REPLACE existing data sets
-----
Total deployed data sets: ..... 30
Total skipped data sets: ..... 0
Total failed to deploy data sets: .. 0

Details:
|- Deployed data sets:
||- rte_plib_hilev.rte_name.EMBEDS
||- rte_plib_hilev.rte_name.PGMSCN
||- rte_plib_hilev.rte_name.PGMSDS
...

```

Figure 37: DEPLOY action output in \$REPORT – data set deployment state

For more information about the deployment states, see “[Data set deployment states](#)” on page 72.

The report sample in “[Figure: DEPLOY action output in \\$REPORT – z/OS UNIX data sets](#)” on page 72 shows a portion of the report that lists the z/OS UNIX data sets that were deployed:

```
*****
z/OS UNIX System Services:

Extracting files to: /proj/tdci2/MS/MSDEPL/PGMS
-----
KS3
KS3/rdm
KS3/rdm/jobs
KS3/rdm/jobs/RDMUNPX
...

```

Figure 38: DEPLOY action output in \$REPORT – z/OS UNIX data sets

Data set deployment states

The **DEPLOY** action job output categorizes the processed data sets as follows:

Deployed

The **DEPLOY** action has successfully restored these data sets to the target system.

Skipped

The **DEPLOY** action skipped these data sets because they contain user data, such as configuration information and history-related files. These data sets should not be overwritten. It is normal to see numerous data sets skipped by the **DEPLOY** action.

Failed

The **DEPLOY** action did not restore these data sets to the target system. You should investigate any data sets that failed to deploy.

DEPLOY action return codes

The **DEPLOY** action uses the DFSMSdss ADRDSSU program to restore the data sets. It performs an unconditional restore for the main VSAM and non-VSAM packages (fully replaces the data sets) and a conditional restore for the history packages (does not replace existing data sets). Note the following behaviors:

- If history dump data sets (**.PACKHN or **.PACKHV) are not found or failed to restore, a return code of 4 will be generated.
- If the main dump data sets (**.PACKMN or **.PACKMV) are not found or failed to restore, a return code of 8 will be generated.

The **DEPLOY** action replaces all main package VSAM and non-VSAM files, but does not replace any history-related files. Therefore, it is normal for PACKHN and PACKHV deploy flows to end with RC=8. If you want to avoid this, for example if you roll out maintenance, do not transfer these packages to the target system. While normally a return code of 8 would cause Configuration Manager to stop, in this particular situation (history files), a return code of 8 is considered acceptable and will not prevent Configuration Manager from continuing to function.

KCIOMEGA workflows

The KCIOMEGA program that runs IBM Z® Monitoring Configuration Manager is a general-purpose job template engine. KCIOMEGA performs batch processing based on the job template that you specify. The processing that KCIOMEGA performs is known as a *workflow*.

Note: KCIALPHA is an APF-authorized version of KCIOMEGA. APF authorization is required for some actions.

In KCIOMEGA terms, Monitoring Configuration Manager is a workflow.

The KCIOMEGA program has two input data sets:

KCIFLOW

Contains a job template written in the KCIOMEGA *skeleton language*. The language is similar to a subset of JCL with additional syntax introduced by KCIOMEGA.

This job template is also known as a *workflow skeleton* or simply *skeleton*.

KCIOMEGA dynamically interprets the statements in the skeleton and performs the corresponding processing (the workflow).

Skeletons can invoke other skeletons, resulting in composite workflows that run sub-workflows.

The KCIOMEGA skeleton language is unpublished; not intended for users. However, it's human-readable plain text. If you're familiar with JCL syntax, the additional KCIOMEGA syntax is relatively straightforward to understand.

KCIVARS

Contains name-value pairs that set workflow variables.

KCIOMEGA replaces variable names in skeletons with the values from this data set.

Skeletons can refer to variable names in various contexts, such as data set names and “if ... then” conditions. Variables can determine the actions that workflows perform and the data sets that workflows use.

In the context of Monitoring Configuration Manager, workflow variables specify which action to perform and the location of the runtime environment definition on which you want to perform that action.

Action job output

Review the types of output data sets that are produced by the **KCIOMEGA** program.

Configuration Manager writes job output for the **KCIOMEGA** actions to a number of output data sets. Output from invoked utilities is also generated, with each utility writing to its own output data set.

Note: **KCIALPHA** is an APF-authorized version of the **KCIOMEGA** program. The information in this topic also applies when using **KCIALPHA**.

The following table lists the output data sets that Configuration Manager generates and retains.

Output data set	Description
<u>KCIPRINT</u>	KCIOMEGA program messages and messages about Configuration Manager processing
<u>KCITRACE</u>	KCIOMEGA workflow trace output, used primarily by IBM Software Support for troubleshooting
KCIVARSO	KCIVARS input that was used in the JCL
DSNPROUT	Contents of the RTEDEF library used in the configuration. When the DEBUG action option is used, this data set contains the complete list of members in the RTEDEF library. <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p>Note: The DEBUG action option should only be used under the guidance of IBM Software Support.</p> </div>
<u>\$VALRPT</u>	Parameter validation report
<u>MIGRPT</u>	Output for the MIGRATE action
<u>EXCLRPT</u>	List of members excluded (not replaced) in the runtime libraries during the GENERATE action
<u>REPORT</u>	Products installed report. This report appears automatically in the job output for the CREATE , MIGRATE , and GENERATE actions and can also be generated using the KFJMAINT maintenance workflow by specifying ACTION REPORT .
<u>\$REPORT</u>	Output for the DEPLOY action
JESMSG LG, JESJCL and JESYSMSG	Standard JES-produced output

In addition to the output data sets listed in “Standard sysout data sets” on page 74, Configuration Manager also generates other output that it does not retain and deletes from the spool automatically.

If a Configuration Manager job fails, the DD statement that contains the error message is retained and printed.

Notes:

- JES3 does not allow deleting DD output. If your site uses JES3, you might see additional output, which can be considerable.

- With APAR OA65222, the **DEBUG** action option (**OPTION DEBUG**), is introduced. This option generates additional output that is otherwise suppressed by default. This option should only be used under the guidance of IBM Software Support. Prior to this enhancement, all generated output was retained and printed to the spool, which could result in hundreds of output data sets.

To review the Configuration Manager job output, use a tool such as SDSF to view the JES output data sets.

KCIPRINT sysout data set

The KCIPRINT output data set contains messages from the KCIOMEGA program interspersed with messages about Configuration Manager processing.

Here is an example KCIPRINT for a successful job:

Figure 39: Example KCIPRINT output data set for a successful Configuration Manager job

```
KFU00001I KCIOMEGA is starting; SYSPLEX=sysplex LPAR=lpar DATE=...
KFU00002I INVOKE processing is about to commence; MEMBER=KFJOMEGA
...
Workflow has completed successfully
KFU00004I KCIOMEGA is ending; RC=rc SYSPLEX=sysplex LPAR=lpar DATE=...
```

Parameter values used

The KCIPRINT sysout data set from a **GENERATE** action contains an ordered list of the RTEDEF library members that the action uses. For example:

```
01. Using parameters in rte_plib_hilev.RTEDEF(rte_name)
02. Using parameters in rte_plib_hilev.RTEDEF(KDS$PARM)
03. Using parameters in rte_plib_hilev.RTEDEF(GBL$PARM)
```

The corresponding KCITRACE sysout data set contains an alphabetical list of parameters and symbols (such as workflow variables) with their values. The list is preceded by the following heading:

```
>Parameter and symbol values
```

Note to users with PARMGEN experience: The **GENERATE** action of Configuration Manager creates an *rte_plib_hilev.rte_name.WCONFIG(rte_name)* member that is similar to the member created by PARMGEN, with one key difference: the member created by Configuration Manager contains default parameter values; it does not reflect the values in your RTEDEF library members. To see the parameter values used by Configuration Manager, refer to the `>Parameter and symbol values` heading in the KCITRACE sysout data set. As such, the WCONFIG data sets as a whole should be considered as a *black box*.

KCITRACE sysout data set

The KCITRACE output data set contains the KCIPRINT contents, and, in addition, also includes the source of each workflow skeleton and additional messages.

KCITRACE records that start with two consecutive plus signs (++) show the previous record after variable substitution. For example:

```
//RTEDEF DD DSN=%RTEDEF_DSN%,PASS=YES
++RTEDEF DD DSN=MYID.MONITORS.RTEDEF,PASS=YES
```

Renamed SYSPRINT sysout data sets

The KCIOMEGA program runs a workflow, such as the **GENERATE** action of Configuration Manager, that can invoke many programs, resulting in long job output listings. All of these programs run in the same job step as KCIOMEGA.

Many programs write to the ddname SYSPRINT. To avoid a job output listing with multiple SYSPRINT ddnames for the same step name, KCIOMEGA renames SYSPRINT sysout data sets to match the corresponding step

name in the workflow skeleton shown in KCITRACE. This makes it easier to find the SYSPRINT for each step in the skeleton.

Parameter validation report (\$VALRPT)

If the set of parameters that you specified in the RTEDEF library is invalid (for example, a required parameter is missing or a defined value is incorrect), then the sysout data sets include the parameter validation report (\$VALRPT). If this report is generated, then review the report, correct the parameters, and resubmit the job.

If you are familiar with PARMGEN: The Configuration Manager parameter validation report contains the same information as the parameter validation report generated by PARMGEN, with the difference that you do not have to manually navigate to the data set and open it.

Products installed report (REPORT)

The *products installed report* provides information about the installation status and Configuration Manager support of products at your site. The report identifies the following products:

- Products that are installed and supported by Configuration Manager
- Products that are installed but not supported by Configuration Manager
- Products that are not installed but supported by Configuration Manager

The report appears automatically in the job output for the **CREATE**, **MIGRATE**, and **GENERATE** actions. You can also generate the report using the KFJMAINT maintenance workflow by specifying **ACTION REPORT**. (No other option or parameter is required to run the KFJMAINT job with the **REPORT** action.)

Tip: By producing the report using KFJMAINT, you can verify that your product is supported by Configuration Manager before beginning the configuration process. For a complete list of supported products, see “Products supported by Configuration Manager” on page 8.

The report is printed to the job output identified by the REPORT DD statement.

The report lists the product code, the product version, and the product name. If the version is not supported in Configuration Manager, then an asterisk appears after the version and the comment (not supported) appears after the name. If the product is not installed, then N/A appears as the version and the comment (not installed) appears after the name.

The following figure shows an example of the products installed report:

```

+-----+
| Products installed in <data_set_name> |
+-----+
| Code | Version | Product name |
+-----+
| KM5   | N/A     | IBM Z OMEGAMON AI for z/OS (not installed) |
| KN3   | 5.1.0*  | IBM OMEGAMON for Networks on z/OS (not supported) |
| KWO   | 5.6.0   | IBM OMEGAMON Dashboard Edition on z/OS |
| KS3   | 6.1.0   | IBM Z OMEGAMON AI for Storage |
| KC5   | 6.1.0   | IBM Z OMEGAMON AI for CICS |
| KGW   | 6.1.0   | IBM Z OMEGAMON AI for CICS Transaction Gateway |
| KI5   | 5.5.0   | IBM OMEGAMON for IMS on z/OS |
| KJJ   | 6.1.0   | IBM Z OMEGAMON AI for JVM |
| KD5   | 6.1.0   | IBM Z OMEGAMON AI for Db2 |
| KQI   | 7.5.0   | IBM OMEGAMON for Messaging Integration Bus |
| KMQ   | 7.5.0   | IBM OMEGAMON for Messaging on z/OS |
| KYN   | 7.1.1   | ITCAM for Application Diagnostics Agent |
| KDS   | 6.3.0   | Tivoli Enterprise Monitoring Server |
| KOB   | 7.5.0   | OMEGAMON Enhanced 3270 User Interface |
| KNA   | 4.4.0*  | IBM Z NetView Enterprise Management Agent (not supported) |
| KRN   | 2.6.0   | IBM Tivoli Advanced Catalog Management for z/OS |
| KRH   | 2.6.0   | IBM Tivoli Advanced Reporting and Management for KFSMSshm (not supported) |
| KRG   | 2.6.0   | IBM Tivoli Advanced Audit for DFSMSshm |
| KRJ   | 3.3.0   | IBM Tivoli Advanced Allocation Management for z/OS |
| KRK   | 3.3.0   | IBM Tivoli Automated Tape Allocation Manager for z/OS |
| KRV   | 2.4.0   | IBM Tivoli Advanced Backup and Recovery for z/OS |
| KAY   | 1.1.0   | IBM Z OMEGAMON Data Provider |
+-----+

```

Figure 40: Products installed report (REPORT)

Parameters

In general, IBM Z® Monitoring Configuration Manager and PARMGEN use the same parameters with the same default values. In a few cases, Monitoring Configuration Manager introduces a new parameter or sets a different default value for an existing parameter.

For parameters not described here, see [Where to find information](#).

Parameters that determine what products are configured

You can control the products that are configured in your runtime environment using the **CONFIGURE_*** parameters.

Each product that is configured in the runtime environment has a corresponding parameter with the prefix **CONFIGURE**. The **CONFIGURE_*** parameters are defined in the *rte_name* member in the runtime environment definition library (RTEDEF) and have the permissible values of Y and N.

For example, for a runtime environment that contains a monitoring server, an OMEGAMON for z/OS monitoring agent, the OMEGAMON enhanced 3270 user interface, and OMEGAMON Data Provider, each of the corresponding parameters are set to Y; the parameters for the other products are set to N:

CONFIGURE_TEMS_KDS	Y * TEMS
CONFIGURE_E3270UI_KOB	Y * Enhanced 3270
CONFIGURE_CICS_KC5	N * CICS TS
CONFIGURE_CICS_TG_KGW	N * CICS TG
CONFIGURE_DB2_AGENT_KD5	N * Db2
CONFIGURE_IMS_KI5	N * IMS
CONFIGURE_JVM_KJJ	N * JVM
CONFIGURE_ZOS_KM5	Y * z/OS
CONFIGURE_MESSAGING_KMQ	N * MQ
CONFIGURE_MESSAGING_KQI	N * Integration Bus
CONFIGURE_NETVIEW_KNA	N * Netview
CONFIGURE_MFN_KN3	N * Network
CONFIGURE_STORAGE_KS3	N * Storage
CONFIGURE_OMEGAVIEW_KWO	N * Integration Monitor
CONFIGURE_ITCAMAD_KYN	N * ITCAM for Applications
CONFIGURE_ACM_KRN	N * Advanced Catalog Mgmt
CONFIGURE_ARD_KRH	N * Advanced Rpt and Mgmt
CONFIGURE_AAD_KRG	N * Advanced Audit
CONFIGURE_AAM_KRJ	N * Advanced Alloc Mgmt
CONFIGURE_ATAM_KRK	N * Automated Tape Alloc
CONFIGURE_ABR_KRV	N * Advanced Backup and Rec
CONFIGURE_ODP_KAY	Y * Data Provider

Tip: Explicitly setting the **CONFIGURE_*** parameters to N for the products that are not used can result in less CPU and time used as the respective product-specific workflows are bypassed. Otherwise, unspecified configuration flags default to N as early as possible in the workflow.

The following table lists the products and corresponding **CONFIGURE** parameters.

<i>Table 11: CONFIGURE_* parameters</i>	
Product	Parameter
Tivoli Enterprise Monitoring Server	CONFIGURE_TEMS_KDS
OMEGAMON® Enhanced 3270 User Interface	CONFIGURE_E3270UI_KOB
IBM Z OMEGAMON AI for CICS TS	CONFIGURE_CICS_KC5
IBM Z OMEGAMON AI for CICS TG	CONFIGURE_CICS_TG_KGW
IBM Z® OMEGAMON® AI for Db2	CONFIGURE_DB2_AGENT_KD5
IBM OMEGAMON® for IMS on z/OS	CONFIGURE_IMS_KI5
IBM Z® OMEGAMON® AI for JVM	CONFIGURE_JVM_KJJ
IBM Z® OMEGAMON® AI for z/OS®	CONFIGURE_ZOS_KM5

Product	Parameter
IBM OMEGAMON® for Messaging on z/OS - MQ	CONFIGURE_MESSAGING_KMQ
IBM OMEGAMON® for Messaging on z/OS - Integration Bus	CONFIGURE_MESSAGING_KQI
IBM Z® NetView Enterprise Management Agent	CONFIGURE_NETVIEW_KNA
IBM Z® OMEGAMON® AI for Networks	CONFIGURE_MFN_KN3
IBM Z OMEGAMON® AI for Storage	CONFIGURE_STORAGE_KS3
IBM Z® OMEGAMON® Integration Monitor	CONFIGURE_OMEGAVIEW_KWO
IBM Tivoli Composite Application Manager (ITCAM) for Application Diagnostics Agent	CONFIGURE_ITCAMAD_KYN
IBM Tivoli Advanced Catalog Management for z/OS	CONFIGURE_ACM_KRN
IBM Tivoli Advanced Reporting and Management for DFSMSHsm	CONFIGURE_ARD_KRH
IBM Tivoli Advanced Audit for DFSMSHsm	CONFIGURE_AAD_KRG
IBM Tivoli Advanced Allocation Management for z/OS	CONFIGURE_AAM_KRJ
IBM Tivoli Automated Tape Allocation Manager for z/OS	CONFIGURE_ATAM_KRK
IBM Tivoli Advanced Backup and Recovery for z/OS	CONFIGURE_ABR_KRV
IBM Z® OMEGAMON Data Provider	CONFIGURE_ODP_KAY

Related topic:

[“How to: Add a new agent or product to a runtime environment” on page 161](#)

[“How to: Add OMEGAMON Data Provider to a runtime environment” on page 162](#)

Parameters in the initial runtime environment configuration profile

The **CREATE** action creates an initial set of parameters that define a basic runtime environment. You can edit and add to this initial set to meet your specific requirements.

These parameters are also described in the OMEGAMON® shared documentation or in the documentation for each product. The descriptions provided here include additional information to help you get started.

GBL_DSN_CICS_CTG_DLL

The CICS® Transaction Gateway (TG) dynamic link library.

Description

The CICS® Transaction Gateway (TG) dynamic link library.

Default value

SYS1.SCTGDLL

Permissible values

An MVS™ data set name.

GBL_DSN_CSF_SCSFMODE

This parameter contains the Integrated Cryptographic Service Facility (ICSF) load library that contains the CSNB* modules used for password encryption.

Description

This parameter is valid in both Configuration Manager and PARMGEN if ICSF is installed and configured on the z/OS® system.

In Configuration Manager:

This parameter is explicitly added in the GBL\$PARM member that is generated after the **CREATE** or **MIGRATE** action, as it is relevant for several security-related aspects of the product configuration (such as password encryption).

If your installation does not use the ICSF, you can remove or comment out this parameter in your RTEDEF (GBL\$PARM) or RTEDEF (GBL\$*lpar*).

In PARMGEN:

By default, this parameter is commented out in the WCONFIG (\$GBL\$USR) member.

Required or optional

This parameter is required if any of the following conditions are in effect:

- Password encryption is enabled for any components.
- A SOAP server is enabled on a remote Tivoli® Enterprise Monitoring Server.
- Granular control of command requests is enabled (compatibility mode is *disabled*): the **KDS_KMS_SECURITY_COMPATMD** parameter is set to N.
- zAware feature is enabled for OMEGAMON® on z/OS®.

Default value

CSF.SCSFMODE0

Permissible values

An MVS™ data set name, maximum length 44

Related parameters

- RTE_SECURITY_KAES256_KEY
- KDS_KMS_SECURITY_COMPATMD

GBL_DSN_DB2_DSNEXT

The Db2® exit library.

Description

The Db2® exit library.

The OMEGAMON® collector uses the Db2® exit load modules in this library.

Example value:

```
DSN.VC10.SDSNEXT
```

Permissible values

An MVS™ data set name.

GBL_DSN_DB2_LOADLIB_Vn

The load library for the version of Db2® that your site is running.

Description

The load library for the version of Db2® that your site is running.

In the parameter name, **n** is the Db2® version number. For example, **GBL_DSN_DB2_LOADLIB_V12**.

Specify a **GBL_DSN_DB2_LOADLIB_Vn** parameter for each Db2® version that you want to monitor.

This parameter is required if the runtime environment contains the Db2® monitoring agent.

Example value:

```
DSN.VC10.SDSNLOAD
```

Permissible values

An MVS™ data set name.

GBL_DSN_DB2_RUNLIB_Vn

The run library for the version of Db2® that your site is running.

Description

The run library for the version of Db2® that your site is running.

In the parameter name, **n** is the Db2® version number. For example, **GBL_DSN_DB2_RUNLIB_V12**.

Specify a **GBL_DSN_DB2_RUNLIB_Vn** parameter for each Db2® version that you want to monitor. The library should contain the modules DSNTIAD and DSNTIAUL to be used to run in batch.

IBM Z® Monitoring Configuration Manager uses the library to generate **GRANT** and **BIND** jobs that prepare the Db2® subsystems for monitoring.

This parameter is required if the runtime environment contains the Db2® monitoring agent.

Example value:

```
DSN.VC10.RUNLIB.LOAD
```

Permissible values

An MVS™ data set name.

GBL_DSN_IMS_RESLIB

The IMS SDFSRESL library.

Description

The IMS SDFSRESL library contains the CQSREG00 action module required to enable the Common Queue Server (CQS). The CQS and shared queues allow users to take advantage of the Parallel Sysplex® environment.

Note: The DISCOVER action of IBM Z® Monitoring Configuration Manager discovers the value of the KI2_I1nn_CLASSIC_IMS_RESLIB parameter, which also specifies an IMS SDFSRESL library. Depending on how IMS is configured at your site, the same value might be appropriate for GBL_DSN_IMS_RESLIB.

Required or optional

Required if the runtime environment configures the IMS monitoring agent.

Default value

IMS.SDFSRESL

GBL_DSN_IMS_SCEXLINK

The IMS Connect product load library.

Description

The IMS Connect product load library.

The IMS monitoring agent uses the IMS Connect Extensions Publisher API. The agent requires the IMS Connect Extensions product and functional support load libraries to connect to and collect performance and statistics data from the IMS Connect address space.

This parameter is required if the runtime environment configures the IMS monitoring agent.

Default value

IMS.SCEXLINK

Permissible values

An MVS™ data set name.

GBL_DSN_IMS_SFUNLINK

The IMS Connect functional support load library.

Description

The IMS Connect functional support load library.

The IMS monitoring agent uses the IMS Connect Extensions Publisher API. The agent requires the IMS Connect Extensions product and functional support load libraries to connect to and collect performance and statistics data from the IMS Connect address space.

This parameter is required if the runtime environment configures the IMS monitoring agent.

Default value

IMS.SFUNLINK

Permissible values

An MVS™ data set name.

GBL_DSN_WMQ_SCSQANLE

The IBM® MQ language library.

Description

The IBM® MQ language library.

This parameter is required if the runtime environment configures the MQ monitoring agent.

Default value

CSQ.V9R0M0.SCSQANLE

Permissible values

An MVS™ data set name.

GBL_DSN_WMQ_SCSQAUTH

The IBM® MQ authorized load library.

Description

The IBM® MQ authorized load library.

This parameter is required if the runtime environment configures the MQ monitoring agent.

Default value

CSQ.V9R0M0.SCSQAUTH

Permissible values

An MVS™ data set name.

GBL_DSN_NETVIEW_CNMLINK

Identifies the IBM Z® NetView CNMLINK library

Description

This parameter identifies the IBM Z® NetView CNMLINK library .

It is required only if using an IBM Z® NetView Agent.

Default value

NETVIEW.VNRNMN.CNMLINK

Permissible values

An MVS™ data set name.

GBL_HFS_JAVA_DIRn

The z/OS® UNIX® System Services path of the Java™ home directory.

Description

The z/OS® UNIX® System Services path of the Java™ home directory.

The path consists of the concatenated values of the **GBL_HFS_JAVA_DIR1** and **GBL_HFS_JAVA_DIR2** parameters.

Typically, you only specify **GBL_HFS_JAVA_DIR1**. **GBL_HFS_JAVA_DIR2** is provided as a convenience to specify the remainder of a long directory path.

This parameter is required if you are enabling the self-describing agent (SDA) functionality in the z/OS® monitoring server (TEMS) and agents.

For the runtime members, see the **TEMS_JAVA_BINPATH** parameter in the KSDSPROF member of the RKANDATV library.

As an example, if **GBL_HFS_JAVA_DIR1** is set to `/my/own/copy` and **GBL_HFS_JAVA_DIR2** is set to `/of/java`, then the resulting directory path is a concatenation of these two values: `/my/own/copy/of/java`.

Default value

For **GBL_HFS_JAVA_DIR1**: `/usr/lpp/java/IBM/J8.0_64`

For **GBL_HFS_JAVA_DIR2**: *none* (empty string)

Permissible values

A z/OS® UNIX® directory path. Must begin with a forward slash (/).

Do not specify a trailing `/bin` directory in the value. IBM Z® Monitoring Configuration Manager appends `/bin` to the value that you specify.

GBL_TARGET_HILEV

The high-level qualifiers of the target libraries, such as TKANDATV and TKANMOD.

Description

The high-level qualifiers of the target libraries, such as TKANDATV and TKANMOD.

The **CREATE** action of IBM Z® Monitoring Configuration Manager sets the **GBL_TARGET_HILEV** parameter to the high-level qualifiers of the data set name specified by the KCIFLOW DD statement in the job step that performs the **CREATE** action.

This parameter is required.

As an example, if the KCIFLOW DD statement of the job step that performs the **CREATE** action specifies the data set name MONSUITE . TKANCUS, then the **CREATE** action sets the value of **GBL_TARGET_HILEV** to MONSUITE.

Permissible values

MVS™ data set high-level qualifiers.

GBL_USS_TKANJAR_PATH

The path of the z/OS® UNIX® System Services directory that the SMP/E installation jobs define using the ddname TKANJAR.

Depending on your local site practices, this path might refer to a copy, rather than the original SMP/E-managed directory.

Required or optional

Required if the runtime environment configures either of the following monitoring agents:

- CICS® Transaction Gateway (TG). The corresponding configuration parameter is **CONFIGURE_CICS_TG_KGW**.
- Java™ Virtual Machine (JVM). The corresponding configuration parameter is **CONFIGURE_JVM_KJJ**.

Default value

/usr/lpp/kan/bin/IBM

Permissible values

z/OS® UNIX® directory path. Must begin with a forward slash (/).

Related parameters

GBL_DSN_SYS1_SBPXEXEC

RTE_USS_RTEDIR

RTE_USS_MKDIR_MODE

GBL_USS_TKAYHFS_PATH

This parameter specifies the SMP/E TKAYHFS ddname installation directory.

Description

This parameter specifies the path of the z/OS UNIX directory that the SMP/E installation jobs for IBM Z OMEGAMON Data Provider define using the ddname TKAYHFS.

Depending on your local site practices, this path might refer to a copy of the SMP/E-managed directory rather than the original.

This directory is read-only for Monitoring Configuration Manager; Monitoring Configuration Manager does not write to this directory.

Default value

/usr/lpp/omdp

Permissible values

Character string, maximum length 44

RTE_COMM_PROTOCOLn

This parameter sets the communication protocol choices of all components in the runtime environment.

Note: This parameter is valid in Configuration Manager only. It is not valid in PARMGEN.

Description

The **RTE_COMM_PROTOCOLn** (n: 1 - 7) parameters set the value of the **KDS_TEMS_COMM_PROTOCOLn** and **Kpp_AGT_COMM_PROTOCOLn** parameters.

This parameter offers an easy way to set all components to the same values, rather than setting parameters individually for each component.

Required or optional

Optional

Default value

None

Permissible values

Table 12: RTE_COMM_PROTOCOLn permissible values

Value	Protocol description	Corresponding protocol-specific parameters
IPPIPE	Non-secure TCP over IPv4	RTE_TCP_HOST RTE_TCP_PORT_NUM
IP	Non-secure UDP over IPv4	RTE_TCP_HOST RTE_TCP_UDP_PORT_NUM

Value	Protocol description	Corresponding protocol-specific parameters
IP6PIPE	Non-secure TCP over IPv6	RTE_TCP_HOST RTE_TCP_PIPE6_PORT_NUM
IP6	Non-secure UDP over IPv6	RTE_TCP_HOST RTE_TCP_UDP6_PORT_NUM
IPSPPIPE	Secure (SSL/TLS) TCP over IPv4	RTE_TCP_HOST RTE_TCP_PIPES_PORT_NUM
IPS6PIPE	Secure (SSL/TLS) TCP over IPv6	RTE_TCP_HOST RTE_TCP_PIPE6S_PORT_NUM
SNA	NCS RPC: Systems Network Architecture implementation of the Network Computing System Remote Procedure Call API	Kpp_TEMS_VTAM_APPL_LL_BROKER Kpp_TEMS_VTAM_LU62_DLOGMOD Kpp_TEMS_VTAM_LU62_MODETAB RTE_VTAM_NETID

Default values of **KDS_TEMS_COMM_PROTOCOLn** and **Kpp_AGT_COMM_PROTOCOLn**:

n	Value
1	IPPIPE
2	SNA

Example

Parameter	Value	Description
RTE_COMM_PROTOCOL1	IPSPPIPE	First choice: secure TCP over IPv4
RTE_COMM_PROTOCOL2	IPPIPE	Second choice: non-secure TCP over IPv4

Related parameters

- [KDS_TEMS_COMM_PROTOCOLn](#)
- [Kpp_AGT_COMM_PROTOCOLn](#)

RTE_NAME

The runtime environment name.

Description

The runtime environment name.

IBM Z® Monitoring Configuration Manager uses this name for various purposes, including:

- MVS™ member names
- MVS™ data set name qualifiers
- z/OS® UNIX® System Services directory name, all uppercase

Example value: RTE1

Permissible values

1 - 8 characters.

RTE_PLIB_HILEV

This parameter specifies the high-level qualifiers of the runtime environment definition (RTEDEF) library.

Description

This parameter specifies the high-level qualifiers of the runtime environment definition (RTEDEF) library.

The **RTE_PLIB_HILEV** parameter sets the default values of several parameters that specify runtime member locations:

RTE_HILEV
RTE_VSAM_HILEV
GBL_DSN_SYS1_PROCLIB
GBL_DSN_SYS1_VTAMLIB
GBL_DSN_SYS1_VTAMLST

Permissible values

To avoid exceeding the z/OS 44-character limit for data set names, the combined length of **RTE_NAME** and **RTE_PLIB_HILEV** should not exceed 28 characters. For example, if **RTE_NAME** is 8 characters, then **RTE_PLIB_HILEV** should not exceed 20 characters.

RTE_SECURITY_CLASS

This parameter specifies a System Authorization Facility (SAF) security class name for OMEGAMON enhanced 3270 user interface security controls.

Description

Use this parameter to specify the SAF security class for OMEGAMON enhanced 3270 user interface (enhanced 3270UI) security controls. The enhanced 3270UI performs security validation processing by authenticating the user identity using the SAF interface. The existence of the SAF user and its validity (that is, whether it is suspended) are always checked.

This parameter applies to the OMEGAMON enhanced 3270 user interface and the OMEGAMON monitoring agents that use the enhanced 3270UI. Individual products have additional SAF security settings that are specific to the respective product (for example, how to secure product-specific Take Action requests). To secure other products, see the product-specific documentation for information.

Important (for Configuration Manager users only): If a value is not specified for override parameter **Kpp_SECURITY_ACTION_CLASS** (where *pp* is C5, M5, or N3), then the **RTE_SECURITY_CLASS** parameter value will be assigned as the default value.

Required or optional

Optional

Default value

None

Permissible values

A valid SAF class name, which can be a string of up to 8 characters. If you are using ACF2 as your external security resource manager, specify a maximum of 3 characters.

Related parameters

- [KOB_SAF_ACTION_CLASS_NAME](#)
- [KC5_SECURITY_ACTION_CLASS](#)
- [KM5_SECURITY_ACTION_CLASS](#)
- [KN3_SECURITY_ACTION_CLASS](#)

RTE_SECURITY_FOLD_PASSWORD_FLAG

This parameter controls if password and password phrase (passphrase) values are folded to uppercase.

Description

By default, the TMS:Engine folds logon password and password phrase (passphrase) values to uppercase (parameter default value Y).

If you want to use mixed-case password or passphrase values, set this parameter to N so that the characters do not fold to uppercase, allowing mixed-case values to persist.

Note: Support for mixed-case passwords on your system requires activation of the SETROPTS PASSWORD(MIXEDCASE) option in RACF®.

Default value

Y

Permissible values

Y, N

Related parameters

RTE_SECURITY_USER_LOGON

RTE_SECURITY_USER_LOGON

The security system to be used for the runtime environment.

Description

The security system to be used for the runtime environment.

If you specify a security system, verify that it is installed and configured correctly for your site.

The **RTE_SECURITY_USER_LOGON** parameter specifies which system will be used to validate users signing on to the Tivoli® Enterprise Portal (TEP), but it does not enable validation. To enable validation of users signing on to TEP, the **KDS_TEMS_SECURITY_KDS_VALIDATE** parameter value must be Y (its default value).

Default value

NONE

Permissible values

NONE

No security.

RACF

IBM® z/OS® Security Server.

ACF2

CA ACF2.

If you specify ACF2, you must set the **GBL_DSN_ACF2_MACLIB** parameter to the name of the ACF2 macro library.

TSS

CA Top Secret.

NAM

Network Access Manager.

SAF

IBM® z/OS® System Authorization Facility API.

RTE_STC_PREFIX

The prefix of started task names for this runtime environment.

Description

The prefix of started task names for this runtime environment.

The **CREATE** action of IBM Z® Monitoring Configuration Manager sets the following value:

OMEG

(an abbreviation of OMEGAMON®).

Default value

IBM

Permissible values

1 - 4 characters.

RTE_TCP_HOST

This parameter contains the TCP/IP hostname or IP address of the z/OS® system where the runtime environment is being defined.

Description

This parameter contains the TCP/IP hostname or IP address of the z/OS® system where the runtime environment is being defined. To obtain the hostname and IP address, you can enter **TSO HOMETEST** at a command line.

For a high-availability hub monitoring server, this parameter value must be set to the dynamic virtual IP address (DVIPA) and must match the value in parameter **KDS_TEMS_TCP_HOST**.

For Configuration Manager:

To ensure consistent settings for the agents and components in the runtime environment, this parameter provides the default value for some other parameters that specify the hostname.

If you configure a monitoring server as part of your runtime environment, this parameter value is used as the default **KDS_TEMS_TCP_HOST** parameter value. Depending on other settings in the runtime environment, this parameter value might also be used as the default value for the **Kpp_TEMS_TCP_HOST** parameters.

If you do not configure a monitoring server as part of the runtime environment, this parameter value is used as the default value for the **Kpp_TEMS_TCP_HOST** parameters.

For PARMGEN:

This parameter does not impact the default setting of any other parameter value.

Default value

%SYSIPHOSTNAME%

Permissible values

Character string, maximum length 39

Related parameters

- [RTE_TCP_PORT_NUM](#)
- [KDS_TEMS_TCP_HOST](#)
- [KDS_HUB_TCP_HOST](#)
- [Kpp_TEMS_TCP_HOST](#)

RTE_TCP_PORT_NUM

This parameter contains the well-known port for the IP.PIPE communication protocol (the TCP/IP protocol that supports IPv4).

Description

Well-known port for the IP.PIPE communication protocol (the TCP/IP protocol that supports IPv4).

For Configuration Manager:

This parameter sets the default values of several parameters that specify port numbers, including:

KDS_TEMS_TCP_PIPE_PORT_NUM
Kpp_TEMS_TCP_PIPE_PORT_NUM

For PARMGEN:

This parameter does not impact the default setting of any other parameter value.

Default value

1918

Permissible values

1 - 65535

Related parameters

- [RTE_TCP_HOST](#)
- [KDS_TEMS_TCP_PIPE_PORT_NUM](#)

RTE_TEMS_NAME_NODEID

Identifies the monitoring server for internal processing.

Description

Identifies the monitoring server for internal processing.

The **KDS_HUB_TEMS_NAME_NODEID** parameter of remote monitoring servers must refer to the **RTE_TEMS_NAME_NODEID** of the hub monitoring server. For example, if the hub sets **RTE_TEMS_NAME_NODEID** to HUB:TEMS, then the runtime environments for remote monitoring servers must set **KDS_HUB_TEMS_NAME_NODEID** to HUB:TEMS.

The **CREATE** action of IBM Z® Monitoring Configuration Manager sets the following value:

```
rte_name:TEMS
```

where TEMS stands for Tivoli® Enterprise Monitoring Server, reflecting current terminology.

Default value

```
rte_name:CMS
```

RTE_TEMS_TRANSPORT_MODE

Use this parameter to specify the Hypertext Transfer Protocol to use for communication with the Tivoli Enterprise Monitoring Server.

Description

This parameter specifies the Hypertext Transfer Protocol to use for communication between the Tivoli Enterprise Monitoring Server (TEMS) and other components.

Note: HTTP and HTTPS communication protocol configuration specified in **Kpp_X_KDE_TRANSPORT_HTTP_OPTIONS** and **Kpp_X_KDE_TRANSPORT_GBL_OPTIONS** can potentially override **RTE_TEMS_TRANSPORT_MODE** parameter output in the *KppENV* members.

Permissible values**HTTPS**

Enables the HTTPS communication protocol for the Tivoli Enterprise Monitoring Server, and disables the HTTP communication protocol for all agents. Use parameter **KDS_TEMS_HTTPS_PORT_NUM** to specify the HTTPS port.

HTTP

Enables HTTP communication protocol for the Tivoli Enterprise Monitoring Server, and disables the HTTPS communication protocol. Use parameter **KDS_TEMS_HTTP_PORT_NUM** to specify the HTTP port.

NONE

Disables HTTP and HTTPS communication protocols for all agents.

Default value

HTTPS

Related parameters

- [KDS_TEMS_HTTP_PORT_NUM](#)
- [KDS_TEMS_HTTPS_PORT_NUM](#)
- [Kpp_X_KDE_TRANSPORT_HTTP_OPTIONS](#)
- [Kpp_X_KDE_TRANSPORT_GBL_OPTIONS](#)

RTE_USS_RTEDIR

The path where runtime members are stored in z/OS® UNIX® System Services.

Description

The path where runtime members are stored in z/OS® UNIX® System Services.

The runtime environment name, parameter **RTE_NAME**, is appended to the value of **RTE_USS_RTEDIR** as a directory name.

The TSO user ID that runs IBM Z® Monitoring Configuration Manager jobs must have permission to write to this directory, otherwise the **GENERATE** action will fail.

As an example, if **RTE_USS_RTEDIR** is `/var/rtehome` and **RTE_NAME** is `RTE1`, then runtime members are stored in:

```
/var/rtehome/RTE1/*
```

Default value

```
/var/rtehome
```

RTE_VTAM_APPLID_PREFIX

The global VTAM® applid prefix to be used to build the VTAM® applids for products in this runtime environment.

Description

The global VTAM® applid prefix to be used to build the VTAM® applids for products in this runtime environment.

The **CREATE** action of IBM Z® Monitoring Configuration Manager sets the following value:

```
OM $xx$ 
```

where xx is the value of the z/OS® static system symbol **SYSCLONE**. **SYSCLONE** is a 1- or 2-character shorthand notation for the system (LPAR) name.

Default value

```
CTD
```

RTE_X_SECURITY_EXIT_LIB

This parameter specifies the library containing the security exits used for the runtime environment.

Note: This parameter is valid in both PARMGEN and Configuration Manager; however, its function differs slightly. This topic describes this parameter as it applies to Configuration Manager.

Description

This parameter specifies the library containing the security exits used for the runtime environment.

The **RTE_X_SECURITY_EXIT_LIB** parameter specifies the name of the global runtime environment library that contains all of the OMEGAMON and IBM Tivoli Monitoring-related product security exits (such as

KOBSUPDT OMEGAMON KppSUPDI exits, Tivoli Monitoring Services: Engine security exits, external security exits).

Example value: TEST1.TST.DEMO.SECEXITS

Default value

`<rte_plib_hilev>.<rte_name>.SECEXITS`

Permissible values

A valid data set name.

Parameters with significant default values

The runtime environment defined by the initial set of parameters is configured not just by the relatively small number of parameters in that set, but also by the default values of many other parameters.

The following parameter is not included in the initial set, but its default value significantly affects the runtime environment.

RTE_TYPE

This parameter determines whether runtime members are a full stand-alone set or shared with SMP/E target installation libraries.

Description

This parameter determines whether runtime members are a full stand-alone set or shared with SMP/E target installation libraries.

Default value

Configuration Manager: SHARING

PARMGEN: FULL

Permissible values

The following descriptions apply to Configuration Manager:

FULL

Stand-alone runtime members. Runtime members have no dependency on target libraries.

SHARING

Some runtime members refer to the target libraries.

The high-level qualifiers of the target libraries are specified by the **GBL_TARGET_HILEV** parameter.

SHARING reduces the storage requirement for each runtime environment.

If **RTE_TYPE** is SHARING, then the value of the RTE_SHARE parameter must be SMP.

Parameters with different default values than PARMGEN

IBM Z® Monitoring Configuration Manager sets some different default parameter values than PARMGEN.

In some cases, instead of changing the default value of a parameter, Monitoring Configuration Manager sets a different example value in the initial set of parameters.

Table 15: Parameters with different default values in PARMGEN and Configuration Manager		
Parameter	PARMGEN default value and reason for change	Monitoring Configuration Manager default value
GBL_DSN_CSF_SCSFMODE0	In PARMGEN, this parameter is commented out in the WCONFIG (\$GBL\$USR) member. In Configuration Manager, it is explicitly added in the GBL\$PARM member that is generated after the CREATE or MIGRATE action, as it is relevant for several security-related aspects of the product configuration (such as password encryption). As described in the parameter description, if your installation does not use the Integrated Cryptographic Service Facility (ICSF), you can remove or comment out this parameter in your RTEDEF (GBL\$PARM) or RTEDEF (GBL\$lpar).	CSF . SCSFMODE0
GBL_DSN_SYS1_PROCLIB GBL_DSN_SYS1_VTAMLIB GBL_DSN_SYS1_VTAMLST	SYS1 . PROCLIB SYS1 . VTAMLIB SYS1 . VTAMLST Sites typically have their own procedures for copying members to system libraries, limited to z/OS® system administrators with special permissions.	rte_hilev.SYS1.PROCLIB rte_hilev.SYS1.VTAMLIB rte_hilev.SYS1.VTAMLST
KDS_KMS_SDA	N Using the self-describing agent (SDA) function is best practice.	Y
KDS_TEMS_TCP_KDEB_INTERFACELIST Kpp_AGT_TCP_KDEB_INTERFACELIST	None Specifying a default value is best practice.	!* Inherited from parameter RTE_TCP_KDEB_INTERFACELIST
KMQ_HISTCOLL_DATA_FLAG	N Collecting historical data is best practice.	Y
KMQ_STARTMON_ACTIVEONLY	NO Only monitoring active queue managers is best practice.	YES
KYN_XAI01_SUBAGENT_JAVAHOME	/usr/lpp/java/J7.1 There is no need for a default value specifically for this subagent. Use the existing global parameters as a default value instead.	Concatenation of the following two parameter values: gbl_hfs_java_dir1 gbl_hfs_java_dir2

Parameter	PARMGEN default value and reason for change	Monitoring Configuration Manager default value
RTE_TYPE	FULL SHARING reduces the storage requirement for each runtime environment.	SHARING Note: When RTE_TYPE is SHARING, then the value of the RTE_SHARE parameter must be SMP, which indicates sharing with SMP/E target libraries. Configuration Manager does not use base or sharing-with-base runtime environments (as in PARMGEN).
RTE_USS_RTEDIR	/rtehome Creating a new subdirectory of the root directory is bad practice.	/var/rtehome
RTE_X_SECURITY_EXIT_LIB	rte_hilev.rtename.RKANSAMU Identifies the security exits library currently used in PARMGEN.	rte_hilev.rte_name.SECEXITS

Parameters introduced by Monitoring Configuration Manager

IBM Z® Monitoring Configuration Manager introduces some parameters that do not exist in PARMGEN.

Global (GBL) parameter introduced by Configuration Manager

The global parameters provide default settings for installation and common system library names.

The global parameter introduced by IBM Z® Monitoring Configuration Manager is explained in this section.

GBL_USS_TKAYHFS_PATH

This parameter specifies the SMP/E TKAYHFS ddname installation directory.

Description

This parameter specifies the path of the z/OS UNIX directory that the SMP/E installation jobs for IBM Z OMEGAMON Data Provider define using the ddname TKAYHFS.

Depending on your local site practices, this path might refer to a copy of the SMP/E-managed directory rather than the original.

This directory is read-only for Monitoring Configuration Manager; Monitoring Configuration Manager does not write to this directory.

Default value

/usr/lpp/omdp

Permissible values

Character string, maximum length 44

GBL_UTIL_BINDER

This parameter allows you to override the default binder program name.

Description

To use a binder program other than the default IEWL, specify the name of the program in this parameter. If the name is in the list of documented IEWL aliases, then KCIOMEGA will rename the SYSPRINT, as it does for IEWL.

Required or optional

Required

Default value

IEWL

Permissible values

A valid MVS program name

Runtime environment (RTE) parameters introduced by Configuration Manager

The runtime environment parameters provide configuration settings for an individual runtime environment and default settings for the OMEGAMON components and products configured in that runtime environment.

The runtime environment parameters introduced by IBM Z® Monitoring Configuration Manager are explained in this section. The **RTE_COMM_PROTOCOLn**, **RTE_TCP_***, and **RTE_VTAM_NETID** parameters offer an easy way to set all components to the same values, rather than setting parameters individually for each component.

RTE_COMM_PROTOCOLn

This parameter sets the communication protocol choices of all components in the runtime environment.

Note: This parameter is valid in Configuration Manager only. It is not valid in PARMGEN.

Description

The **RTE_COMM_PROTOCOLn** (n: 1 - 7) parameters set the value of the **KDS_TEMS_COMM_PROTOCOLn** and **Kpp_AGT_COMM_PROTOCOLn** parameters. This parameter offers an easy way to set all components to the same values, rather than setting parameters individually for each component.

Required or optional

Optional

Default value

None

Permissible values

Table 16: RTE_COMM_PROTOCOLn permissible values

Value	Protocol description	Corresponding protocol-specific parameters
IPPIPE	Non-secure TCP over IPv4	RTE_TCP_HOST RTE_TCP_PORT_NUM
IP	Non-secure UDP over IPv4	RTE_TCP_HOST RTE_TCP_UDP_PORT_NUM
IP6PIPE	Non-secure TCP over IPv6	RTE_TCP_HOST RTE_TCP_PIPE6_PORT_NUM
IP6	Non-secure UDP over IPv6	RTE_TCP_HOST RTE_TCP_UDP6_PORT_NUM

Value	Protocol description	Corresponding protocol-specific parameters
IPSPIPE	Secure (SSL/TLS) TCP over IPv4	RTE_TCP_HOST RTE_TCP_PIPES_PORT_NUM
IPS6PIPE	Secure (SSL/TLS) TCP over IPv6	RTE_TCP_HOST RTE_TCP_PIPE6S_PORT_NUM
SNA	NCS RPC: Systems Network Architecture implementation of the Network Computing System Remote Procedure Call API	Kpp_TEMS_VTAM_APPL_LL_BROKER Kpp_TEMS_VTAM_LU62_DLOGMOD Kpp_TEMS_VTAM_LU62_MODETAB RTE_VTAM_NETID

Default values of **KDS_TEMS_COMM_PROTOCOLn** and **Kpp_AGT_COMM_PROTOCOLn**:

n	Value
1	IPSPIPE
2	SNA

Example

Parameter	Value	Description
RTE_COMM_PROTOCOL1	IPSPIPE	First choice: secure TCP over IPv4
RTE_COMM_PROTOCOL2	IPSPIPE	Second choice: non-secure TCP over IPv4

Related parameters

- [KDS_TEMS_COMM_PROTOCOLn](#)
- [Kpp_AGT_COMM_PROTOCOLn](#)

RTE_TCP_KDEB_INTERFACELIST

This Configuration Manager-only parameters directs all components in the runtime environment to connect to a specific TCP/IP local interface.

Description

The **RTE_TCP_KDEB_INTERFACELIST** parameter sets the value of the **KDS_TEMS_TCP_KDEB_INTERFACELIST** and **Kpp_AGT_TCP_KDEB_INTERFACELIST** parameters.

This parameter offers an easy way to set all components to the same values, rather than setting parameters individually for each component.

If the z/OS® image has more than one TCP/IP interface or network adapter, you can use this parameter to direct components to connect to a specific TCP/IP local interface.

Default value

!* (exclamation point followed by an asterisk)

Permissible values

Character string, maximum length 44, specifying one or more network interfaces to use.

To set a network interface list, supply one of the following values:

- The hostname or IP address of the preferred interface.
- A list of hostnames or IP addresses, in descending order of preference. Use a blank (space) to separate the entries.
- An asterisk (*) to prefer the interface associated with the default hostname for the z/OS® image. To display this value, enter TSO HOMETEST at the command line.
- An exclamation point followed by an asterisk (!*) to use only the interface associated with the default hostname for the z/OS® image.

- An exclamation point followed by a hostname or IP address (!*hostname*) to use only the interface associated with *hostname*.

Note:

- If you set the value of this parameter to !* or !*hostname*, you must specify the same value for every component and product configured in all runtime environments on the same z/OS® image.
- In the default character set (LANG=en_US.ibm-037), the code for an exclamation point is x'5A'. If you are using a character set other than the default, a different character might map to that code. To require a specific network interface, use the character that maps to x'5A' in your character set.

For a high-availability hub, specify the value of this parameter as !*dvipa_hostname*, where *dvipa_hostname* is the private DVIPA name set for the **KDS_TEMS_TCP_HOST** parameter.

RTE_TCP_PIPE6_PORT_NUM

Sets the port number for all components in the runtime environment that use the TCP over IPv6 communication protocol.

Description

The **RTE_TCP_PIPE6_PORT_NUM** parameter sets the value of the **KDS_TEMS_TCP_PIPE6_PORT_NUM** and **Kpp_TEMS_TCP_PIPE6_PORT_NUM** parameters.

This parameter is used only if one of the parameters that sets communication protocol choices, ***_COMM_PROTOCOLn**, specifies the value for this protocol, IP6PIPE.

This parameter offers an easy way to set all components to the same values, rather than setting parameters individually for each component.

Default value

none

Permissible values

A port number.

RTE_TCP_PIPE6S_PORT_NUM

Sets the port number for all components in the runtime environment that use the secure TCP over IPv6 communication protocol.

Description

The **RTE_TCP_PIPE6S_PORT_NUM** parameter sets the value of the **KDS_TEMS_TCP_PIPE6S_PORT_NUM** and **Kpp_TEMS_TCP_PIPE6S_PORT_NUM** parameters.

This parameter is used only if one of the parameters that sets communication protocol choices, ***_COMM_PROTOCOLn**, specifies the value for this protocol, IPS6PIPE.

This parameter offers an easy way to set all components to the same values, rather than setting parameters individually for each component.

Default value

none

Permissible values

A port number.

RTE_TCP_PIPES_PORT_NUM

Sets the port number for all components in the runtime environment that use the secure TCP over IPv4 communication protocol.

Description

The **RTE_TCP_PIPES_PORT_NUM** parameter sets the value of the **KDS_TEMS_TCP_PIPES_PORT_NUM** and **Kpp_TEMS_TCP_PIPES_PORT_NUM** parameters.

This parameter is used only if one of the parameters that sets communication protocol choices, ***_COMM_PROTOCOLn**, specifies the value for this protocol, IPSPIPE.

This parameter offers an easy way to set all components to the same values, rather than setting parameters individually for each component.

Default value

none

Permissible values

A port number.

RTE_TCP_STC

Sets the TCP/IP stack for all components in the runtime environment that use the IP communication protocol.

Description

The **RTE_TCP_STC** parameter sets the value of the **KDS_TEMS_TCP_STC** and **Kpp_AGT_TCP_STC** parameters.

This parameter offers an easy way to set all components to the same values, rather than setting parameters individually for each component.

This parameter sets the value of the **TCP/IP_USERID** parameter in the **KppINTCP** member of the **RKANPARU** library.

Setting **RTE_TCP_STC** to **#** (pound or hash sign) sets the value of **TCP/IP_USERID** to a blank (space), which allows TCP/IP to decide the stack associated with the address space, for better load balancing.

Default value

(pound or hash sign)

Permissible values

If the LPAR contains more than one TCP/IP stack, specify the started task name of the TCP/IP stack you want to use. Alternatively, specify a hash sign (**#**), which is translated to a blank and allows the TCP/IP environment to choose the stack to use, either through TCP/IP definitions or through the use of the **SYSTCPD DD** statement.

Whichever method is used to select a TCP/IP stack in a multi-stack environment, the Tivoli® Management Services components continue to use that stack, even if a different stack becomes the primary stack. Therefore, in a multi-stack environment, it is best to specify the started task name of the TCP/IP stack to be used, rather than specifying a wildcard or a blank.

RTE_TCP_UDP_PORT_NUM

Sets the port number for all components in the runtime environment that use the UDP over IPv4 communication protocol.

Description

The **RTE_TCP_UDP_PORT_NUM** parameter sets the value of the **KDS_TEMS_TCP_UDP_PORT_NUM** and **Kpp_TEMS_TCP_UDP_PORT_NUM** parameters.

This parameter is used only if one of the parameters that sets communication protocol choices, ***_COMM_PROTOCOLn**, specifies the value for this protocol, IP.

This parameter offers an easy way to set all components to the same values, rather than setting parameters individually for each component.

Default value

none

Permissible values

A port number.

RTE_TCP_UDP6_PORT_NUM

Sets the port number for all components in the runtime environment that use the UDP over IPv6 communication protocol.

Description

The **RTE_TCP_UDP6_PORT_NUM** parameter sets the value of the **KDS_TEMS_TCP_UDP6_PORT_NUM** and **Kpp_TEMS_TCP_UDP6_PORT_NUM** parameters.

This parameter is used only if one of the parameters that sets communication protocol choices, ***_COMM_PROTOCOLn**, specifies the value for this protocol, IP6.

This parameter offers an easy way to set all components to the same values, rather than setting parameters individually for each component.

Default value

none

Permissible values

A port number.

RTE_VTAM_NETID

Sets the VTAM® network ID for all components in the runtime environment that use the SNA communication protocol.

Note: This parameter is valid in both PARMGEN and Configuration Manager; however, its function differs slightly. This topic describes this parameter as it applies to Configuration Manager.

Description

The **RTE_VTAM_NETID** parameter sets the value of the **KDS_TEMS_VTAM_NETID** and **Kpp_TEMS_VTAM_NETID** parameters.

This parameter is used only if one of the parameters that sets communication protocol choices, ***_COMM_PROTOCOLn**, specifies the value for this protocol, SNAPPIPE.

This parameter offers an easy way to set all components to the same values, rather than setting parameters individually for each component.

Default value

none

Permissible values

A VTAM® network ID.

RTE_X_OVERRIDE_EMBEDS_LIB

This parameter specifies the name of the source library for override embed members.

Description

Override embed files can be used to add user-defined parameters and values that might be overwritten during the **GENERATE** action.

If the **KFJ_USE_EMBEDS** parameter is set to Y, this parameter is automatically defined in the *rte_plib_hilev.RTEDEF(rte_name)* member during an initial **CREATE** or **MIGRATE** action when creating an RTE.

Note: If the **KFJ_USE_EMBEDS** parameter was not set to Y when creating the RTE, you must manually add this parameter to the *rte_plib_hilev.RTEDEF(rte_name)* member to enable the use of override embed members for the RTE.

This library contains all the override embed members for all products installed in the respective CSI.

Example value: TEST1.TST.DEMO.EMBEDS

Default value

<rte_plib_hilev>.<rte_name>.EMBEDS

Permissible values

A valid MVS data set name.

Related parameters

[KFJ_USE_EMBEDS](#)
[KFJ_EMBEDS_LIB](#)

RTE_X_SECURITY_EXIT_LIB

This parameter specifies the library containing the security exits used for the runtime environment.

Note: This parameter is valid in both PARMGEN and Configuration Manager; however, its function differs slightly. This topic describes this parameter as it applies to Configuration Manager.

Description

This parameter specifies the library containing the security exits used for the runtime environment.

The **RTE_X_SECURITY_EXIT_LIB** parameter specifies the name of the global runtime environment library that contains all of the OMEGAMON and IBM Tivoli Monitoring-related product security exits (such as KOBSPDPT OMEGAMON *KppSUPDI* exits, Tivoli Monitoring Services: Engine security exits, external security exits).

Example value: TEST1.TST.DEMO.SESEXITS

Default value

<rte_plib_hilev>.<rte_name>.SESEXITS

Permissible values

A valid data set name.

Configuration Manager (KFJ) parameters

The Configuration Manager parameters have the prefix KFJ.

KFJ_ADRDSSU_ADMIN

This parameter specifies whether to include the ADMINISTRATOR keyword with z/OS DFSMSdss commands.

Description

The **PACKAGE** and **DEPLOY** actions use z/OS DFSMSdss commands when creating and restoring dump data sets. The **ADMINISTRATOR** keyword allows you to act as a DFSMSdss-authorized storage administrator. Specify this value in accordance with security settings at your site.

Y

Include the **ADMINISTRATOR** keyword

N

Do not include the **ADMINISTRATOR** keyword

Notes:

- You might need authority to run **ADDRSSU**, which is the program that is invoked when using DFSMSdss.
- You might need additional authority to use the **ADMINISTRATOR** keyword. For more information about using the **ADMINISTRATOR** keyword, see the following topics in the *z/OS DFSMSdss Storage Administration Reference*: [ADMINISTRATOR keyword](#), [FACILITY class profiles for the ADMINISTRATOR keyword](#)

Default value

N

Permissible values

Y, N

KFJ_EMBEDS_LIB

This parameter identifies the data set that contains the override embed values for the RTE.

Description

Use this parameter to specify the name of the data set (the *embeds data set*) that will contain the override embed values to use for the RTE. The data set must be accessible by Monitoring Configuration Manager. An example of the data set name is *highlevel.CFM.RTEDEF.EMBEDS*.

This parameter is active only if **KFJ_USE_EMBEDS** is set to Y.

The default name of the embeds data set is *rte_plib_hilev.rte_name.EMBEDS*, which is intended for a single RTE. By using **KFJ_EMBEDS_LIB**, you can customize the name of the embeds data set and use the same override embed parameters and values for multiple RTEs. If you will be using different override values for different RTEs, consider using the *rte_name* or *sysname* values in the data set name.

If the specified data set does not exist when the **CREATE** or **MIGRATE** action runs, Monitoring Configuration Manager creates the data set and populates it with the override embed parameters and values for the products that are installed in the respective CSI used to build the RTE. For the **MIGRATE** action, this library will contain the override embed parameters and values from the RTE being migrated from.

If the specified data set exists when the **CREATE** or **MIGRATE** action runs, none of the existing members in it are changed or removed; only new members are added, which can occur if the CSI used to build the RTE has changed to add products.

Required or optional

Optional

Default value

rte_plib_hilev.rte_name.EMBEDS

Permissible values

A valid MVS data set name

Example

TEST1.TST.DEMO.EMBEDS

Related parameters

KFJ_USE_EMBEDS

"RTE_X_OVERRIDE_EMBEDS_LIB" on page 98

KFJ_LOCAL_HILEV

Assigns a local non-VSAM high-level qualifier.

The KFJ_LOCAL_HILEV parameter is specified in the RTEDEF(PCK\$PARM) or RTEDEF(PCK\$*par*) member, and identifies the non-VSAM high-level qualifier to be used for allocating the local runtime data sets when the GENERATE action is run. The names of the non-VSAM data sets will be generated by appending the appropriate suffix to this parameter.

KFJ_LOCAL_HILEV maps to the local value of RTE_HILEV that is used on the deployment target system.

Required or optional

Optional

Default value

%KFJ_LOCAL_PLIB_HILEV%

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_LOCAL_KD5_RUN_ALLOC

This parameter controls if the job for Db2-related data set allocation is submitted. This parameter is used for remote deployment scenarios.

Description

In a remote deployment scenario, you can use this parameter to specify whether to submit the Db2-related data set allocation job. This job allocates all operational data sets required for the enabled functions (for example, to collect data for Thread History). This job does not overwrite existing operational data sets.

Specify one of the following values:

GENERATE

Trigger the ALLOCDS JCL job during the GENERATE action. With this option, you cannot customize the KD2 operational data set allocation parameters.

DEPLOY

Trigger the ALLOCDS JCL job during the DEPLOY action. With this option, you can customize the KD2 operational data set allocation parameters.

NONE

Do not trigger the ALLOCDS JCL job. With this option, you can customize the KD2 operational data set allocation parameters.

Tip: For non-remote deployment scenarios (PACKAGE, DEPLOY), you can use parameter KD2_OMPE_RUNALLOC to perform the same function.

Required or optional

Optional

Default value

GENERATE

Permissible values

GENERATE, DEPLOY, NONE

Related parameters

KD2_OMPE_RUNALLOC

KFJ_LOCAL_PDS_HILEV

Specifies the PDS V1 local high-level qualifier.

The KFJ_LOCAL_PDS_HILEV parameter is specified in the RTEDEF(PCK\$PARM) or RTEDEF(PCK\$*lpar*) member, and identifies the high-level qualifier for the local Persistent Data Store (PDS) V1 data sets. It maps to the local value of RTE_PDS_HILEV that is used on the deployment target system.

Required or optional

Optional

Default value

%KFJ_LOCAL_HILEV%.%RTE_NAME%

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_LOCAL_PLIB_HILEV

This parameter specifies the high-level qualifiers for local libraries.

Description

This parameter allows for local generation of runtime environment data sets RTEDEF, EMBEDS, and SECEXITS.

Specifying this member triggers the generation of member PCK\$PARM (or PCK\$*lpar*) in the RTEDEF data set. This parameter populates the default values for other **KFJ_LOCAL_*** high-level qualifier parameters in the PCK\$* members, such as **KFJ_LOCAL_HILEV** and **KFJ_LOCAL_VSAM_HILEV**.

For remote deployment scenarios, this parameter allows local libraries to be used for creating, configuring, and packaging a runtime environment that can then be deployed to another system that uses different high-level qualifiers from the configuration system. For more information, see [“Remote deployment example using local libraries” on page 147](#).

This parameter can be used with the following actions: **CREATE, MIGRATE, GENERATE, PACKAGE, DELETE**

This parameter is specified in the KCIVARS DD statement in the action job JCL and is not stored in an RTEDEF member.

Default value

N/A

Related parameters

- [RTE_PLIB_HILEV](#)
- [KFJ_PACK_HILEV](#)

KFJ_LOCAL_SMS_MGMTCLAS

Identifies the MGMTCLAS for local non-VSAM libraries.

The KFJ_LOCAL_SMS_MGMTCLAS parameter is specified in the RTEDEF(PCK\$PARM) or RTEDEF(PCK\$*lpar*) member and indicates the SMS Management Class to be used when allocating the local non-VSAM runtime data sets. KFJ_LOCAL_SMS_MGMTCLAS maps to the local value of RTE_SMS_MGMTCLAS that is used on the deployment target system.

Required or optional

Optional

Default value

None

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_LOCAL_SMS_STORCLAS

Identifies the STORCLAS for local non-VSAM libraries.

This parameter is specified in the RTEDEF(PCK\$PARM) or RTEDEF(PCK\$*par*) member, and defines the SMS Storage Class to be used when allocating the local non-VSAM runtime data sets. KFJ_LOCAL_SMS_STORCLAS maps to the local value of RTE_SMS_STORCLAS that is used on the deployment target system.

Required or optional

Optional

Default value

None

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_LOCAL_SMS_UNIT

Specifies the Unit name for local non-VSAM libraries.

The KFJ_LOCAL_SMS_UNIT parameter is specified in the RTEDEF(PCK\$PARM) or RTEDEF(PCK\$*par*) member, and identifies the unit name to be used when allocating the local non-VSAM runtime data sets. KFJ_LOCAL_SMS_UNIT maps to the local value of RTE_SMS_UNIT that is used on the deployment target system.

This is a required field if the runtime data sets are not to be SMS-managed.

Required or optional

Required if the runtime data sets are not to be SMS-managed

Default value

None

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_LOCAL_SMS_VOLUME

Specifies the VOLSER for the local non-VSAM libraries.

This parameter is specified in the RTEDEF(PCK\$PARM) or RTEDEF(PCK\$*par*) member, and identifies the volume serial number to be used when allocating the local non-VSAM runtime data sets. KFJ_LOCAL_SMS_VOLUME maps to the local value of RTE_SMS_VOLUME that is used on the deployment target system.

This is a required field if the runtime data sets are not to be SMS-managed.

Required or optional

Required if the runtime data sets are not to be SMS-managed

Default value

None

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_LOCAL_SMS_VSAM_MGMTCLAS

Specifies the MGMTCLAS for any local VSAM libraries.

This parameter specifies the SMS Management Class to be used when allocating the local VSAM runtime data sets. KFJ_LOCAL_SMS_VSAM_MGMTCLAS maps to the local value of RTE_SMS_VSAM_MGMTCLAS that is used on the deployment target system.

Required or optional

Optional

Default value

None

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_LOCAL_SMS_VSAM_STORCLAS

Specifies the STORCLAS for local VSAM libraries.

The KFJ_LOCAL_SMS_VSAM_STORCLAS parameter is specified in the RTEDEF(PCK\$PARM) or RTEDEF(PCK\$*par*) member, and identifies the SMS Storage Class to be used when allocating the local VSAM runtime data sets. KFJ_LOCAL_SMS_VSAM_STORCLAS maps to the local value of RTE_SMS_VSAM_STORCLAS that is used on the deployment target system.

Required or optional

Optional

Default value

None

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_LOCAL_SMS_VSAM_VOLUME

Specifies the VOLSER for local VSAM libraries.

This parameter is specified in the RTEDEF(PCK\$PARM) or RTEDEF(PCK\$*par*) member, and identifies the volume serial number to be used when allocating the local VSAM runtime data sets. KFJ_LOCAL_SMS_VSAM_VOLUME maps to the local value of RTE_SMS_VSAM_VOLUME that is used on the deployment target system.

This is a required field if the runtime data sets are not to be SMS-managed.

Required or optional

Required if the runtime data sets are not to be SMS-managed

Default value

None

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_LOCAL_TARGET_HILEV

Assigns the local SMP/E target high-level qualifier.

This parameter is specified in the RTEDEF(PCK\$PARM) or RTEDEF(PCK\$*par*) member and identifies the high-level qualifier to be used when referencing the local SMP/E target data sets as part of the GENERATE action. The names of SMP/E target data sets will be generated by appending the appropriate suffix to this parameter.

KFJ_LOCAL_TARGET_HILEV maps to the local value of the GBL_TARGET_HILEV parameter, which is used on the deployment target system.

Required or optional

Optional

Default value

None

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_LOCAL_USS_RTEDIR

Specify the local RTE HFS/zFS home directory.

This parameter is specified in the RTEDEF(PCK\$PARM) or RTEDEF(PCK\$*lpar*) member. If any products in this RTE require z/OS® UNIX® System Services directories to be created, specify the local RTE HFS/zFS home directory. KFJ_LOCAL_USS_RTEDIR maps to the local value of RTE_USS_RTEDIR that is used on the deployment target system.

Required or optional

Required

Default value

`/var/rtehome`

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_LOCAL_USS_TKANJAR_PATH

Specifies the local z/OS® UNIX® System Services directory that the SMP/E installation jobs define using the ddname TKANJAR.

This parameter is specified in the RTEDEF(PCK\$PARM) or RTEDEF(PCK\$*lpar*) member. The **KFJ_LOCAL_USS_TKANJAR_PATH** parameter maps to the value of the **GBL_USS_TKANJAR_PATH** parameter that is used on the deployment target system.

Required or optional

Required if the runtime environment configures either of the following monitoring agents:

- CICS® Transaction Gateway (TG). The corresponding configuration parameter is **CONFIGURE_CICS_TG_KGW**.
- Java™ Virtual Machine (JVM). The corresponding configuration parameter is **CONFIGURE_JVM_KJJ**.

Default value

`/usr/lpp/kan/bin/IBM`

Location where the parameter value is stored

This value is not stored in a configuration member.

Related parameters

[“GBL_USS_TKANJAR_PATH” on page 83](#)

KFJ_LOCAL_USS_TKAYHFS_PATH

This parameter specifies the local z/OS UNIX System Services directory that the SMP/E installation jobs define using the ddname TKAYHFS.

Description

This parameter is specified in the RTEDEF (PCK\$PARM) or RTEDEF (PCK\$*lpar*) member.

This parameter maps to the value of the **GBL_USS_TKAYHFS_PATH** parameter that is used on the deployment target system.

Default value

`/usr/lpp/omdp`

Permissible values

Character string, maximum length 44

KFJ_LOCAL_VSAM_HILEV

Specifies the local VSAM high-level qualifier.

This parameter is specified in the RTEDEF(PCK\$PARM) or RTEDEF(PCK\$lpar) member, and identifies the VSAM high-level qualifier to be used for allocating the local runtime data sets when the GENERATE action is run. The names of the VSAM data sets will be generated by appending the appropriate suffix to this parameter. KFJ_LOCAL_VSAM_HILEV maps to the local value of RTE_VSAM_HILEV that is used on the deployment target system.

Required or optional

Optional

Default value

%KFJ_LOCAL_PLIB_HILEV%

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_MIGRATE_WCONFIG

This parameter identifies the WCONFIG data set of the PARMGEN-configured runtime environment from which configuration settings are to be imported during migration.

Description

The **MIGRATE** action imports configuration settings from a runtime environment that is configured with PARMGEN to one that is configured with Configuration Manager. The **KFJ_MIGRATE_WCONFIG** parameter is used by the **MIGRATE** action to identify the WCONFIG data set of the source runtime environment (which is configured by PARMGEN).

Important: The source WCONFIG data set cannot be the same as the target WCONFIG data set. Make sure that the high-level qualifier specified for the source WCONFIG data set is not the same as the high-level qualifier of the target data sets (specified with the **RTE_PLIB_HILEV** parameter on the **MIGRATE** action).

Required or optional

Required for the **MIGRATE** action

Default value

None

Permissible values

highlevel.WCONFIG

Where *highlevel* is different from the target high-level qualifier specified in **RTE_PLIB_HILEV**

Example

```
TEST1.TST.DEMO.WCONFIG
```

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_PACK_DATACLAS

Specifies the DATACLAS for non-VSAM package data sets.

This parameter is used by the PACKAGE and DEPLOY action, and specifies the SMS Data Class to be used when allocating or reading the non-VSAM package data sets. This parameter is not stored in any RTEDEF member and will be specified in the KCIVARS DD statement of your Configuration Manager batch JCL.

Required or optional

Optional

Default value

None

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_PACK_HILEV

This parameter specifies the non-VSAM high-level qualifiers for allocating and reading the package data sets used in remote deployment.

Description

This parameter specifies the high-level qualifiers for the package data sets that are allocated and populated by the **PACKAGE** action and deployed (restored) by the **DEPLOY** action for remote deployment of runtime environments.

This parameter is required for the **DEPLOY** action if you transfer your package data sets from the configuration system to the remote (target) system with a qualifier that is different from the **RTE_PLIB_HILEV** parameter value.

For details about using this parameter, see [“PACKAGE” on page 64](#) and [“DEPLOY” on page 67](#).

This parameter is specified in the KCIVARS DD statement in the action job JCL and is not stored in an RTEDEF member.

Default value

None

Related parameters

- [RTE_PLIB_HILEV](#)
- [KFJ_LOCAL_PLIB_HILEV](#)

KFJ_PACK_MGMTCLAS

Specifies the MGMTCLAS for non-VSAM package data sets.

The KFJ_PACK_MGMTCLAS parameter is used by the PACKAGE and DEPLOY action, and specifies the SMS Management Class to be used when allocating or reading the non-VSAM package data sets. This parameter is not stored in any RTEDEF member and will be specified in the KCIVARS DD statement of your Configuration Manager batch JCL.

Required or optional

Optional

Default value

None

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_PACK_STORCLAS

Specifies the STORCLAS for non-VSAM package data sets.

The KFJ_PACK_STORCLAS parameter is used by the PACKAGE and DEPLOY action, and specifies the SMS Storage Class to be used when allocating or reading the non-VSAM package data sets. This parameter is not stored in any RTEDEF member and will be specified in the KCIVARS DD statement of your Configuration Manager batch JCL.

Required or optional

Optional

Default value

None

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_PACK_TERSE

Specifies to use the terse data set option.

The KFJ_PACK_TERSE parameter works as a switch during PACKAGE and DEPLOY actions:

- **Y** – Terse the packages created during PACKAGE action, that is, append suffix "TRS" to the package data set. Instruct the DEPLOY action to read tersed package data sets. Use KFJ_PACK_TERSE Y if you transfer package data sets using FTP.
- **N** – Do not terse the package data sets created by the PACKAGE action. The DEPLOY action will read the DUMP data sets using the documented naming convention. Use KFJ_PACK_TERSE N when working with Virtual Tapes.

This parameter is not stored in any RTEDEF member and will be specified in the KCIVARS DD statement of your Configuration Manager batch JCL.

Required or optional

Optional

Default value

N

Permissible values

Y, N

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_PACK_UNIT

Specifies the Unit value for non-VSAM package data sets.

The KFJ_PACK_UNIT parameter is used by the PACKAGE and DEPLOY action, and specifies the UNIT name to be used when allocating or reading the non-VSAM package data sets. This parameter is not stored in any RTEDEF member and will be specified in the KCIVARS DD statement of your Configuration Manager batch JCL.

This parameter is required if the package data sets are not to be SMS-managed.

Required or optional

Required if the package data sets are not to be SMS-managed

Default value

None

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_PACK_VOLUME

Specifies the volume for the non-VSAM package data set.

The KFJ_PACK_VOLUME parameter is used by the PACKAGE and DEPLOY actions, and specifies the volume serial number/name to be used when allocating or reading the non-VSAM package data sets. This parameter is not stored in any RTEDEF member and is specified in the KCIVARS DD statement of the Configuration Manager batch JCL.

This parameter is required if the package data sets are not to be SMS-managed.

Required or optional

Required if the package data sets are not to be SMS-managed

Default value

None

Location where the parameter value is stored

This value is not stored in a configuration member.

KFJ_PDCOL_ADR_ARCHIVEThis parameter specifies the name of the ADRDSSU archive for the **PDCOLLECT** action to restore.

Note: This parameter is for use by IBM Software Support only.

DescriptionThis parameter specifies the name of the **ADRDSSU** archive for the **PDCOLLECT** action to restore.**Default value**

None

Permissible values

Character string, maximum length 44

Related parameters[“KFJ_PDCOLLECT_RESTORE” on page 114](#)[“KFJ_PDCOL_RESTORE_HLQ” on page 112](#)**KFJ_PDCOL_COLLECT_data**This parameter (with its variations) specifies the diagnostic data items that the **PDCOLLECT** action collects. You can control the collection of each of the items individually.**Description**This parameter specifies the diagnostic data that the **PDCOLLECT** action collects at a detailed level. The variations of this parameter are referred to as *collection flags*. You can control each of the flags individually.

The following table lists the collection flags that are available and describes the content that each setting collects.

<i>Table 19: PDCOLLECT collection flags</i>	
Parameter	Description
KFJ_PDCOL_COLLECT_STC	Collects the started task log
KFJ_PDCOL_COLLECT_NETSTAT	Collects network information using the netstat utility
KFJ_PDCOL_COLLECT_DASTATS	Collects SDSF Display Active Users (DA) statistics
KFJ_PDCOL_COLLECT_RK	Collects libraries RKANSAMU, RKANPARU, RKANCMDU, RKANDATV, RKCPDEFW, RKD2PAR, RKD2SAM
KFJ_PDCOL_COLLECT_TRG	Collects libraries <i>x</i> KANCMD, <i>x</i> KANSAM, <i>x</i> KANPAR, where <i>x</i> is T or R (shared or full RTE)
KFJ_PDCOL_COLLECT_LISTCAT	Collects LISTCAT command output, which lists all the data sets under the runtime environment qualifiers
KFJ_PDCOL_COLLECT_RTEDEF	Collects the runtime environment definition (RTEDEF) library
KFJ_PDCOL_COLLECT_EMBEDS	Collects the EMBEDS data set
KFJ_PDCOL_COLLECT_USERJCL	Collects the data set specified in the GBL_USER_JCL parameter
KFJ_PDCOL_COLLECT_WCONFIG	Collects the WCONFIG data set
KFJ_PDCOL_COLLECT_SNMP	Collects member RKANSAMU (KN3SNMP) or the data set specified in parameter KN3_SNMP_CONFIG_FILE

Parameter	Description
KFJ_PDCOL_COLLECT_STDENV	Collects the data sets mapped to the SDTENV DD statement
KFJ_PDCOL_COLLECT_SDADIR	Collects various information about the self-describing agent (SDA) directory, permission bits, contents, spawn_cmd .sh contents
KFJ_PDCOL_COLLECT_QA1	Collects QA1 files from the Tivoli Enterprise Monitoring Server started task <div style="border: 1px solid blue; padding: 5px;"> <p>Note: When producing output in the PAX (.pax) file format, the PDCOLLECT action writes the QA1 files to the z/OS® UNIX® System Services file system in hexadecimal print format. Additionally, the hex print data is converted to ASCII format and placed in the /converted subdirectory.</p> </div>
KFJ_PDCOL_COLLECT_OPSLOG	Collects data sets mapped to the RKMSLG DD statement
KFJ_PDCOL_COLLECT_MANIFEST	Collects component level information

Default value

For all flags except parameter **KFJ_PDCOL_COLLECT_OPSLOG**: Y
For parameter **KFJ_PDCOL_COLLECT_OPSLOG**: N

Permissible values

Y, N

KFJ_PDCOL_DA_INTERVAL

This parameter specifies the interval (in seconds) for SDSF DA panel statistics collection.

Description

This parameter specifies the interval (in seconds) for the PDCOLLECT action to use for SDSF Display Active Users (DA) panel statistics collection.

Default value

10

Permissible values

Integer

Related parameters

[“KFJ_PDCOL_DA_SAMPLES” on page 110](#)

KFJ_PDCOL_DA_SAMPLES

This parameter specifies the number of samples to take per interval for SDSF DA panel statistics collection.

Description

This parameter specifies the number of samples for the PDCOLLECT action to take per interval for SDSF Display Active Users (DA) panel statistics collection.

Default value

10

Permissible values

Integer

Related parameters

[“KFJ_PDCOL_DA_INTERVAL” on page 110](#)

KFJ_PDCOL_HLQ

This parameter specifies the high-level qualifier for the data sets that the **PDCOLLECT** action uses.

Description

This parameter specifies the high-level qualifier for the data sets that the **PDCOLLECT** action uses.

These data sets include the collection configuration data set (PDCLDEF) and generated output files in terse format (PDCOLPDS . TRS).

Default value

&SYSUID.KCIPDCOL

Permissible values

An MVS data set name, maximum length 31

KFJ_PDCOL_JOB_FILTER

This parameter specifies if job filtering is enabled for the **PDCOLLECT** action.

Description

This parameter specifies if job filtering is enabled for the **PDCOLLECT** action.

When job filtering is enabled, you can use the asterisk (*) and percent sign (%) as wildcard characters in the **KFJ_PDCOL_JOB_NAME** and **KFJ_PDCOL_JOB_ID** parameter values to create the job filters.

Default value

Y

Permissible values

Y, N

Related parameters

[“KFJ_PDCOL_JOB_ID” on page 111](#)

[“KFJ_PDCOL_JOB_NAME” on page 111](#)

KFJ_PDCOL_JOB_ID

This parameter specifies the job ID of the started task for which the **PDCOLLECT** action collects diagnostic data.

Description

This parameter specifies the job ID (located in the SDSF job output queue) of the started task for which the **PDCOLLECT** action collects diagnostic data.

This parameter must be specified with parameter **KFJ_PDCOL_JOB_NAME** and should point to the same job output.

Default value

When **KFJ_PDCOL_JOB_FILTER** is set to N: JOBID

When **KFJ_PDCOL_JOB_FILTER** is set to Y: *

Permissible values

Character string, maximum length 8

KFJ_PDCOL_JOB_NAME

This parameter specifies the job name of the started task for which the **PDCOLLECT** action collects diagnostic data.

Description

This parameter specifies the job name (located in the SDSF job output queue) of the started task for which the **PDCOLLECT** action collects diagnostic data.

This parameter must be specified with parameter **KFJ_PDCOL_JOB_ID** and should point to the same job output.

Default value

JOBNAME

Permissible values

Character string, maximum length 8

KFJ_PDCOL_JOB_OUTPUT

This parameter specifies the name of the data set containing the job output of the started task for which the **PDCOLLECT** action collects diagnostic data.

Description

This parameter specifies the name of the data set containing the job output of the OMEGAMON address space started task for which the **PDCOLLECT** action collects diagnostic data. Use this parameter if your OMEGAMON address space logs have been copied to a sequential data set.

Note: If parameters **KFJ_PDCOL_JOB_NAME** and **KFJ_PDCOL_JOB_ID** are specified in the KCIVARS DD statement, parameter **KFJ_PDCOL_JOB_OUTPUT** is ignored.

Default value

None

Permissible values

An MVS data set name, maximum length 44

KFJ_PDCOL_RESTORE_HLQ

This parameter specifies the high-level qualifier for the **PDCOLLECT** action to use for **ADDRSSU** archive restore.

Note: This parameter is for use by IBM Software Support only.

Description

This parameter specifies the high-level qualifier for the **PDCOLLECT** action to use for **ADDRSSU** archive restore.

The **ADDRSSU** program uses this parameter value as the first qualifier for a rename option.

Default value

None

Permissible values

An MVS data set name, maximum length 8. The value must be a single qualifier.

Related parameters

[“KFJ_PDCOL_ADR_ARCHIVE” on page 109](#)

[“KFJ_PDCOLLECT_RESTORE” on page 114](#)

KFJ_PDCOLLECT_ADDRSSU

This parameter specifies that the **PDCOLLECT** action store the collected diagnostic data in an **ADDRSSU** archive.

Description

This parameter specifies that the **PDCOLLECT** action store the collected diagnostic data in an **ADRSSU** archive.

The file `ADR.ARCHIVE` is generated.

This option requires the use of the `KCIALPHA` program.

Default value

N

Permissible values

Y, N

KFJ_PDCOLLECT_COMPATIBILITY

This parameter specifies that the **PDCOLLECT** action store the collected diagnostic data in a terse data set.

Description

This parameter specifies that the **PDCOLLECT** action store the collected diagnostic data in a terse data set.

With this option, the **PDCOLLECT** action uses the `IEBCOPY UNLOAD` function to package the diagnostic data.

The file `kfj_pdcoll_hlq.PDCOLPDS.TRS` is generated.

Default value

Y

Permissible values

Y, N

KFJ_PDCOLLECT_PAX

This parameter specifies that the **PDCOLLECT** action store the collected diagnostic data in a PAX archive.

Description

This parameter specifies that the **PDCOLLECT** action store the collected diagnostic data in a PAX archive.

This parameter must be specified with parameter **KFJ_PDCOLLECT_PAX_DIR**.

The file `kcipdcol.pax` is generated.

Default value

N

Permissible values

Y, N

Related parameters

[KFJ_PDCOLLECT_PAX_DIR](#)

KFJ_PDCOLLECT_PAX_DIR

This parameter specifies the **PDCOLLECT** action PAX archive path.

Description

This parameter specifies the **PDCOLLECT** action PAX archive path.

This value must be a valid `z/OS® UNIX® System Services` directory and is used as the root to create directory `kcipdcol/`.

This parameter must be specified with parameter **KFJ_PDCOLLECT_PAX**.

Default value

None

Permissible values

Character string, maximum length 50

Related parameters

[KFJ_PDCOLLECT_PAX](#)

KFJ_PDCOLLECT_RESTORE

This parameter specifies that the **PDCOLLECT** action restore diagnostic data from a given archive.

Note: This parameter is for use by IBM Software Support only.

Description

This parameter specifies that the **PDCOLLECT** action restore diagnostic data from a given archive.

Default value

N

Permissible values

Y, N

Related parameters

[“KFJ_PDCOL_ADR_ARCHIVE” on page 109](#)

[“KFJ_PDCOL_RESTORE_HLQ” on page 112](#)

KFJ_SECURITY_EXITS_LIB

This parameter specifies the name of the security exits library for the runtime environment.

Description

Use this parameter to specify the name of the data set that will contain the security exits for the runtime environment.

The default name for the security exits data set is *rte_plib_hilev.rte_name*.SECEXITS. By using the **KFJ_SECURITY_EXITS_LIB**, you can customize the name of the security exits data set.

The **KFJ_SECURITY_EXITS_LIB** parameter works with the **CREATE** and **MIGRATE** actions only.

If the specified data set does not exist when the **CREATE** or **MIGRATE** action runs, Configuration Manager allocates the data set and populates it with the default security exit members. For the **MIGRATE** action, this library will contain the security exit members from the source PARMGEN runtime environment.

Required or optional

Optional

Default value

rte_plib_hilev.rte_name.SECEXITS

Permissible values

A valid data set name

Example

TEST1.TST.DEMO.MYEXITS

Location where the parameter value is stored

This value is not stored in a configuration member.

Related parameters

[“RTE_X_SECURITY_EXIT_LIB” on page 90](#)

KFJ_SYSNAME

This parameter defines the system name (SYSNAME), LPAR name, or SYSSMFID to use in runtime environment configuration members.

Note: **KFJ_SYSNAME** is an internal parameter that automatically assigns the value of **SYSNAME**, but it is possible to specify a different value for **KFJ_SYSNAME** in the **KCIVARS DD** as well.

Description

This parameter is specified in the **KCIVARS DD** statement of the JCL that runs the Configuration Manager jobs (for example, the **KFJJMCM** sample job in **TKANSAM**).

This parameter is used for the following purposes:

- Used with **OPTION MULTIPLE** to create (or migrate) multiple runtime environments in a single RTEDEF library. The **KFJ_SYSNAME** parameter value replaces *lpar* in member name *Kpp\$lpar* for the generated members. For more information, see [“CREATE” on page 37](#) and [“MIGRATE” on page 60](#).
- Used for remote deployment to generate runtime environments on a single local system that can then be deployed to remote target systems. For more information, see [“Deploying remote runtime environments” on page 145](#).
- Used to assign the system name if you use a **SYSNAME** that is larger than 4 characters (5 - 8 characters) and the **SYSSMFID** is not assigned or the default **SYSSMFID** setting is not acceptable. Normally, a **SYSNAME** is 1 - 4 characters in length; however, if a system has a longer **SYSNAME**, you can use the **KFJ_SYSNAME** parameter to override it. The **KFJ_SYSNAME** parameter value replaces the value assigned to the **&SYSNAME** parameter when generating member names for the **DISCOVER**, **GENERATE**, and **MIGRATE** actions. The following rules apply:
 - If the value for **&SYSNAME** is 4 characters or less, the value for **KFJ_SYSNAME** equals the **&SYSNAME**.
 - If the value for **&SYSNAME** is 5 - 8 characters in length, the value for **KFJ_SYSNAME** equals the **&SYSSMFID** parameter, which is the SMF ID.
 - If the value for **&SYSSMFID** is not set explicitly, the system defaults to the CPU model number.

If the **KFJ_SYSNAME** value has been set for **&SYSSMFID**, warning message **KFJ00205W** is issued in the **KCIPRINT** member. This message states that you can provide (override) a custom **KFJ_SYSNAME** value (in **KCIVARS DD**) when running the Configuration Manager job. This action allows you to avoid duplicate member names in the RTEDEF data set, in case the **&SYSSMFID** is set to default to the CPU model number.

KFJ_USE_EMBEDS

This parameter controls whether override embed members are enabled for the RTE.

Description

A value of **Y** indicates that override embed members are enabled for the RTE.

For a **CREATE** or **MIGRATE** action, when this parameter is set to **Y**, Monitoring Configuration Manager creates a data set (the *embeds data set*) that contains the override embed parameters and values for the products that are installed in the respective CSI used to build the RTE. For the **MIGRATE** action, this library will contain the override embed parameters and values from the RTE being migrated from.

Unless specified otherwise in parameter **KFJ_EMBEDS_LIB**, the embeds data set name is *rte_plib_hilev.rte_name.EMBEDS*, where *rte_plib_hilev* is the high-level qualifier (HLQ) and *rte_name* is the name of the RTE specified on the action. By using this default naming convention, you can isolate the override embed values into the respective libraries per RTE.

Note: If the default or specified embeds data set exists when the **CREATE** or **MIGRATE** action runs, none of the existing members in it are changed or removed; only new members are added, which can occur if the CSI used to build the RTE has changed to add products.

During an initial **CREATE** or **MIGRATE** action when creating an RTE, when this parameter is set to Y, Monitoring Configuration Manager also defines the data set to the RTE using the **RTE_X_OVERRIDE_EMBEDS_LIB** parameter.

Required or optional

Optional

Default value

N

Permissible values

Y, N

Related parameters

KFJ_EMBEDS_LIB

RTE_X_OVERRIDE_EMBEDS_LIB

Target copy (TRG) parameters

The target copy parameters provide configuration settings for managing copies of SMP/E target libraries.

TRG_COPY_HILEV

Non-VSAM high-level qualifier for SMP/E target library copy

Description

This field specifies the high-level qualifier to be used when allocating the SMP/E target copy non-VSAM data sets. The names of these data sets will be generated by appending the appropriate suffix to this field.

Required or optional

Optional

Default value

%RTE_PLIB_HILEV%

TRG_COPY_MGMTCLAS

MGMTCLAS for non-VSAM libraries for SMP/E target library copy

Description

This field specifies the SMS management class to be used when allocating the SMP/E target copy non-VSAM data sets.

This field is required if the data sets are not to be SMS-managed. Leave this field blank if your installation does not use the SMS MGMTCLAS parameter or it is optional.

Required or optional

Optional

Default value

%RTE_SMS_MGTMCLAS%

TRG_COPY_NAME

SMP/E target library copy name

Description

This field specifies the name of the member in the RTEDEF library for SMP/E target copy settings. This member is required when using Configuration Manager to create and maintain an SMP/E target copy.

Required or optional

Optional

Default value

None

TRG_COPY_STORCLAS

STORCLAS for non-VSAM libraries for SMP/E target library copy

Description

This field specifies the SMS storage class to be used when allocating the SMP/E target copy non-VSAM data sets.

This field is required if the data sets are not to be SMS-managed. Leave this field blank if your installation does not use the SMS STORCLAS parameter or it is optional.

Required or optional

Optional

Default value

%RTE_SMS_STORCLAS%

TRG_COPY_TKANJAR_PATH

z/OS UNIX System Services directory for SMP/E target copy

Description

This field specifies the z/OS UNIX directory that will be used as a copy for the SMP/E installation jobs.

This field specifies the directory for a copy of SMP/E TKANJAR files, which is needed for OMEGAMON for CICS TG on z/OS (KGW) and OMEGAMON for JVM (KJJ).

TRG_COPY_TKAYHFS_PATH

This parameter specifies the z/OS UNIX System Services directory for the SMP/E TKAYHFS installation directory copy.

Description

This field specifies the z/OS UNIX directory that will be used as a copy for the SMP/E TKAYHFS installation directory.

This field specifies the directory for a copy of SMP/E TKAYHFS files, which is needed for OMEGAMON Data Provider (KAY).

Default value

`/var/rtehome/<trg_copy_name>/kay`

Permissible values

Character string, maximum length 44

TRG_COPY_UNIT

Unit for non-VSAM libraries for SMP/E target library copy

Description

This field specifies the unit name to be used when allocating the SMP/E target copy non-VSAM data sets.

This field is required if the data sets are not to be SMS-managed. Leave this field blank if your installation does not use the unit name or it is optional.

Required or optional

Optional

Default value

%RTE_SMS_UNIT%

TRG_COPY_VOLUME

VOLSER for non-VSAM libraries for SMP/E target library copy

Description

This field specifies the volume serial numbers to be used when allocating the SMP/E target copy non-VSAM data sets.

This field is required if the data sets are not to be SMS-managed. Leave this field blank if your installation does not use the volume serial number or it is optional.

Required or optional

Optional

Default value

%RTE_SMS_VOLUME%

Sparse parameter tables: The first row sets the default values for subsequent rows

For parameters that are arranged in tables, IBM Z® Monitoring Configuration Manager uses the first row to set the defaults for subsequent rows.

You can use this behavior to specify “sparse” parameter tables. You only need to specify parameters whose values differ from the first row. Sparse parameter tables are especially useful for configuring the Db2® monitoring agent.

As with PARMGEN, you can also specify complete parameter tables: every parameter in every row.

Example: Db2® profile configuration

The following example shows a sparse parameter table that configures three Db2® profiles (PROD, TEST, and DEV):

```

KD2_PF                                BEGIN

KD2_PF01_ROW                          01
KD2_PF01_PROFID                       "PROD"
* Configure and autostart Near-Term History (NTH)
KD2_PF01_HIS_START                     Y
* Store NTH data to VSAM data sets for e3270UI thread history
KD2_PF01_HIS_STORE                     THVSAM
KD2_PF01_THRDHIS_LOG_NUM               10
* Storage units
KD2_PF01_HIS_VSAM_SU                   CYLS
KD2_PF01_HIS_VSAM_MB                   50

KD2_PF02_ROW                          02
KD2_PF02_PROFID                       "TEST"
* TEST requires fewer resources for NTH than PROD
KD2_PF02_THRDHIS_LOG_NUM               3
KD2_PF02_HIS_VSAM_MB                   10

KD2_PF03_ROW                          03

```

```

KD2_PF03_PROFID          "DEV"
* No NTH in DEV
KD2_PF03_HIS_START      N
KD2_PF                  END

```

The first row configures the PROD profile and sets the default values for subsequent rows (profiles).

The TEST profile omits the following parameters, falling back to the values from the first row:

```

KD2_PFnn_HIS_START
KD2_PFnn_HIS_STORE
KD2_PFnn_HIS_VSAM_SU

```

The TEST profile sets different, lower values than PROD for the following parameters, because TEST requires fewer resources for Near-Term History:

```

KD2_PF02_THRDHIS_LOG_NUM
KD2_PF02_HIS_VSAM_MB

```

The DEV profile does not configure Near-Term History.

Example: Db2® subsystem configuration

The following example shows a sparse parameter table that configures the Db2® monitoring agent to monitor three Db2® subsystems (DB2P, DB2T, and DB2D):

```

KD2_DB                  BEGIN

KD2_DB01_ROW           01
KD2_DB01_DB2_SSID     "DB2P"
KD2_DB01_DB2_DESCRIPTION "PROD Db2 subsystem"
KD2_DB01_DB2_PROFID   "PROD"
KD2_DB01_DB2_VER      "12"
KD2_DB01_DB2_SYSNAME  "ZOSP"
KD2_DB01_DB2_DS_GROUP "N"
KD2_DB01_DB2_MONITOR_START "Y"
KD2_DB01_DB2_PORT_NUM "2000" * OMEGAMON Db2 PE Server TCP/IP port number
KD2_DB01_DB2_LOADLIB  "DSN.VCR1M0.SDSNLOAD"
KD2_DB01_DB2_DSNTIAD  "DSNTIAD"
KD2_DB01_DB2_RUNLIB   "DSN.VCR1M0.RUNLIB.LOAD"

KD2_DB02_ROW           02
KD2_DB02_DB2_SSID     "DB2T"
KD2_DB02_DB2_DESCRIPTION "TEST Db2 subsystem"
KD2_DB02_DB2_PROFID   "TEST"
KD2_DB02_DB2_SYSNAME  "ZOST"
KD2_DB02_DB2_PORT_NUM "2001"

KD2_DB03_ROW           03
KD2_DB03_DB2_SSID     "DB2D"
KD2_DB03_DB2_DESCRIPTION "DEVT Db2 subsystem"
KD2_DB03_DB2_PROFID   "DEVT"
KD2_DB03_DB2_SYSNAME  "ZOSD"
KD2_DB03_DB2_PORT_NUM "2002"
KD2_DB03_DB2_LOADLIB  "DSN.VC10.SDSNLOAD"
KD2_DB03_DB2_RUNLIB   "DSN.VC10.RUNLIB.LOAD"

KD2_DB                  END

```

The first row configures the agent to monitor the DB2P subsystem and sets the default values for subsequent rows (subsystems).

The second row falls back to the values set by the first row for Db2® version, LOADLIB, and RUNLIB, but sets its own values for the profile, system (LPAR) name, and port number.

The third row specifies its own values for LOADLIB and RUNLIB.

Runtime environment definition (RTEDEF) library

A runtime environment definition is a set of parameters. Parameters are stored in a runtime environment definition (RTEDEF) library. The set of parameters for each runtime environment is organized into several RTEDEF members.

The data set name of the RTEDEF library consists of the high-level qualifiers that you specify to IBM Z® Monitoring Configuration Manager in the **RTE_PLIB_HILEV** workflow variable in the KCIVARS input data set, followed by the fixed low-level qualifier RTEDEF:

```
rte_plib_hilev.RTEDEF
```

You can allocate the RTEDEF library yourself or you can use the **CREATE** action to allocate it and populate it with [initial members](#) for you.

An RTEDEF library can contain multiple runtime environment definitions. A single RTEDEF library can contain all of the runtime environment definitions for a sysplex. Or you can choose to store each runtime environment definition in a separate RTEDEF library.

If you are going to set up a High Availability TEMS (HA TEMS), make sure there is only one runtime environment defined in the RTEDEF (that is, the one used for the HA TEMS).

RTEDEF library members are also known as *configuration profile* members.

Runtime environment definition library members

RTEDEF library members follow a naming convention that identifies the contents of the member and whether the member applies to a specific LPAR or to all LPARs.

Use the naming convention described in the following table to store parameters in the correct members. In the LPARs column of the following table, *Current* means the LPAR on which the **GENERATE** action is performed.

Table 20: RTEDEF member naming convention			
Member name	Parameters	LPARs	Description
<i>rte_name</i>	RTE_* CONFIGURE_*	All	Runtime environment configuration profile. <i>rte_name</i> matches the value of the RTE_NAME workflow variable in the KCIVARS data set of the job that performs the Monitoring Configuration Manager action.
KC5@lpar KD5@lpar KI5@lpar KN3@lpar	Kpp_*	Current	LPAR-specific product configuration profile created by the DISCOVER action. Note: The DISCOVER action writes a comment member, <i>Kpp#lpar</i> , if a <i>Kpp@lpar</i> member already exists. For more information, see “Members created by the DISCOVER action” on page 43.
SYS@lpar	Symbols	Current	LPAR-specific system symbols and KCIPARSE extracted variables found by the DISCOVER action. Note: The DISCOVER action writes a comment member, <i>SYS#lpar</i> , if a <i>SYS@lpar</i> member already exists. For more information, see “Members created by the DISCOVER action” on page 43.

Member name	Parameters	LPARs	Description
Kpp\$PARM	Kpp_*	All	Product configuration profile, where <i>pp</i> is the product code. For the list of supported code values, see “Products supported by Configuration Manager” on page 8.
Kpp\$lpar	Kpp_*	Current	LPAR-specific product configuration profile, where <i>pp</i> is the product code. For the list of supported code values, see “Products supported by Configuration Manager” on page 8.
GBL\$PARM	GBL_*	All	Global configuration profile.
GBL\$lpar	GBL_*	Current	LPAR-specific global configuration profile.
VAR\$GLOB	Variables	All	Variables configuration profile. For more information, see “Variables in parameter values” on page 137.
VAR\$lpar	Variables	Current	LPAR-specific variables configuration profile. For more information, see “Variables in parameter values” on page 137.
PCK\$PARM	KFJ_LOCAL_*	All	Configuration profile created by the PACKAGE action that contains data set high-level qualifiers and other settings for deploying to a target system. For more information, see “Special considerations for SYSPLEX rollout” on page 124.
PCK\$lpar	KFJ_LOCAL_*	Current	LPAR-specific configuration profile created by the PACKAGE action that contains data set high-level qualifiers and other settings for deploying to a target system. For more information, see “Special considerations for SYSPLEX rollout” on page 124.
trg_copy_name	TRG_COPY_* GBL_* CONFIGURE_*	All	Target copy configuration profile. <i>trg_copy_name</i> matches the value of the TRG_COPY_NAME workflow variable in the KCIVARS DD statement of the job that performs the Monitoring Configuration Manager action with option TRGCOPY.

The following figure illustrates an example of the RTEDEF library member naming and hierarchy. Depending on your configuration, there might be more or fewer members in your RTEDEF. The complete list of members is provided in the previous table.

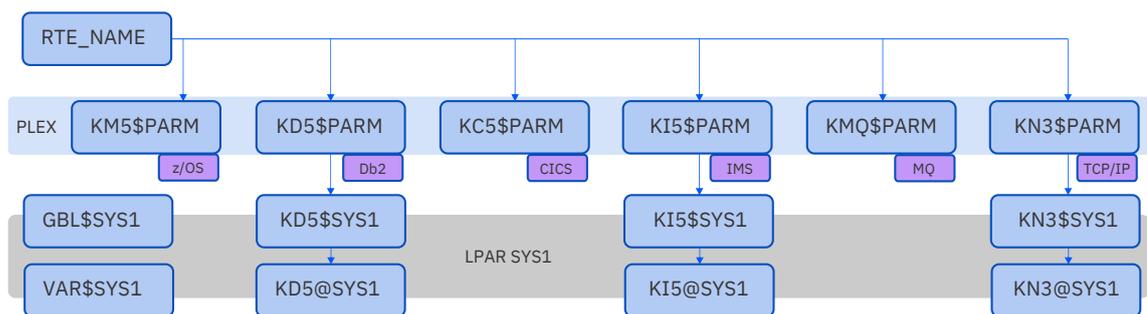


Figure 41: Example runtime environment definition library member naming and hierarchy

Note:

If a product has parameters with different prefixes, use the *Kpp* from the corresponding **CONFIGURE_*** parameter as the member name prefix for all the parameters. For example:

- Store all Db2® agent parameters, **KD2_*** and **KD5_***, in **KD5*** members, to match the **CONFIGURE_DB2_AGENT_KD5** parameter.

- Store all **KC2_*** and **KC5_*** parameters in KC5* members.
- Store all **KI2_*** and **KI5_*** parameters in KI5* members.
- Store all **KDF_*** and **KS3_*** parameters in KS3* members.

Concatenation order of runtime environment definition library members

The **GENERATE** action builds the set of parameters for a runtime environment by concatenating RTEDEF members in a well-defined order.

At a high level, the order in which the parameters are read can be categorized as follows:

- System symbols and KCIPARSE-extracted variables found by the **DISCOVER** action
- Target copy (if option **TRGCOPY** is specified)
- Variables (if system variables are enabled for the runtime environment)
- Runtime environment
- Product
- Global
- Package (if remote deployment applies)

The following table provides the specific order (by RTEDEF member) in which the parameters are read. In this list, parameters that are set in member `SYS@lpar` are read first and parameters that are set in member `PCK$lpar` are read last. In the following table, `lpar` identifies the LPAR on which the **GENERATE** action runs.

Important: If a parameter is set more than once, the last value that is read is used.

Order position	RTEDEF member	Notes
1	<code>SYS@lpar</code>	The <code>SYS@lpar</code> member is always processed first, even if the system variables flag (RTE_SYSV_SYSVAR_FLAG) is disabled.
2	<code>trg_copy_name</code>	The <code>trg_copy_name</code> member is processed only if option TRGCOPY is specified.
3	<code>VAR\$GLOB</code>	The <code>VAR\$GLOB</code> member is processed only if the system variables flag (RTE_SYSV_SYSVAR_FLAG) is enabled.
4	<code>VAR\$lpar</code>	The <code>VAR\$lpar</code> member is processed only if the system variables flag (RTE_SYSV_SYSVAR_FLAG) is enabled.
5	<code>rte_name</code>	
6	<code>Kpp@lpar</code>	For KC5, KI5, KD5, and KN3 only
7	<code>Kpp\$PARM</code>	
8	<code>Kpp\$lpar</code>	
9	<code>GBL\$PARM</code>	
10	<code>GBL\$lpar</code>	

Order position	RTEDEF member	Notes
11	PCK\$PARM	The PCK\$PARM member is processed only if the runtime environment is designed for <u>remote deployment</u> and parameter KFJ_LOCAL_PLIB_HILEV is specified in the KCIVARS DD statement.
12	PCK\$lpar	The PCK\$lpar member is processed only if the runtime environment is designed for <u>remote deployment</u> and parameter KFJ_LOCAL_PLIB_HILEV is specified in the KCIVARS DD statement.

Tip: After you run the **GENERATE** action, you can view an ordered list of the RTEDEF library members that the action uses. For more information, see “Parameter values used” on page 75.

Initial runtime environment library members

The **CREATE** action populates the RTEDEF library with an initial set of configuration profile members for a basic runtime environment.

There are hundreds of OMEGAMON parameters. However, to configure a runtime environment that uses basic functions, you only need to specify the few dozen parameters in these initial members. All other parameters use their default values.

Edit the members to specify appropriate parameters for your runtime environment.

Characteristics of the runtime environment defined by the initial members

The parameters in the initial members define a runtime environment with the following characteristics:

Characteristic	Defined by these parameters
Static hub monitoring server.	<p>KDS_TEMS_TYPE HUB</p> <p>Default parameter value (not specified in the initial set of parameters):</p> <p>KDS_TEMS_HA_TYPE none</p>
All <i>installed</i> components configured. The CREATE action detects which component products, such as monitoring agents, are installed and sets the corresponding CONFIGURE_* parameters to Y.	<p>CONFIGURE_* Y</p>
Runtime members shared with target libraries. Rather than being a full stand-alone set, the runtime members refer to some SMP/E installation target libraries (or, more typically, a copy that you have created of those target libraries).	<p>Default parameter values (not specified in the initial set of parameters):</p> <p><u>“RTE_TYPE” on page 91</u> SHARING RTE_SHARE SMP</p>

The initial members include only a few product-specific members

Only a few component products require you to specify parameter values for their basic functions. Most work out-of-the-box using default parameter values.

The **CREATE** action creates *Kpp*\$PARM members only for the following components, and only if they are installed:

Monitoring server (KDS\$PARM)
 CICS® TG monitoring agent (KGW\$PARM)

The **CREATE** action creates these members based only on which components are installed. The **CREATE** action is not sensitive to **CONFIGURE_*** parameters in an existing RTEDEF (*rte_name*) member. The **CREATE** action neither reads nor overwrites existing RTEDEF members. For example, if the CICS® TG monitoring agent is installed in the target libraries, then the **CREATE** action creates an RTEDEF (KGW\$PARM) member, even if an RTEDEF (*rte_name*) member already exists and specifies **CONFIGURE_CICS_TG_KGW**.

To set parameter values for other products, you must create the corresponding product-specific *Kpp** members.

Special considerations for SYSPLEX rollout

Specify an additional parameter, **KFJ_LOCAL_PLIB_HILEV**, in your KCIVARS DD statement when running the **CREATE** action to have the RTEDEF created using the value in **KFJ_LOCAL_PLIB_HILEV**.

When this parameter is specified, the generated *kfj_local_plib_hilev*.RTEDEF data set will contain an additional member called PCK\$PARM. This member allows locally generated runtime environment data sets to have a different high-level qualifier than the data sets that will be used on the deployment (target) system.

Just like for other members in the RTEDEF, there is a SYSPLEX-wide member called RTEDEF (PCK\$PARM) and an LPAR-specific member, RTEDEF (PCK\$1par), supported. The **CREATE** action will generate the RTEDEF (PCK\$PARM) member.

If any of the parameters in PCK\$PARM are not defined, Configuration Manager will use the **KFJ_LOCAL_PLIB_HILEV** parameter to generate default values for all of the high-level qualifier parameters. For the remaining parameters, the target runtime environment parameter will be used as the default.

The target runtime environment parameter will be used to allocate the respective data sets.

For more information about the PCK\$PARM member, see “RTEDEF(PCK\$PARM)” on page 126.

Using an SMP/E target copy

Specifying the **CREATE** action with option **TRGCOPY** creates member RTEDEF (*trg_copy_name*). This member contains source and destination information used to create and maintain a copy of your SMP/E target libraries. This member is used only when the **CREATE** or **GENERATE** action is run using option **TRGCOPY**. For more information, see “RTEDEF(*trg_copy_name*)” on page 128 and “Using SMP/E target library copies” on page 156.

RTEDEF(*rte_name*)

The RTEDEF (*rte_name*) member is the runtime environment configuration profile. This member contains parameters with the prefixes **RTE** and **CONFIGURE**.

Figure 42: Initial RTEDEF (*rte_name*) member created by the **CREATE** action

<u>RTE_NAME</u>	<rte_name>
<u>RTE_PLIB_HILEV</u>	<rte_plib_hilev>
<u>RTE_SECURITY_USER_LOGON</u>	NONE
<u>RTE_SECURITY_FOLD_PASSWORD_FLAG</u>	Y
<u>RTE_SECURITY_CLASS</u>	""
<u>RTE_X_SECURITY_EXIT_LIB</u>	<rte_plib_hilev>.<rte_name>.SECEXITS
<u>RTE_TEMS_NAME_NODEID</u>	<rte_name>:TEMS
<u>RTE_TEMS_TRANSPORT_MODE</u>	HTTPS
<u>RTE_COMM_PROTOCOLn</u>	IPPIPE
<u>RTE_TCP_HOST</u>	<sysiphostname>
<u>RTE_TCP_PORT_NUM</u>	1918
<u>RTE_VTAM_APPLID_PREFIX</u>	OM<sysclone>
<u>RTE_STC_PREFIX</u>	OMEG
<u>RTE_USS_RTEDIR</u>	"/var/rtehome"
<u>CONFIGURE_TEMS_KDS</u>	Y * TEMS
<u>CONFIGURE_E3270UI_KOB</u>	Y * Enhanced 3270
<u>CONFIGURE_CICS_KC5</u>	Y * CICS TS
<u>CONFIGURE_CICS_TG_KGW</u>	Y * CICS TG
<u>CONFIGURE_DB2_AGENT_KD5</u>	Y * Db2
<u>CONFIGURE_IMS_KI5</u>	Y * IMS
<u>CONFIGURE_JVM_KJJ</u>	Y * JVM
<u>CONFIGURE_ZOS_KM5</u>	Y * z/OS
<u>CONFIGURE_MESSAGING_KMQ</u>	Y * MQ
<u>CONFIGURE_MESSAGING_KQI</u>	Y * Integration Bus
<u>CONFIGURE_NETVIEW_KNA</u>	Y * Netview
<u>CONFIGURE_MFN_KN3</u>	Y * Network

CONFIGURE_STORAGE_KS3	Y	* Storage
CONFIGURE_OMEGAVIEW_KWO	Y	* Integration Monitor
CONFIGURE_ITCAMAD_KYN	Y	* ITCAM for Applications
CONFIGURE_ACM_KRN	Y	* Advanced Catalog Mgmt
CONFIGURE_ARD_KRH	Y	* Advanced Rpt and Mgmt
CONFIGURE_AAD_KRG	Y	* Advanced Audit
CONFIGURE_AAM_KRJ	Y	* Advanced Alloc Mgmt
CONFIGURE_ATAM_KRK	Y	* Automated Tape Alloc
CONFIGURE_ABR_KRV	Y	* Advanced Backup and Rec
CONFIGURE_ODP_KAY	Y	* Data Provider

Note: `RTE_X_OVERRIDE_EMBEDS_LIB` is included if you specify `KFJ_USE_EMBEDS` set to Y.

Non-default values

The values of the following parameters in this initial member are different than the default values:

RTE_TEMS_NAME_NODEID

This parameter sets the node ID of the monitoring server that is configured in this runtime environment. The only difference between the value in this initial member and the default value: the default value ends in CMS rather than TEMS. Given the related parameter names, TEMS is a more intuitive value.

RTE_VTAM_APPLID_PREFIX

LPARs in a sysplex might have their own instances of the same VTAM® application. The fixed default VTAM® applid prefix, CTD, does not help to identify the LPAR to which each instance belongs. This initial member sets the value to `OM<sysclone>`, where:

- `OM` is an abbreviation of OMEGAMON®.
- `<sysclone>` is the 1- or 2-character value of the z/OS® static system symbol `SYSCZONE`. `SYSCZONE` is shorthand notation for the system (LPAR) name.

RTE_STC_PREFIX

The default started task prefix is IBM.

This initial member sets the value to `OMEG` (an abbreviation of OMEGAMON®).

Significant default values

Some characteristics of the runtime environment configured using this initial member are determined by the following default parameter values:

RTE_TYPE	SHARING
RTE_SHARE	SMP

RTEDEF (KDS\$PARM)

The RTEDEF (KDS\$PARM) member contains parameters that configure the monitoring server. These parameters have the prefix **KDS**.

Figure 43: Initial RTEDEF (KDS\$PARM) member created by the CREATE action

```
* Tivoli Enterprise Monitoring Server
KDS_TEMS_TYPE      HUB
```

RTEDEF (KGW\$PARM)

The RTEDEF (KDS\$PARM) member contains parameters that configure the CICS® Transaction Gateway monitoring agent. These parameters have the prefix **KGW**.

Figure 44: Initial RTEDEF (KGW\$PARM) member created by the CREATE action

```
* CICS Transaction Gateway
KGW_SA              BEGIN                * Table begin *
```

```

KGW_SA01_ROW                01
KGW_SA01_CTG_DAEMON_STC    CTGPROC * Sample CTG Daemon *
KGW_SA01_CTG_DAEMON_PORT_NUM 2980 * 00000-65535
KGW_SA01_CTG_DAEMON_HOST    LOCALHOST
KGW_SA01_SAPI_CLIENT_CTGTRACE 0 * 0-4
* END KGW_SA01 row 1 (add more rows as needed!)
KGW_SA                      END * Table end *

```

RTEDEF (GBL\$PARM)

The RTEDEF (GBL\$PARM) member is the global configuration profile. This member contains global parameters. These parameters have the prefix **GBL**.

The **CREATE** action only populates the RTEDEF (GBL\$PARM) member with parameters that are relevant to the monitoring agents that are installed. For example:

Figure 45: Initial RTEDEF (GBL\$PARM) member created by the CREATE action

```

* Global parameters (used by installed products)

* High-level qualifier of SMP/E target libraries
GBL_TARGET_HILEV           "<HLQs of KCIFLOW dsname of CREATE action>"

* Java home directory (KYN, KDS)
GBL_HFS_JAVA_DIR1          "/usr/lpp/java/IBM/J8.0_64"

* SMP/E target directory containing TKANJAR files (KGW, KJJ)
GBL_USS_TKANJAR_PATH       "/usr/lpp/kan/bin/IBM"

* SMP/E target directory containing TKAYHFS files (KAY)
GBL_USS_TKAYHFS_PATH       "/usr/lpp/omdp"

* ICSF load library containing CSNB* modules for password encryption
* (KS3, KI5)
GBL_DSN_CSF_SCSFMODE0     "CSF.SCSFMODE0"

* Db2 libraries (KD5)
GBL_DSN_DB2_RUNLIB_V12    "DSN.VCR1M0.RUNLIB.LOAD"
GBL_DSN_DB2_LOADLIB_V12   "DSN.VCR1M0.SDSNLOAD"
GBL_DSN_DB2_DSNEEXIT      "DSN.VCR1M0.DSNEEXIT"

* CICS Transaction Gateway (KGW)
GBL_DSN_CICS_CTG_DLL      "SYS1.SCTGDLL"

* IMS libraries (KI5)
GBL_DSN_IMS_RESLIB        "IMS.SDFSRESL"
GBL_DSN_IMS_SCEXLINK      "IMS.SCEXLINK"
GBL_DSN_IMS_SFUNLINK      "IMS.SFUNLINK"

* MQ and Integration Broker (KMQ, KQI)
GBL_DSN_WMQ_SCSQANLE      "CSQ.V9R0M0.SCSQANLE"
GBL_DSN_WMQ_SCSQAUTH      "CSQ.V9R0M0.SCSQAUTH"

* Netview CNMLINK library
GBL_DSN_NETVIEW_CNMLINK   "NETVIEW.VNRNMN.CNMLINK"

```

GBL_TARGET_HILEV

The **CREATE** action sets the **GBL_TARGET_HILEV** parameter to the high-level qualifiers of the data set name specified by the KCIFLOW DD statement in the job step that performs the **CREATE** action. Combined with the default parameter values **RTE_TYPE** SHARING and **RTE_SHARE** SMP, this value configures the runtime environment to share the same target library that was used to perform the **CREATE** action.

RTEDEF (PCK\$PARM)

The RTEDEF (PCK\$PARM) member contains parameters that allow remote deployment of a runtime environment.

This member essentially specifies a mapping. A shadow value is specified that represents the value used to generate the runtime environment data sets on the local system (that is, the configuration LPAR). Note that this is the only place this mapping is done. All other values in the RTEDEF members should be specified using the

attributes and high-level qualifiers of the target system (that is, the system that the runtime environment will be deployed to).

The list of ***_HILEV** parameters, along with **RTE_USS_RTEDIR** and **GBL_USS_TKANJAR_PATH**, are the only parameters supported for this mapping.

Note: If the **GENERATE** action detects that non-supported high-level qualifiers are being used in the RTEDEF, an error message will be displayed in KCIPRINT indicating incompatible parameters, and the **GENERATE** action workflow stops.

The RTEDEF (PCK\$PARM) member contains the values described in the following table:

<i>Table 23: Parameters in RTEDEF(PCK\$PARM) member</i>		
Parameter in member PCK\$PARM or PCK\$lpar	Target runtime environment parameter mapped in RTEDEF (rte_name)	Description
KFJ_LOCAL_PLIB_HILEV	RTE_PLIB_HILEV	
KFJ_LOCAL_HILEV	RTE_HILEV	
KFJ_LOCAL_VSAM_HILEV	RTE_VSAM_HILEV	Default VSAM high-level qualifier to be used when allocating VSAM data sets
KFJ_LOCAL_PDS_HILEV	RTE_PDS_HILEV	Default PDS high-level qualifier to be used to allocate PDS V1 data sets (if needed)
KFJ_LOCAL_TARGET_HILEV	GBL_TARGET_HILEV	SMP/E target high-level qualifier found in RTEDEF (GBL\$PARM/GBL\$lpar)
KFJ_LOCAL_USS_RTEDIR	RTE_USS_RTEDIR	Runtime environment z/OS® UNIX® System Services root path used by Configuration Manager to save all z/OS® UNIX®-related artifacts
KFJ_LOCAL_USS_TKANJAR_PATH	GBL_USS_TKANJAR_PATH	SMP/E target directory containing TKANJAR files
KFJ_LOCAL_USS_TKAYHFS_PATH	GBL_USS_TKAYHFS_PATH	SMP/E TKAYHFS ddname install directory
KFJ_LOCAL_SMS_VOLUME	RTE_SMS_VOLUME	To support non-SMS environments
KFJ_LOCAL_SMS_VSAM_VOLUME	RTE_SMS_VSAM_VOLUME	To support non-SMS environments
KFJ_LOCAL_SMS_UNIT	RTE_SMS_UNIT	To support non-SMS environments
KFJ_LOCAL_SMS_STORCLAS	RTE_SMS_STORCLAS	
KFJ_LOCAL_SMS_VSAM_STORCLAS	RTE_SMS_VSAM_STORCLAS	
KFJ_LOCAL_SMS_MGMTCLAS	RTE_SMS_MGMTCLAS	
KFJ_LOCAL_SMS_VSAM_MGMTCLAS	RTE_SMS_VSAM_MGMTCLAS	

Sample member RTEDEF(PCK\$PARM)

The sample member, RTEDEF (PCK\$PARM), is shown below.

Figure 46: Example of PCK\$PARM

```

* Local RTE parameters

KFJ_LOCAL_PLIB_HILEV          TSQUID.LOCAL
KFJ_LOCAL_HILEV
KFJ_LOCAL_VSAM_HILEV
KFJ_LOCAL_PDS_HILEV

* High-level qualifier of local SMP/E target libraries
KFJ_LOCAL_TARGET_HILEV       "MONSUITE"

* Path to local z/OS UNIX files
KFJ_LOCAL_USS_RTEDIR         "/var/rtehome"

* Local SMP/E target directory containing TKANJAR files (KGW, KJJ)
KFJ_LOCAL_USS_TKANJAR_PATH   "/usr/lpp/kan/bin/IBM"

* Local SMP/E directory containing TKAYHFS files (KAY)
KFJ_LOCAL_USS_TKAYHFS_PATH   "/usr/lpp/omdp"

* SMS Parameters
KFJ_LOCAL_SMS_VOLUME
KFJ_LOCAL_SMS_VSAM_VOLUME
KFJ_LOCAL_SMS_UNIT
KFJ_LOCAL_SMS_STORCLAS
KFJ_LOCAL_SMS_VSAM_STORCLAS
KFJ_LOCAL_SMS_MGMTCLAS
KFJ_LOCAL_SMS_VSAM_MGMTCLAS

```

Example

The **RTE_PLIB_HILEV** value in RTEDEF (MYRTE) is as follows:

```

RTE_PLIB_HILEV                TDCIT.REG

```

Specifying **KFJ_LOCAL_PLIB_HILEV** with a value of SYS3.PREGEN in RTEDEF (PCK\$PARM) will allocate the data sets that will use a value of **RTE_PLIB_HILEV** using SYS3.PREGEN instead of TDCIT.REG. When using the **DEPLOY** action after having run the **PACKAGE** action and transferring the dump data sets to the target system, the original value as specified in RTEDEF (MYRTE), that is TDCIT.REG in this specific example, will be used to restore the data sets.

RTEDEF(*trg_copy_name*)

The RTEDEF(*trg_copy_name*) member is the SMP/E target copy configuration profile. This member contains parameters with the prefixes **GBL**, **CONFIGURE**, and **TRG**.

Figure 47: Initial RTEDEF(*trg_copy_name*) member created by the **CREATE** action with option **TRGCOPY**

```

* High-level qualifier of SMP/E target libraries
GBL_TARGET_HILEV              "<HLQs of KCIFLOW dsname of CREATE action>"

* SMP/E target directory containing TKANJAR files (KGW, KJJ)
GBL_USS_TKANJAR_PATH          "/usr/lpp/kan/bin/IBM"

TRG_COPY_NAME                  <trg_copy_name>
* High-level qualifier of the copy of SMP/E target libraries
TRG_COPY_HILEV                <trg_copy_hilev>
* Directory for a copy of SMP/E TKANJAR files (KGW, KJJ)
TRG_COPY_TKANJAR_PATH         "/var/rtehome/<trg_copy_name>/kan/bin/IBM"
* Directory for a copy of SMP/E TKAYHFS files (KAY)
TRG_COPY_TKAYHFS_PATH         "/var/rtehome/<trg_copy_name>/kay"

CONFIGURE_TEMS_KDS             Y * TEMS
CONFIGURE_E3270UI_KOB         Y * Enhanced 3270
CONFIGURE_CICS_KC5            Y * CICS TS
CONFIGURE_CICS_TG_KGW         Y * CICS TG
CONFIGURE_DB2_AGENT_KD5       Y * Db2
CONFIGURE_IMS_KI5             Y * IMS
CONFIGURE_JVM_KJJ             Y * JVM
CONFIGURE_ZOS_KM5             Y * z/OS
CONFIGURE_MESSAGING_KMQ       Y * MQ
CONFIGURE_MESSAGING_KQI       Y * Integration Bus
CONFIGURE_NETVIEW_KNA         Y * Netview
CONFIGURE_MFN_KN3             Y * Network

```

CONFIGURE_STORAGE_KS3	Y	* Storage
CONFIGURE_OMEGAVIEW_KWO	Y	* Integration Monitor
CONFIGURE_ITCAMAD_KYN	Y	* ITCAM for Applications
CONFIGURE_ACM_KRN	Y	* Advanced Catalog Mgmt
CONFIGURE_ARD_KRH	Y	* Advanced Rpt and Mgmt
CONFIGURE_AAD_KRG	Y	* Advanced Audit
CONFIGURE_AAM_KRJ	Y	* Advanced Alloc Mgmt
CONFIGURE_ATAM_KRK	Y	* Automated Tape Alloc
CONFIGURE_ABR_KRV	Y	* Advanced Backup and Rec
CONFIGURE_ODP_KAY	Y	* Data Provider

GBL_*

The source location from where a copy is taken

TRG_COPY_*

The target destination where the files are copied to

CONFIGURE_*

Products for which configuration files will be copied from the source (**GBL_***) to the destination (**TRG_COPY_***)

Additional TRG_* parameters

Target copy libraries are always non-VSAM. You can also include the following parameters in your SMP/E target copy member when allocating SMP/E target copy libraries:

TRG_COPY_VOLUME

Specifies the VOLSER for target copy non-VSAM libraries.

TRG_COPY_UNIT

Specifies the unit name for target copy non-VSAM libraries.

TRG_COPY_STORCLAS

Specifies the STORCLAS for target copy non-VSAM libraries.

TRG_COPY_MGMTCLAS

Specifies the MGMTCLAS for target copy non-VSAM libraries.

Runtime members

The **GENERATE** action generates runtime members in locations (MVS™ data set names and z/OS® UNIX® System Services paths) that are specified by parameters.

Runtime member locations

MVS™ data sets:

```
rte_hilev.rte_name.RK*  
rte_hilev.rte_name.ssid.RK*  
rte_vsam_hilev.rte_name.RK*  
rte_vsam_hilev.rte_name.ssid.RK*  
gbl_dsn_sys1_proclib  
gbl_dsn_sys1_vtamlib  
gbl_dsn_sys1_vtamlst
```

z/OS® UNIX® directory paths:

```
rte_uss_dir/rte_name/*
```

where:

- *ssid* is the identifier of a subsystem to be monitored, such as a Db2® subsystem or an IMS subsystem
- Other identifiers in *italics* represent parameter values

Parameters that specify runtime member locations

RTE_HILEV

High-level qualifiers of non-VSAM data sets.

RTE_VSAM_HILEV

High-level qualifiers of VSAM data sets.

RTE_HILEV and **RTE_VSAM_HILEV** have the same default value: *rte_plib_hilev*

GBL_DSN_SYS1_PROCLIB

GBL_DSN_SYS1_VTAMLIB

GBL_DSN_SYS1_VTAMLST

The data set names where the **GENERATE** action writes members intended for your system libraries.

Default values:

```
rte_hilev.SYS1.PROCLIB  
rte_hilev.SYS1.VTAMLIB  
rte_hilev.SYS1.VTAMLST
```

You must use your own site-specific procedures to copy the members from these locations to your actual PROCLIB, VTAMLIB, and VTAMLST system libraries.

RTE_USS_RTEDIR

z/OS® UNIX® directory. Default value: /var/rtehome

RTE_NAME

The runtime environment name is appended to **RTE_HILEV** and **RTE_VSAM_HILEV** as a low-level qualifier and to **RTE_USS_RTEDIR** as a directory name.

The following diagram illustrates the relationships between parameters and runtime member locations:

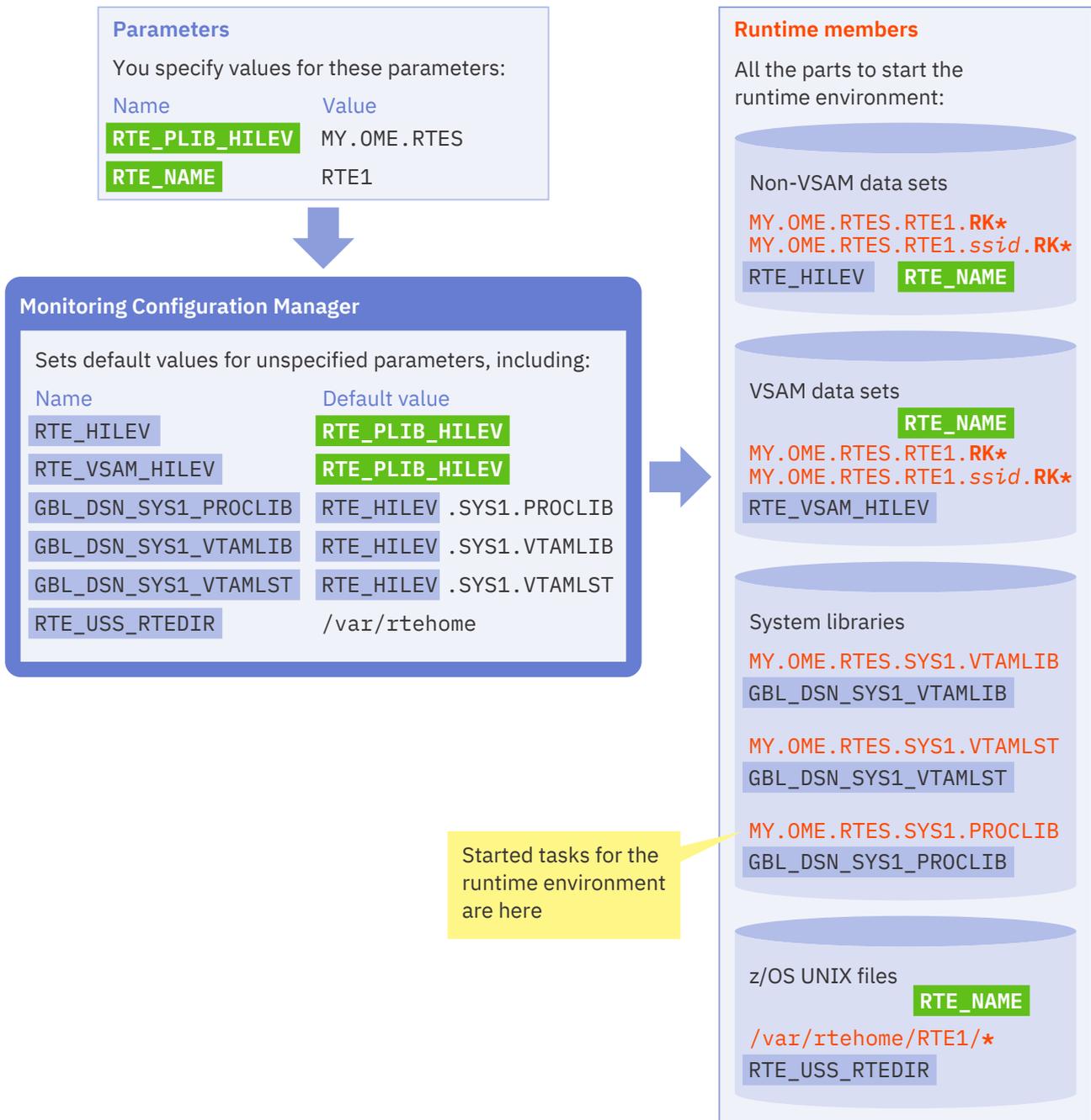


Figure 48: How parameters affect the locations of runtime members

Tip: Monitoring Configuration Manager writes concise started tasks to:
`rte_hilev.SYS1.PROCLIB`
 and versions with verbose comments to the same location used by PARMGEN:

```

rte_plib_hilev.rte_name.RKANSAMU
rte_plib_hilev.rte_name.RKD2SAM (for Db2®)
  
```

Communication between monitoring components

In a typical topology, monitoring agents communicate with remote monitoring servers, and remote monitoring servers communicate with a single, central hub monitoring server.

Typical topology

The following diagram shows a simple example.

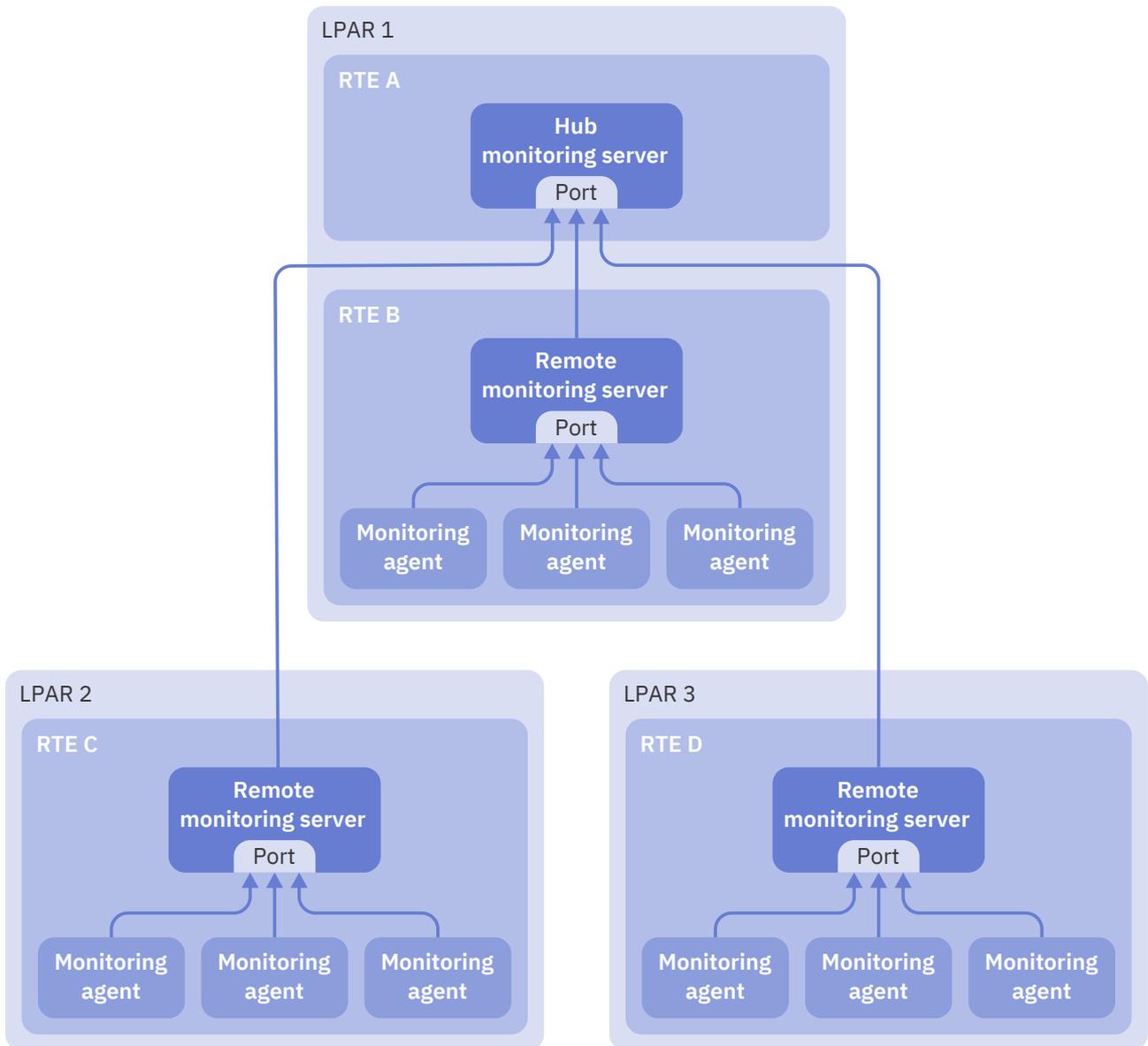


Figure 49: Typical topology of runtime environments in a sysplex

The number of monitoring agents in a runtime environment and the number of LPARs depends on your site.

In a typical topology, monitoring agents communicate with the remote monitoring server that is in the same runtime environment as the agents. Remote monitoring servers are described as *remote* to distinguish them from the *hub* monitoring server. Remote monitoring servers are typically *local* to the monitoring agents with which they communicate, in the sense that they are in the same runtime environment and, hence, the same LPAR.

The default communication protocol for all components is Transport Control Protocol over Internet Protocol version 4 (TCP/IPv4). The hub monitoring server listens on a port for messages from remote monitoring servers. Remote monitoring servers listen on a port for messages from monitoring agents.

Required parameters

Communication between components involves several parameters. However, for most of these parameters you can use default values.

To configure communication between a remote monitoring server and a hub server, you only need to specify the following parameters:

- In the runtime environment that contains the hub monitoring server (in the previous figure, RTE A):

KDS_TEMS_TYPE

HUB, rather than the default REMOTE.

RTE_TCP_PORT_NUM

The port on which the hub listens for messages from remote monitoring servers.

- In a runtime environment that contains a remote monitoring server (such as RTE D):

KDS_HUB_TEMS_NAME_NODEID

Must match the **RTE_TEMS_NAME_NODEID** parameter of the hub. The default value of **RTE_TEMS_NAME_NODEID** is *rte_name:CMS*.

KDS_TCP_PORT_NUM

Must match the **RTE_TCP_PORT_NUM** parameter of the hub.

KDS_HUB_TCP_HOST

Must match the host name or IP address of the LPAR that contains the hub.

You don't need to specify any parameters to configure communication between monitoring agents and a remote monitoring server in the same runtime environment.

The following diagram shows the required parameters in each runtime environment and their relationships:

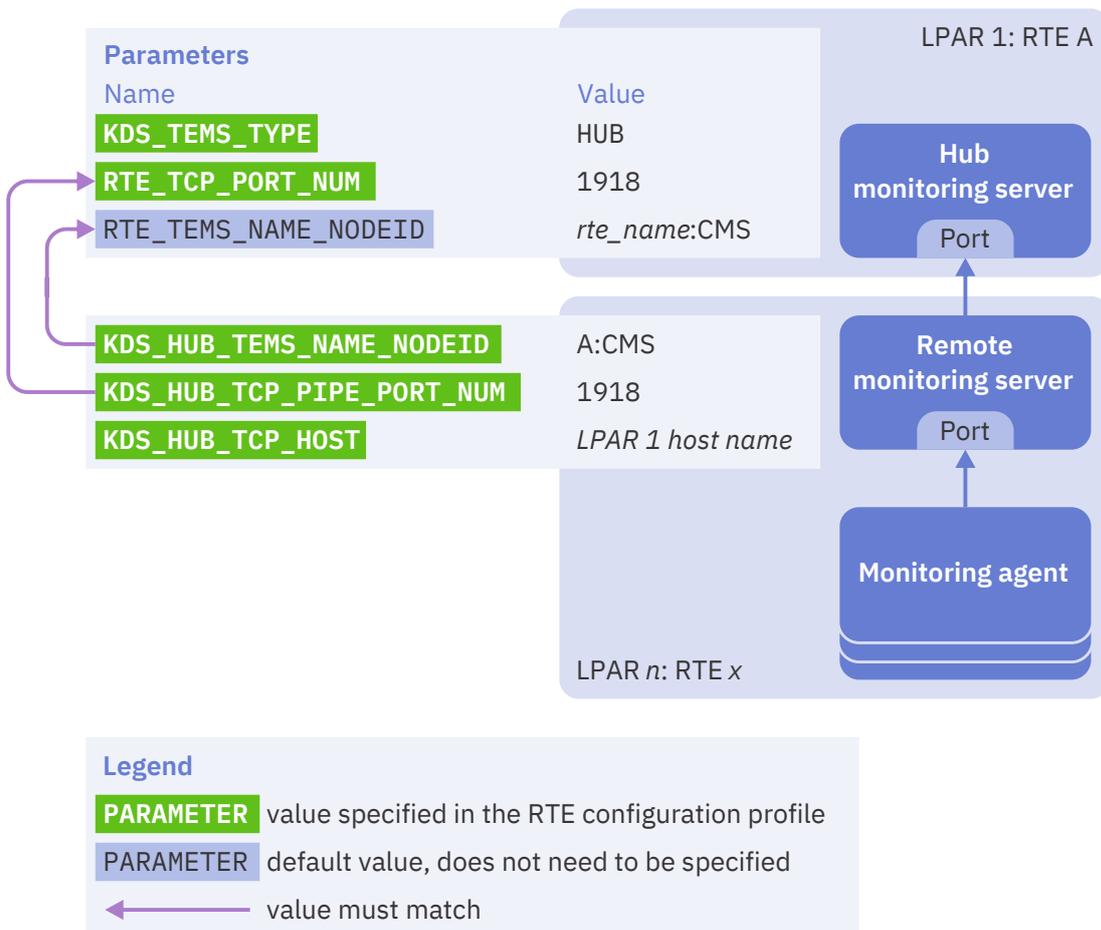


Figure 50: Parameters required to configure a typical topology

Other parameters

The following diagram shows a more comprehensive overview of the parameters to configure communication between components, including the default values of parameters omitted from the previous diagram.

For a typical topology, you do not need to be aware of these other parameters. This diagram is provided as a reference for configuring different topologies.

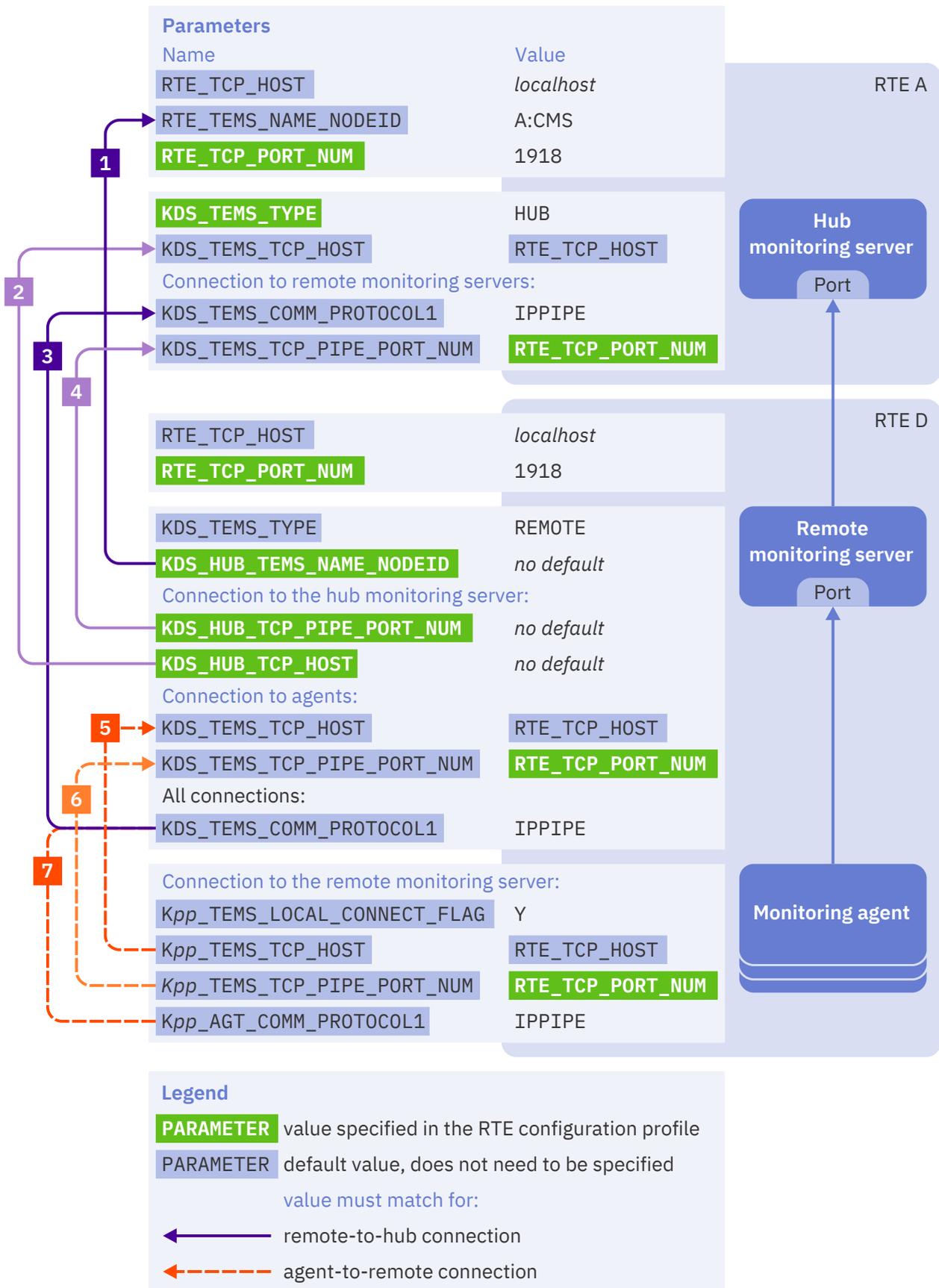


Figure 51: Parameters to configure communication between components, including significant default values

Communication between a remote monitoring server and the hub monitoring server

For a remote monitoring server to communicate with the hub monitoring server, the following parameters must match:

Figure label	This parameter in the RTE of the remote monitoring server	...must match this parameter in the RTE of the hub
1	KDS_HUB_TEMS_NAME_NODEID The remote monitoring server must refer to the node ID of the hub.	RTE_TEMS_NAME_NODEID
2	KDS_HUB_TCP_HOST The remote monitoring server must refer to the host name of the hub.	KDS_TEMS_TCP_HOST
3	KDS_TEMS_COMM_PROTOCOLn The remote monitoring server and hub must have at least one communication protocol in common.	KDS_TEMS_COMM_PROTOCOLn
4	KDS_HUB_TCP_PIPE_PORT_NUM The remote monitoring server must refer to the port on which the hub is listening.	KDS_TEMS_TCP_PIPE_PORT_NUM

Communication between monitoring agents and a remote monitoring server

For monitoring agents to communicate with a remote monitoring server, the following parameters must match:

Figure label	This parameter for the monitoring agent...	...must match this parameter for the remote monitoring server
5	Kpp_TEMS_TCP_HOST	KDS_TEMS_TCP_HOST
6	Kpp_TEMS_TCP_PIPE_PORT_NUM	KDS_TEMS_TCP_PIPE_PORT_NUM
7	Kpp_AGT_COMM_PROTOCOLn	KDS_TEMS_COMM_PROTOCOLn

If monitoring agents communicate with the remote monitoring server that is in the same runtime environment, as specified by the default parameter value **Kpp_TEMS_LOCAL_CONNECT_FLAG** Y, then all of these parameters match by default.

Choice of communication protocols

You can specify up to seven communication protocols for each monitoring agent and server. When attempting to communicate, a monitoring agent or server tries its protocols in order. If the first choice fails, it tries the second choice, and so on. You can either set the communication protocol choices individually for the monitoring server and each agent in a runtime environment, or you can use the **RTE_COMM_PROTOCOLn** parameters to set them all together.

Variables in parameter values

Many parameter values can optionally refer to variables.

IBM Z® Monitoring Configuration Manager supports the same variables as PARMGEN.

Tip: Instead of using variables, consider using LPAR-specific [RTEDEF members](#).

Variables versus LPAR-specific RTEDEF members

Variables enable you to reuse a configuration profile member for different LPARs where LPARs require different parameter values.

However, using variables adds a precursor step to runtime environment started tasks. The step resolves variable values. The additional processing delays runtime environment startup.

LPAR-specific RTEDEF members, introduced by Monitoring Configuration Manager, offer an alternative to using variables for LPAR-specific parameter values.

Using LPAR-specific RTEDEF members instead of variables removes the variable-resolution precursor step from started tasks.

If you use LPAR-specific RTEDEF members instead of variables, started tasks are simpler and runtime environments start faster.

Using variables

To use variables, you must set the **RTE_SYSV_SYSVAR_FLAG** parameter to Y.

Variables, like parameters, are defined using name-value pairs and are stored in members of the RTEDEF library.

The following example, without variables, sets the parameter named **RTE_TCP_PORT_NUM** to the literal value 1918:

```
RTE_TCP_PORT_NUM 1918
```

The following example sets the parameter to the value of the variable **RTE_PORT**:

```
RTE_TCP_PORT_NUM &RTE_PORT.
```

Suppose your sysplex contains two LPARs: ZOS1 and ZOS2. In general, these LPARs have similar runtime environment configurations. However, on ZOS1 you want the monitoring server to listen on port 1918, whereas on ZOS2 you want the monitoring server to listen on port 1919.

In the variables configuration profile member for LPAR ZOS1, VAR\$ZOS1, you set the **RTE_PORT** variable to 1918:

```
RTE_PORT 1918
```

In VAR\$ZOS2, you set **RTE_PORT** to 1919:

```
RTE_PORT 1919
```

RTEDEF members that define variables

In the LPARs column of the following table, *Current* means: the LPAR on which the **GENERATE** action is performed.

Table 26: RTEDEF members that define variables, and the LPARs to which they apply		
Member name	LPARs	Description
VAR\$GLOB	All	Variables configuration profile.
VAR\$lpar	Current	LPAR-specific variables configuration profile.

If a variable is defined in both VAR\$GLOB and VAR\$lpar, then the value in VAR\$lpar is used.

Unique variable names

While Configuration Manager supports runtime environments with variables (parameter RTE_SYSV_SYSVAR_FLAG is set to Y), it does not support cases where the parameter value is a variable with the same name as the parameter itself.

For example, Configuration Manager does not allow the following parameter setting:

```
RTE_USS_RTEDIR &RTE_USS_RTEDIR
```

In this case, you must change the name of the variable to be different from the parameter name, as shown in the following example where the prefix MY_ has been added to the variable name:

```
RTE_USS_RTEDIR &MY_RTE_USS_RTEDIR
```

Variables migrated from PARMGEN

Unlike Configuration Manager, PARMGEN does allow the name of a variable to be the same as the parameter to which it is assigned. If any of these same-name settings exist in your PARMGEN parameter definitions, the variables must be renamed for use by Configuration Manager.

When you migrate configuration settings from PARMGEN to Configuration Manager using the **MIGRATE** action, the **MIGRATE** action automatically rectifies any such incompatible settings, as follows:

- For variable names that are 30 characters or less, the **MIGRATE** action renames the variable by adding the following suffix to the name: `_R`
For example, PARMGEN allows the following parameter setting, which is defined in member WCONFIG(*rte_name*):

```
RTE_USS_RTEDIR &RTE_USS_RTEDIR
```

The value of the variable is set in member GBL_USER_JCL(*rte_name*), as follows:

```
RTE_USS_RTEDIR "value"
```

When the configuration settings are imported from PARMGEN to Configuration Manager, the **MIGRATE** action renames the variable, as follows:

- In member RTEDEF(*rte_name*):

```
RTE_USS_RTEDIR &RTE_USS_RTEDIR_R
```

- In member RTEDEF(VAR\$GLOB):

```
RTE_USS_RTEDIR_R "value"
```

- If the variable name is 31 characters, the **MIGRATE** action only adds the following suffix: `_`
- If the variable name is 32 characters (the maximum length), the **MIGRATE** action only adds a comment with a warning in the respective RTEDEF(VAR\$GLOB) member, indicating that you must rename the variable.

Note: To rename your variables to names other than those provided by the default naming convention of the **MIGRATE** action, you must manually rename the variables either before or after you run the **MIGRATE** action.

Setting up security exits in your runtime environment

Security exits are required for your runtime environment. You can use the **CREATE** and **MIGRATE** actions to set up your security exits library, and use the **GENERATE** action to create the necessary runtime members.

You must set up a library for your runtime environment that contains the OMEGAMON and IBM Tivoli Monitoring-related product security exits (such as KOBSPDPT OMEGAMON KppSUPDI exits, Tivoli Monitoring Services: Engine security exits, and external security exits).

The following points provide an overview of the configuration that is required in Configuration Manager for security exits in your runtime environment:

- A dedicated library must be allocated and populated with the security exits. The default name for the security exits library is `rte_plib_hilev.rte_name`.SECEXITS. (You can override the name of this library using the **KFJ_SECURITY_EXITS_LIB** parameter in the **CREATE** or **MIGRATE** action.) If you use the **CREATE** or **MIGRATE** action to allocate the library, it will be populated with default security exit members. You can also import existing security exit members if you are migrating your runtime environment from PARMGEN.
- A reference to the security exits library is required in the **RTE_X_SECURITY_EXIT_LIB** parameter located in member `rte_plib_hilev.RTEDEF(rte_name)`.

After the security exits library has been set up using Configuration Manager, you can modify the security exit members as needed for your environment. You can then use the **GENERATE** action to rebuild and relink them.

Setting up security exits using CREATE

You can use the **CREATE** action to allocate the security exits library using the default name and populate it with an initial set of configuration profile members. You can also use the **KFJ_SECURITY_EXITS_LIB** parameter to specify another name for the security exits library. If the specified data set does not exist, it will be allocated and populated with the default security exit members. If the specified data set does exist, it will be populated with the default security members, but no existing member will be overwritten. The **CREATE** action also populates the required reference to the library in the **RTE_X_SECURITY_EXIT_LIB** parameter.

For more information about running the **CREATE** action, see [“CREATE” on page 37](#).

Setting up security exits using MIGRATE

If you are migrating your runtime environment from PARMGEN, you can use the **MIGRATE** action to import the PARMGEN security exits into the new runtime environment. Like the **CREATE** action, the **MIGRATE** action allocates the `rte_plib_hilev.rte_name`.SECEXITS library (or, optionally, the library specified in the **KFJ_SECURITY_EXITS_LIB** parameter). The **MIGRATE** action also copies the security exits used by the PARMGEN environment to the specified security exits library. Because the migration also imports runtime environment configuration settings from the PARMGEN environment, the **RTE_X_SECURITY_EXIT_LIB** parameter will contain the name of the security exits library used by the PARMGEN environment; you must review this setting and update it to use the proper `rte_plib_hilev.rte_name`.SECEXITS library (if necessary) before running the **GENERATE** action.

Note: The security exits library used in PARMGEN is identified in `rte_hilev.rtename`.RKANSAMU and is not changed as a result of the **MIGRATE** action. For more information about the differences between PARMGEN and Configuration Manager, see [“Comparison with PARMGEN” on page 11](#).

For more information about running the **MIGRATE** action, see [“MIGRATE” on page 60](#).

Rebuild and relink security exits using GENERATE

The **GENERATE** action automatically performs the required tasks of rebuilding and relinking the security exits. The **GENERATE** action also provides an optional setting, **OPTION SECEXITS**, that allows you to perform the security exits tasks separately from the normal **GENERATE** workflow, which can save valuable CPU cycles.

For more information about running the **GENERATE** action, see [“GENERATE” on page 47](#).

Using override embed members in Configuration Manager

With Monitoring Configuration Manager, you can use override embed members to provide and maintain customization for your runtime environments.

A Monitoring Configuration Manager configuration creates a set of files that get embedded in a number of the most commonly updated runtime members in the user libraries. These override embed members can be used to specify user-defined parameters and values that might otherwise be overwritten by the **GENERATE** action when maintenance or upgrades are performed, or to override existing values.

The override embed members are stored in the *embeds data set*. The default name for this data set is *rte_plib_hilev.rte_name.EMBEDS*, or you can use a customized name. You can use one data set per RTE or you can share a common data set across multiple RTEs.

When using Monitoring Configuration Manager, you can enable the use of override embed members when creating an RTE or for an existing RTE.

The following parameters provide support for using override embed members:

KFJ_USE_EMBEDS

This Monitoring Configuration Manager parameter controls whether override embed members are enabled for the RTE. When set to Y on the initial **CREATE** or **MIGRATE** action when creating an RTE, Monitoring Configuration Manager sets up the embeds data set, populates it with supported override embed parameters (if applicable), and defines it to the RTE using the **RTE_X_OVERRIDE_EMBEDS_LIB** parameter. Valid values are Y and N. The default is N.

KFJ_EMBEDS_LIB

This Monitoring Configuration Manager parameter identifies the data set that contains the override embed values for the RTE. Use a valid MVS data set name for the value.

RTE_X_OVERRIDE_EMBEDS_LIB

This parameter specifies the name of the source library for override embed members for the RTE and is located in RTEDEF (*rte_name*). This parameter and value is set up automatically when the initial **CREATE** or **MIGRATE** action runs to create an RTE and **KFJ_USE_EMBEDS** is set to Y. It needs to be added manually if you decide to add override embed support to an existing RTE.

Example

An example of how to specify the override embed parameters is shown below. By default, the override embed support is disabled. However, this example shows that it is enabled and provides the data set name.

```
//KCIVARS DD *
ACTION          CREATE
RTE_NAME        DEMO
RTE_PLIB_HILEV  TEST1.TST
...
KFJ_USE_EMBEDS  Y          * Y|N valid values
KFJ_EMBEDS_LIB  TEST1.TST.DEMO.MYEMBEDS * override default EMBEDS library
```

Tip: For more information about override embed members, see PARMGEN topics [Override embed members](#) and [Customizing the override embed members](#). The override embed parameters and values are the same regardless if Monitoring Configuration Manager or PARMGEN is used to configure your RTE. Note that customization of the override embed members is also the same except, whereas in PARMGEN it is done from the WCONFIG, in Monitoring Configuration Manager it is done in the embeds data set specified in **RTE_X_OVERRIDE_EMBEDS_LIB**.

Enable override embed members when creating an RTE

Use this procedure to enable override embed members when creating a runtime environment (RTE).

Before you begin

Read [“Using override embed members in Configuration Manager” on page 140](#).

This task applies to creating an RTE using the Monitoring Configuration Manager [“CREATE” on page 37](#) or [“MIGRATE” on page 60](#) action. For more information, see [“Creating your first, minimal runtime environment” on page 19](#).

About this task

Override embed members can be used to specify user-defined parameters and values that might otherwise be overwritten by the **GENERATE** action when maintenance or upgrades are performed, or to override existing values. Using override embed members for your runtime environment requires a data set (the *embeds data set*) that contains the override embed parameters and values.

You can enable the use of override embed members when creating your RTE by including the **KFJ_USE_EMBEDS** parameter set to Y in the KCIVARS DD statement for the **CREATE** or **MIGRATE** actions. With the inclusion of this setting, Monitoring Configuration Manager creates the embeds data set, populates the override embed parameters and values, and defines the data set to the RTE using the **RTE_X_OVERRIDE_EMBEDS_LIB** parameter.

For the **CREATE** action, by default, the embeds data set is populated with the override embed parameters and values for the products that are installed in the respective CSI used to build the RTE. For the **MIGRATE** action, this library will contain the embed parameters and values from the source PARMGEN installation (as specified by parameter **KFJ_MIGRATE_WCONFIG**).

The default embeds data set name is *rte_plib_hilev.rte_name.EMBEDS*. Optionally, you can include the **KFJ_EMBEDS_LIB** parameter to use a custom data set name rather than the default name. Use of this parameter allows you to use the same embeds data set and settings for multiple RTEs.

Note: If you specify the name of an existing data set using parameter **KFJ_EMBEDS_LIB**, its contents will not be overwritten.

Procedure

1. As part of the process to [create an RTE](#), submit a job that performs the **CREATE** action or **MIGRATE** action with **KFJ_USE_EMBEDS** set to Y and, optionally, **KFJ_EMBEDS_LIB** set to a custom data set name. Example JCL:

Figure 52: Example JCL to enable override embed members for a new RTE

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=<lpar>
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=<tlib_hlq>.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=<tlib_hlq>.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION CREATE | MIGRATE
RTE_NAME <rte_name>
RTE_PLIB_HILEV <rte_plib_hilev>
...
KFJ_USE_EMBEDS Y * Y|N valid values
KFJ_EMBEDS_LIB <embeds_data_set_name> * override default EMBEDs library
/*
```

Where *<rte_name>* and *<rte_plib_hilev>* specify the RTE, and *<embeds_data_set_name>* is the optional custom embeds data set name.

2. Complete the creation of the RTE. For more information, see [“Creating or updating a runtime environment” on page 26](#).
3. Update the override embed members as needed.

Result

Monitoring Configuration Manager sets up the embeds data set, populates it with supported override embed parameters (if applicable), and defines it to the RTE using the **RTE_X_OVERRIDE_EMBEDS_LIB** parameter.

Enable override embed members for an existing RTE

Use this procedure to enable override embed members for an existing runtime environment (RTE).

Before you begin

Read [“Using override embed members in Configuration Manager” on page 140](#).

This task assumes that you have an existing runtime environment (RTE) that was created using Monitoring Configuration Manager. For more information, see [“Creating or updating a runtime environment” on page 26](#).

About this task

Override embed members can be used to specify user-defined parameters and values that might otherwise be overwritten by the **GENERATE** action when maintenance or upgrades are performed, or to override existing values. Using override embed members for your runtime environment requires a data set (the *embeds data set*) that contains the override embed parameters and values.

You can enable the use of override embed members after you have created your RTE by running the **CREATE** action with the **KFJ_USE_EMBEDS** parameter set to Y. With the inclusion of this setting, Monitoring Configuration Manager creates the embeds data set and populates it, by default, with the override embed parameters and values for the products that are installed in the respective CSI.

Note: Running the **CREATE** action after the RTE has been created does not affect existing settings.

The default embeds data set name is *rte_plib_hilev.rte_name.EMBEDS*. Optionally, you can include the **KFJ_EMBEDS_LIB** parameter to use a custom data set name rather than the default name. Use of this parameter allows you to use the same embeds data set and settings for multiple RTEs.

Note: If you specify the name of an existing data set using parameter **KFJ_EMBEDS_LIB**, its contents will not be overwritten.

After the **CREATE** action is performed, you manually define the override embed data set to use for the RTE using the **RTE_X_OVERRIDE_EMBEDS_LIB** parameter.

Procedure

1. Submit a job that performs the **CREATE** action with **KFJ_USE_EMBEDS** set to Y and, optionally, **KFJ_EMBEDS_LIB** set to a custom data set name.

Example JCL:

Figure 53: Example JCL to enable override embed members for an existing RTE

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=<lpar>
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=<tlib_hlq>.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=<tlib_hlq>.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION CREATE
RTE_NAME <rte_name>
RTE_PLIB_HILEV <rte_plib_hilev>
...
KFJ_USE_EMBEDS Y * Y|N valid values
KFJ_EMBEDS_LIB <embeds_data_set_name> * override default EMBEDs library
/*
```

Where *<rte_name>* and *<rte_plib_hilev>* specify the RTE, and *<embeds_data_set_name>* is the optional custom embeds data set name.

This job sets up the embeds data set and, if applicable, populates it with supported override embed parameters.

2. Open the RTEDEF(*rte_name*) member and add the **RTE_X_OVERRIDE_EMBEDS_LIB** parameter with the embeds data set name.

Example:

```
RTE_X_OVERRIDE_EMBEDS_LIB      <embeds_data_set_name>
```

3. Update the override embed members as needed.

Update override embed members for an existing RTE after maintenance

Use this procedure to update override embed members for an existing runtime environment (RTE) using Configuration Manager after applying maintenance.

Before you begin

Read [“Using override embed members in Configuration Manager” on page 140](#).

This task assumes the following points:

- You have an existing runtime environment (RTE) that was created using Configuration Manager and has override embed members enabled.
- You used SMP/E to apply maintenance that introduces changes to your override embed members. The PTF ++HOLD text identifies the override embed members that have updates.

About this task

Some Configuration Manager maintenance includes updates to override embed members. If the RTE has override embed members enabled, Configuration Manager does not apply the changes to the override embed members automatically. Instead, you must perform additional steps manually to apply the updates.

Use the following procedure to update the embeds data set without losing your user modifications.

Procedure

1. Browse the library that is specified in parameter **RTE_X_OVERRIDE_EMBEDS_LIB** in member RTEDEF (*rte_name*), and find the override embed member that was updated. Either rename the override embed member or move it to a different library of your choice that is not controlled by Configuration Manager.

Important: You must preserve the override embed member for later use.

2. Submit a job that performs the **CREATE** action with **KFJ_USE_EMBEDS** set to Y and, optionally, **KFJ_EMBEDS_LIB** set to a custom data set name.

Example JCL:

Figure 54: Example JCL to obtain updated override embed members from maintenance

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=<lpar>
//S1      EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=<tlib_hlq>.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=<tlib_hlq>.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION          CREATE
RTE_NAME        <rte_name>
RTE_PLIB_HILEV  <rte_plib_hilev>
...
KFJ_USE_EMBEDS  Y                * Y|N valid values
KFJ_EMBEDS_LIB  <embeds_data_set_name> * override default EMBEDs library
/*
```

Where *<rte_name>* and *<rte_plib_hilev>* specify the RTE, and *<embeds_data_set_name>* is the optional custom embeds data set name.

This job pulls the updated versions of the override embed members from the updated SMP/E environment.

3. Compare the refreshed override embed member with your backup version. The comparison shows the updates introduced by the latest maintenance and any differences from previous user modifications. Identify the user modifications you must keep, and copy them to the refreshed member.
4. Continue with the standard maintenance procedure (for example, perform the **GENERATE** action).

Deploying remote runtime environments

Use the topics in this section to guide you through the deployment of runtime environments to remote systems.

Remote deployment is the process of creating a configuration-only runtime environment on a configuration system and then deploying the configured data sets to another system (the target system) where the runtime environment will be used.

Remote deployment scenario

In a remote deployment scenario, you must create a runtime environment on a specific LPAR (the configuration system), package the runtime environment data sets using the **PACKAGE** action, transfer the data sets to the remote target system (target system), and deploy (restore) the packaged runtime environment data sets on the target LPAR using the **DEPLOY** action.

Before you begin

Review the following information before you implement a remote deployment:

- Make sure the SMP/E target libraries for TKANMOD and TKANCUS on the target system system and TKANMOD are APF-authorized. APF authorization is needed to run the necessary actions on the target system system.
- If you are using the default RTE_TYPE (that is, SHARING with SMP), make sure a copy of the SMP/E target libraries is available on the target system system, using the value you specified in **GBL_TARGET_HILEV** of member RTEDEF (GBL\$PARM).

If you cannot share your SMP/E target libraries on the target system system, you can use the action **BLDREMS** in the utility flow TKANSAM (KFJMAINT) to build the respective TKANSAM, TKANCUS, and TKANMOD data sets and transfer them to the target system system.

- The z/OS operating system versions on your configuration system and target system should be ideally at the same level. If this is not the case, you will have to customize the z/OS specific libraries, such as SCEELKED in RTEDEF (GBL\$PARM) or RTEDEF (GBL\$lpar), to handle this situation.

For example, parameter **GBL_DSN_CEE_SCEELKED** pointing to the default SCEE.SCEELKED system library could point to the z/OS 2.4 version of the library in **GBL\$lpar1** and the z/OS 2.5 version in **GBL\$lpar2**, respectively.

When generating the runtime environment locally, the respective **GBL\$lparn** member will be used.

Related topics

- [“Special considerations for SYSPLEX rollout” on page 124](#)
- [“RTEDEF\(PCK\\$PARM\)” on page 126](#)
- [“Remote deployment example using local libraries” on page 147](#)
- [“Parameters that cannot be customized for remote deployment” on page 154](#)
- [“KFJ_LOCAL_PLIB_HILEV” on page 102](#)
- [“KFJ_PACK_HILEV” on page 107](#)
- Using **KFJ_LOCAL_PLIB_HILEV** in **CREATE**, **MIGRATE**, **GENERATE**, **PACKAGE**, **DELETE**
- [“PACKAGE” on page 64](#)
- [“DEPLOY” on page 67](#)
- [“Deleting libraries used for remote deployment” on page 59](#)

About this task

The steps in the following procedure describe a sample sequence to be performed for a remote deployment. The procedure uses the terms *configuration system* and *target system* to distinguish the systems. Typically, the target system is a *remote system*.

You cannot customize some parameters when you are creating a runtime environment for remote deployment. For more information, see [“Parameters that cannot be customized for remote deployment” on page 154](#).

Tip: If you need to assemble and link elements (for example, when applying maintenance), you can use the **GENERATE** action with **OPTION RELINK**. For more information, see [“RELINK | NORELINK” on page 52](#).

Procedure

1. For the configuration system, run the **CREATE** action to create an initial RTEDEF data set that will contain the configuration settings of your target system.
(Optional) Specify the **KFJ_LOCAL_PLIB_HILEV** value in the **KCIVARS DD** statement to indicate that you want to use different high-level qualifiers or z/OS® UNIX® System Services paths on the configuration system and the target system. The **KFJ_LOCAL_PLIB_HILEV** parameter is not needed if you will use the same high-level qualifiers on both the configuration system and the target system. See the [“CREATE” on page 37](#) action for more details.
2. For the target system, run the **DISCOVER** action to discover the subsystems and system symbols. This action will also create a RTEDEF data set that will contain the **Kpp@lpar** members for the subsystems discovered as well as the **SYS@lpar** member containing the system symbols.
3. For both the configuration system and the target system, transfer the RTEDEF created on the target system to the configuration system, and merge the contents into the RTEDEF created in step 1 on the configuration system.
4. For the configuration system, customize your runtime environment as per your needs, understanding that the customizations will reflect the target system system, such as the high-level qualifiers needed and features enabled in RTEDEF (**Kpp\$PARM** or **Kpp\$lpar**) members.
(Optional) If you used the **KFJ_LOCAL_PLIB_HILEV** parameter, member RTEDEF (**PCK\$PARM**) will be created, which allows you to map local Qshell and z/OS® UNIX® paths for allocating the runtime environment data sets or z/OS® UNIX® directories on the configuration system.

Note: The settings in RTEDEF (**PCK\$PARM**) are applicable for all runtime environments configured in the RTEDEF data sets, that is, for all runtime environments of the respective SYSPLEX. If you want to use a different local high-level qualifier for a specific target system system on the configuration system, you can create member RTEDEF (**PCK\$lpar**) by copying RTEDEF (**PCK\$PARM**) and making the respective changes.

5. For the configuration system, run the **GENERATE** action for your target system runtime environment by adding the **KFJ_SYSNAME** parameter to the **KCIVARS DD** statement. The value of **KFJ_SYSNAME** specifies the **SYSNAME** or **LPAR** name or the **SYSSMFID** if the **LPAR** name is longer than four characters. See [“KFJ_SYSNAME” on page 115](#) for more details.
(Optional) For the configuration system, if you used the **KFJ_LOCAL_PLIB_HILEV** parameter, after the **GENERATE** action completes, your runtime environment data sets are created using the high-level qualifier or z/OS® UNIX® root path name mapping as specified in RTEDEF (**PCK\$PARM** or **PCK\$lpar**). The members RTEDEF (**PCK\$PARM**) and RTEDEF (**PCK\$LPAR**) will be read and used during runtime environment generation.

Note: In this case, the runtime environment will not be able to start up as the target system settings are used to generate the respective configuration settings.

6. For the configuration system, run the **PACKAGE** action to build transferable dump data sets. Refer to [“PACKAGE” on page 64](#) for more details about the data sets created and the options that can be used.
7. For the configuration system and the target system, transfer the dump data sets to the target system using the procedure of your choice (for example, FTPS).

Important: If you transfer your package data sets with a qualifier that is different from `RTE_PLIB_HILEV` parameter value, you must use the `KFJ_PACK_HILEV` parameter when you deploy the runtime environment in the next step.

8. For the target system, run the **DEPLOY** action to unpack or restore the (tersed) data sets. See [“DEPLOY” on page 67](#) for more details and the options that can be used.
9. For the target system, adjust or copy your started task procedures.

Remote deployment example using local libraries

This detailed example shows how to use local libraries to implement remote deployment of a runtime environment.

Overview

Using local libraries in a remote deployment scenario is required when the system that you are deploying to requires different high-level qualifiers than the high-level qualifiers used on your configuration system.

Tip: If you can use the same high-level qualifiers on the configuration and target systems, you do not have to use local libraries for remote deployment.

This example demonstrates the steps required to create the necessary data sets and parameter settings using local libraries for a runtime environment that will be deployed to another system.

This example use the following terms:

Configuration system

The system where the runtime environment to be deployed is configured and packaged.

Note: A configuration runtime environment (that is, the runtime environment that is configured for the purpose of deployment) cannot be started.

Target system

The system there the package data sets are restored (deployed) and where the runtime environment will be active. Typically, the target system is a *remote* system, but it can also be the same system as the configuration system.

Local libraries

The data sets on the configuration system that are used for creating, configuring, and packaging a runtime environment to be deployed to another system. The parameters in the local libraries on the configuration system are configured for the target system.

Local libraries cannot be used for an active runtime environment on the configuration system.

Local libraries are defined and used with parameter `KFJ_LOCAL_PLIB_HILEV`. The local library `RTEDEF` data set contains special member `PCK$PARM`, which is used specifically for packaging and deploying a runtime environment using local libraries.

In this remote deployment scenario, the DASD for the configuration system and the target system is not shared and requires the use of different high-level qualifiers from each other. In this example, the high-level qualifiers for the configuration system are `DEV . OMEG` and the high-level qualifiers for the target system are `PROD . OMEG`.

Note: The following examples include only the parameters related to remote deployment. Additional parameters in the `KCIVARS DD` might be necessary depending on your requirements.

Diagram

The following diagram shows an overview of the steps for implementing remote deployment of a runtime environment using local libraries.

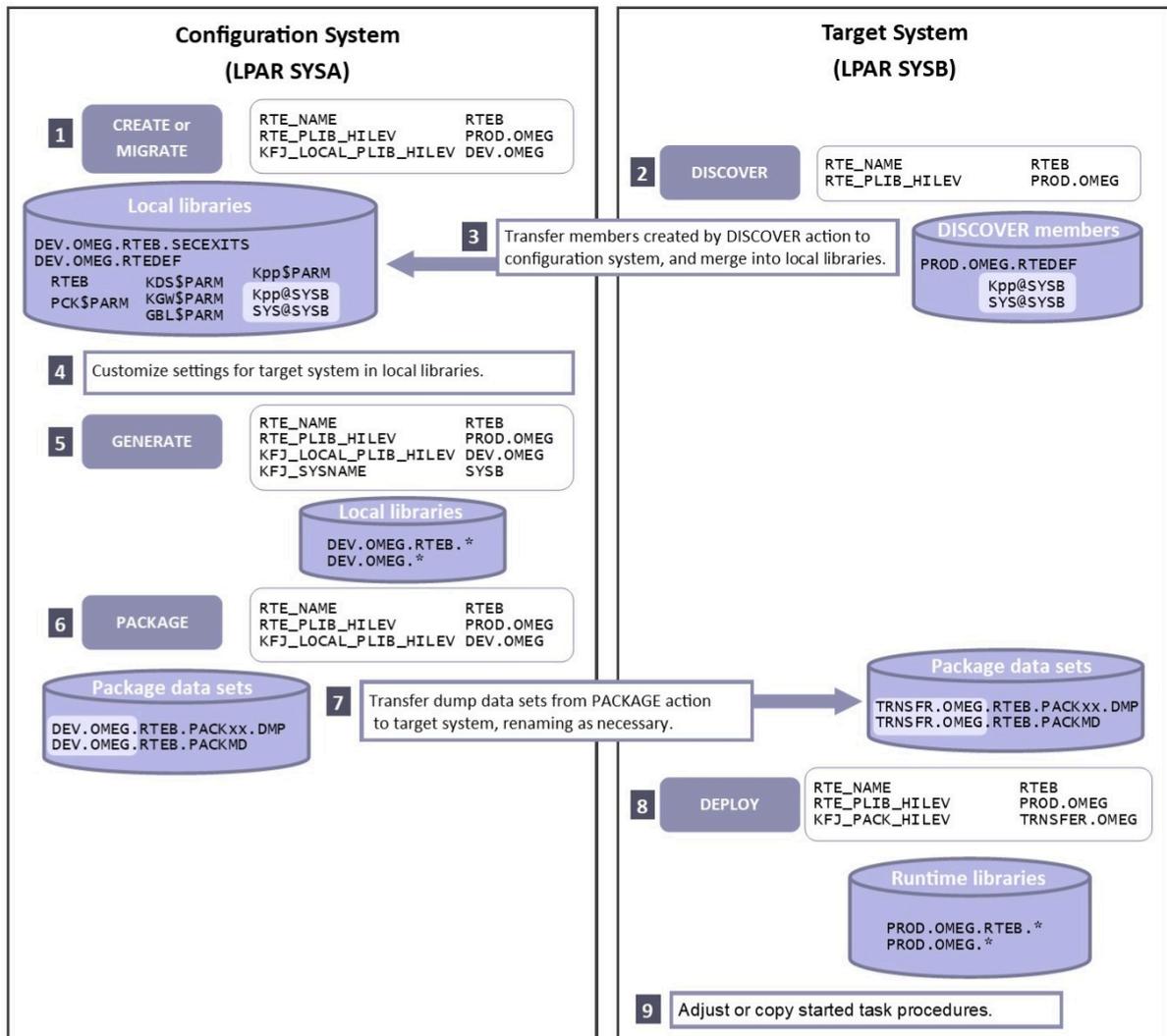


Figure 55: Using local libraries to deploy a runtime environment

For details about each of the numbered steps in the diagram, refer to the following sections.

1. CREATE or MIGRATE actions

On the configuration system, use the **CREATE** or **MIGRATE** action to create the initial local libraries and members that will contain the configuration settings of your target system.

Include the following parameters:

RTE_NAME

Specify the name of the runtime environment that will be active on the target system.

RTE_PLIB_HILEV

Specify the high-level qualifier for the runtime definition (RTEDEF) library to be used on the target system.

KFJ_LOCAL_PLIB_HILEV

Specify the high-level qualifier of the local libraries on the configuration system, which will be used for building the configuration-only runtime environment that will be deployed.

ACTION	CREATE MIGRATE
RTE_NAME	RTEB

```
RTE_PLIB_HILEV      PROD.OMEG
KFJ_LOCAL_PLIB_HILEV  DEV.OMEG
```

Results

Local library data sets and members are created on the configuration system as a result of this step, as follows:

- Local libraries are created on the configuration system using the value specified in **KFJ_LOCAL_PLIB_HILEV** as the high-level qualifier.
- The following members are created in the local library RTEDEF data set:
 - Member *rte_name* is named using the value specified in parameter **RTE_NAME**. The **RTE_PLIB_HILEV** parameter and value appear in this member. For more information about the contents of this member, see [“RTEDEF\(*rte_name*\)” on page 124](#). This member is used for settings for the target system.
 - Member PCK\$PARM is created and contains information about the configuration system. This member is used to map parameters and values in the configuration-only runtime environment to the target system. For more information about the contents of this member, see [“RTEDEF\(PCK\\$PARM\)” on page 126](#).
 - Product and component-specific members *Kpp\$PARM* (or *Kpp\$lpar*) and *GBL\$PARM* (or *GBL\$lpar*) are created. These members are used for settings for the target system. In this example, [KDS\\$PARM](#), [KGW\\$PARM](#), and [GBL\\$PARM](#) are created.

For more information about these RTEDEF members, see [“Initial runtime environment library members” on page 123](#).

Note: The **RTE_NAME** and **RTE_PLIB_HILEV** parameter values are used to populate some parameter values. Other parameter values are set by default to settings for the configuration system. Customization of parameter values occurs in a subsequent step.

- A default security exits library is created: *kfj_local_plib_hilev.rte_name*.SECEXITS

In this example, local libraries using the high-level qualifiers DEV.OMEG are created on the configuration system, as follows:

```
DEV.OMEG.RTEB.SECEXITS
DEV.OMEG.RTEDEF
Members: RTEB
          KDS$PARM
          KGW$PARM
          GBL$PARM
          PCK$PARM
```

2. DISCOVER action

On the target system, run the **DISCOVER** action to discover the subsystems and system symbols.

Include the following parameters, set to the same values that were used in the **CREATE** (or **MIGRATE**) action in the previous step:

RTE_NAME

Specify the name of the runtime environment that will be active on the target system.

RTE_PLIB_HILEV

Specify the high-level qualifier for the runtime definition (RTEDEF) library to be used on the target system.

```
ACTION      DISCOVER
RTE_NAME    RTEB
RTE_PLIB_HILEV  PROD.OMEG
```

Notes:

- Parameter **KFJ_LOCAL_PLIB_HILEV** is not used with the **DISCOVER** action on the target system.
- You must change the program name from **KCIOMEGA** to **KCIALPHA** for the **DISCOVER** job.
- You do not need to run the **CREATE** action on the target system before running the **DISCOVER** action.

Results

This action creates an RTEDEF data set on the target system using the value specified in **RTE_PLIB_HILEV** as the high-level qualifier. This data set contains the *Kpp@lpar* members for the subsystems discovered on the target system and the *SYS@lpar* member containing the system symbols.

In this example, the RTEDEF data set is created on the target system using the high-level qualifiers **PROD.OMEG** and includes the members for the subsystems and system symbols, as follows:

```
PROD.OMEG.RTEDEF
Members: Kpp@SYSB
        SYS@SYSB
```

The *Kpp@lpar* members that are created depend on the subsystems installed on the system.

Note: When discovering MQ subsystems, the **DISCOVER** action creates only a comment member, as indicated by the hash (#) in the member name. For more information, see [“RTEDEF\(KMQ#lpar\)” on page 45](#).

3. Manual steps to transfer and merge your DISCOVER results

Transfer the members created by the **DISCOVER** action (members *Kpp@lpar* and *SYS@lpar*) from the *rte_plib_hilev*.RTEDEF data set on the target system to the *kfj_local_plib_hilev*.RTEDEF data set on the configuration system.

In this example, members *Kpp@SYSB* and *SYS@SYSB* are copied from data set **PROD.OMEG.RTEDEF** on the target system to data set **DEV.OMEG.RTEDEF** on the configuration system.

4. Manual steps to customize your runtime environment before generating runtime members

On the configuration system, customize your configuration-only runtime environment. Perform customization as follows:

- Customize the following RTEDEF members to reflect settings on the target system:
 - Members **KDS\$PARM**, **KGW\$PARM**, and **GBL\$PARM** were created using default values from the configuration system. You must review the parameter values in these members to reflect the appropriate settings for the target system.
 - In the other *Kpp\$PARM* or *Kpp\$lpar* members, specify the high-level qualifiers needed and features enabled.

Important: There are some limitations regarding the parameters that can be customized. Certain parameters are not allowed in the respective RTEDEF members. For the list of parameters, see [“Parameters that cannot be customized for remote deployment” on page 154](#).

- Customize the following RTEDEF member to reflect settings on the configuration system:
 - In member **PCK\$PARM**, you must provide z/OS® UNIX® System Services paths for allocating the runtime environment data sets or z/OS® UNIX® directories on the configuration system.

Important: Because these values are used to generate the runtime members on the configuration system in the next step, you must specify valid local values for parameters **KFJ_LOCAL_USS_RTEDIR** and **KFJ_LOCAL_USS_TKANJAR_PATH**.

Tip: The settings in RTEDEF (PCK\$PARM) are applicable for all runtime environments configured in the RTEDEF data sets; that is, for all runtime environments of the respective SYSPLEX. If you want to use a different local high-level qualifier for a specific target system on the configuration system, you can create member RTEDEF (PCK\$Lpar) by copying RTEDEF (PCK\$PARM) and making the respective changes.

5. GENERATE action

On the configuration system, use the **GENERATE** action to generate runtime members for the runtime environment using the configured parameters. The runtime members are for the runtime environment that will be deployed to the target system and is being configured on the configuration system.

Include the following parameters:

RTE_NAME

Specify the name of the runtime environment that will be active on the target system. Use the same value specified in the **CREATE** action.

RTE_PLIB_HILEV

Specify the high-level qualifier for the runtime definition (RTEDEF) library to be used on the target system. Use the same value specified in the **CREATE** action.

KFJ_LOCAL_PLIB_HILEV

Specify the high-level qualifier of the local libraries on the configuration system. Use the same value specified in the **CREATE** action.

KFJ_SYSNAME

Specify the target system. The value is the SYSNAME or LPAR name, or the SYSSMFID if the LPAR name is longer than four characters. You must specify this parameter for the **DISCOVER**-generated members *Kpp@Lpar* and *SYS@Lpar* to be used.

ACTION	GENERATE
RTE_NAME	RTEB
RTE_PLIB_HILEV	PROD.OMEG
KFJ_LOCAL_PLIB_HILEV	DEV.OMEG
KFJ_SYSNAME	SYSB

Results

The **GENERATE** action creates numerous runtime environment data sets and members in the local libraries on the configuration system using the value specified in **KFJ_LOCAL_PLIB_HILEV** as the high-level qualifier. The parameters in the runtime members are configured to run on the target system using the **RTE_PLIB_HILEV** parameter value as the high-level qualifier.

Important: The configuration runtime environment cannot run on the configuration system. The configuration settings are for the target system.

In this example, the runtime environment data sets and members are generated in the local libraries on the configuration system using the high-level qualifiers DEV.OMEG and the settings defined in DEV.OMEG.RTEDEF, including the PCK\$PARM member. The following data sets are generated and populated:

```
DEV.OMEG.*
DEV.OMEG.RTEB.*
```

The content of the members point to the target system, as shown, for example, in member DEV.OMEG.RTEB.RKANPARU(KPQHINIT):

```
PDSV1=OFF
PDSV2=ON
DSNHILEV=PROD.OMEG.RTEB
```

```
VOLUME=
STORCLAS=
MGMTCLAS=
DATACLAS=
ALLOC_TYPE=CYL
```

The **GENERATE** action also maps the high-level qualifiers and z/OS® UNIX® System Services paths in the local libraries on the configuration system to the runtime libraries on the target system. You can see the mapping results in the **GENERATE** job output in the KCIPRINT DD.

Note: If you use **OPTION DEBUG** with the **GENERATE** action, the mapping results are retained in the KFJSDIFM DD in the job output.

The following figure shows the mapping table that was used in this example:

```
KFJ00211I Configuration Manager is allocating data sets using local settings for high level qualifiers and z/OS UNIX system services paths
```

Configuration LPAR high level qualifiers and z/OS UNIX system services paths	Target LPAR high level qualifiers and z/OS UNIX system services paths
KFJ_LOCAL_PLIB_HILEV	DEV.OMEG
KFJ_LOCAL_HILEV	DEV.OMEG
KFJ_LOCAL_VSAM_HILEV	DEV.OMEG
KFJ_LOCAL_PDS_HILEV	DEV.OMEG.RTEB
KFJ_LOCAL_TARGET_HILEV	DEV.ZSMS21R.SMPE
KFJ_LOCAL_SMS_VOLUME	RTE_PLIB_HILEV
KFJ_LOCAL_SMS_VSAM_VOLUME	RTE_HILIV
KFJ_LOCAL_SMS_UNIT	RTE_VSAM_HILEV
KFJ_LOCAL_SMS_STORCLAS	RTE_PDS_HILEV
KFJ_LOCAL_SMS_VSAM_STORCLAS	GBL_TARGET_HILEV
KFJ_LOCAL_SMS_MGMTCLAS	RTE_SMS_VOLUME
KFJ_LOCAL_SMS_VSAM_MGMTCLAS	RTE_SMS_VSAM_VOLUME
KFJ_LOCAL_USS_RTEDIR	RTE_SMS_UNIT
KFJ_LOCAL_USS_TKANJAR_PATH	RTE_SMS_STORCLAS
KFJ_LOCAL_USS_TKAYHFS_PATH	RTE_SMS_VSAM_STORCLAS
	RTE_SMS_MGMTCLAS
	RTE_SMS_VSAM_MGMTCLAS
	RTE_USS_RTEDIR
	GBL_USS_TKANJAR_PATH
	GBL_USS_TKAYHFS_PATH

Figure 56: Local library mapping to target system (in KCIPRINT DD)

6. PACKAGE action

On the configuration system, run the **PACKAGE** action to build transferable dump data sets.

Include the following parameters:

RTE_NAME

Specify the name of the runtime environment that will be active on the target system. Use the same value specified in the **CREATE** action.

RTE_PLIB_HILEV

Specify the high-level qualifier for the runtime definition (RTEDEF) library to be used on the target system. Use the same value specified in the **CREATE** action.

KFJ_LOCAL_PLIB_HILEV

Specify the high-level qualifier of the local libraries on the configuration system. Use the same value specified in the **CREATE** action.

```
ACTION          PACKAGE
RTE_NAME        RTEB
RTE_PLIB_HILEV  PROD.OMEG
KFJ_LOCAL_PLIB_HILEV  DEV.OMEG
```

Note: (Optional) You can use parameter **KFJ_PACK_HILEV** with the **PACKAGE** action to specify a custom high-level qualifier for the output package data sets, which can be useful to distinguish the package data sets from the many runtime environment data sets in the local libraries.

Results

The **PACKAGE** action allocates the following package data sets:

```
kfj_local_plib_hilev.rte_name.PACKxx.DMP
kfj_local_plib_hilev.rte_name.PACKMD
```

where *xx* is MN (Main non-VSAM), MV (Main VSAM), HN (History non-VSAM), or HV (History VSAM).

Note: If you included parameter **KFJ_PACK_HILEV**, the specified value is used as the high-level qualifier of the package data sets.

For more information about the package data sets, see [“PACKAGE” on page 64](#).

In this example, the following package data sets are created:

```
DEV.OMEG.RTEB.PACKxx.DMP
DEV.OMEG.RTEB.PACKMD
```

7. Manual steps to transfer your package data sets to the remote system

Transfer the package data sets from the configuration system to the target system using the procedure of your choice (for example, FTPS). Make sure to transfer both the DMP and PACKMD data sets. You can rename the high-level qualifier of the data sets if necessary.

In this example, the high-level qualifier of the package data sets is renamed from DEV.OMEG on the configuration system to TRNSFER.OMEG on the target system.

8. DEPLOY on target system

On the target system, use the **DEPLOY** action to restore the package data sets that have been transferred from the configuration system.

Include the following parameters:

RTE_NAME

Specify the name of the runtime environment that will be active on the target system. Use the same value specified in the **CREATE** action.

RTE_PLIB_HILEV

Specify the high-level qualifier for the runtime definition (RTEDEF) library to be used on the target system. Use the same value specified in the **CREATE** action.

KFJ_PACK_HILEV

Specify the high-level qualifier of the package data sets on the target system. This parameter is required if the high-level qualifier does not match the **RTE_PLIB_HILEV** parameter value.

ACTION	DEPLOY
RTE_NAME	RTEB
RTE_PLIB_HILEV	PROD.OMEG
KFJ_PACK_HILEV	TRNSFER.OMEG

Results

The **DEPLOY** action restores the package data sets with the **KFJ_PACK_HILEV** parameter value as the high-level qualifier to data sets with the **RTE_PLIB_HILEV** parameter value as the high-level qualifier. The restored data sets constitute the deployed runtime environment.

In this example, package data sets TRNSFER.OMEG.RTEB.PACKxx.DMP are restored to runtime data sets with high-level qualifier PROD.OMEG. The PROD.OMEG data sets are the runtime libraries for the deployed runtime environment.

Note: Although the PACKMD metadata file itself is not restored, the **DEPLOY** action requires the PACKMD file when local libraries are used for remote deployment.

9. Manual steps to adjust or copy your started task procedures

On the target system, adjust or copy your started task procedures.

Parameters that cannot be customized for remote deployment

You cannot customize some parameters when you are creating a runtime environment for remote deployment.

The following table (“Parameters that cannot be customized for a remote deployment runtime environment” on page 154) lists the parameters that you cannot customize when you are creating a runtime environment for remote deployment.

During the **GENERATE** action, if any of these parameters are explicitly specified in the RTEDEF members, message [KFJ00213E](#) is issued in KCIPRINT and the workflow stops. To continue, you must remove these parameters from the RTEDEF data set members and re-run the **GENERATE** action.

<i>Table 27: Parameters that cannot be customized for a remote deployment runtime environment</i>	
Product or component	Parameter
Global parameters	GBL_USER_JCL
IBM OMEGAMON for CICS	KC2_HSnn_CLASSIC_VSAM_VOLUME KC2_HSnn_CLASSIC_VSAM_STORCLAS KC2_HSnn_CLASSIC_VSAM_MGMTCLAS KC2_HSnn_CLASSIC_VSAM_DATACLAS
IBM OMEGAMON for Db2 Performance Expert ¹	KD2_OMPE_DSHLQ KD2_OMPE_UNIT KD2_OMPE_VOLUME KD2_OMPE_STOCLAS KD2_OMPE_MGMTCLAS KD2_OMPE_VSAM_DSHLQ KD2_OMPE_VSAM_VOLUME KD2_OMPE_VSAM_STOCLAS KD2_OMPE_VSAM_MGMTCLAS KD2_PFn_HIS_DYN_DSNAME KD2_PFn_HIS_DYN_VOLUME KD2_PFn_HIS_DYN_UNIT KD2_PFn_HIS_DYN_SCLAS KD2_PFn_HIS_DYN_MCLAS KD2_PFn_HIS_SEQ_ARC_DS KD2_PFn_HIS_SEQ_ARC_VOLU KD2_PFn_HIS_SEQ_ARC_UNIT KD2_PFn_HIS_SEQ_ARC_SCLAS KD2_PFn_HIS_SEQ_ARC_MCLAS KD2_PFn_HIS_GDG_DSNAME KD2_PFn_HIS_GDG_VOLUME KD2_PFn_HIS_GDG_UNIT KD2_PFn_HIS_GDG_SCLAS KD2_PFn_HIS_GDG_MCLAS KD2_PFn_AEXCP_D2TPTDSN KD2_PFn_AEXCP_D2TPLDSN KD2_PFn_AEXCP_D2TPFDSN KD2_PFn_AEXCP_D2TPVL KD2_PFn_AEXCP_D2TPFSC KD2_PFn_AEXCP_D2TPTFMC
IBM OMEGAMON® for Db2 Performance Expert on z/OS 5.5.0 ¹	KD2_PFn_HIS_VSAM_VOLUME KD2_PFn_HIS_VSAM_SCLAS KD2_PFn_HIS_VSAM_MCLAS KD2_PFn_HIS_SEQ_VOLUME KD2_PFn_HIS_SEQ_UNIT KD2_PFn_HIS_SEQ_SCLAS

Product or component	Parameter
IBM Tivoli OMEGAMON® XE for Db2 Performance Expert on z/OS 5.4.0 ¹	KD2_PFn _n _HIS_LOG _n KD2_PFn _n _HIS_VSAM_VOLUME _n KD2_PFn _n _HIS_VSAM_SCLAS _n KD2_PFn _n _HIS_VSAM_MCLAS _n KD2_PFn _n _HIS_SEQLOG _x KD2_PFn _n _HIS_SEQ_VOLUME _x KD2_PFn _n _HIS_SEQ_UNIT _x KD2_PFn _n _HIS_SEQ_SCLAS _x KD2_PFn _n _HIS_SEQ_MCLAS _x where <i>x</i> is 1 to 7
IBM OMEGAMON for Messaging on z/OS	KQI_HFS_HFSROOT_DIR1
IBM OMEGAMON for Storage on z/OS	KS3_APP_ZFS_DIR
IBM Z OMEGAMON Monitor for z/OS 5.6.0	KM2_HIST_DSTOR_RKM2EDS_DSN _x (where <i>x</i> is 1 to 7)
IBM Tivoli Composite Application Manager (ITCAM) for Application Diagnostics Agent	KYN_XAI01_SUBAGENT_PRODHOME

¹ The list of parameters for IBM Z® OMEGAMON® AI for Db2 (formerly named IBM OMEGAMON® for Db2 Performance Expert on z/OS) cannot be customized if **KFJ_LOCAL_KD5_RUN_ALLOC** is specified with the value GENERATE. If you want to customize these parameters, then use the value DEPLOY or NONE.

Using SMP/E target library copies

Use the Configuration Manager *target copy* feature to create one or more copies of your SMP/E target libraries, from which you can create or update your runtime environments.

Tip: If you are moving from PARMGEN to Configuration Manager, the Configuration Manager target copy feature provides an alternative to the PARMGEN base library feature.

Define SMP/E target copy settings

Create a member in your RTEDEF library to contain your SMP/E target copy settings, and update the settings as needed.

Before you begin

The Configuration Manager target copy feature is used to create and maintain one or more copies of your SMP/E target libraries, from which you can create or update your runtime environments.

The first step in setting up the use of an SMP/E target library copy is to create a dedicated member in your RTEDEF library that will contain the target copy settings. To review the initial content of the SMP/E target copy member that will be created in this task, see [“RTEDEF\(trg_copy_name\)” on page 128](#).

About this task

You use the **CREATE** action with option **TRGCOPY** to create and initially populate a member in an RTEDEF library specifically for SMP/E target copy settings.

The **CREATE** action supports the following settings when creating the SMP/E target copy member:

OPTION TRGCOPY

Creates a target copy member inside the RTEDEF library. The member is named using the **TRG_COPY_NAME** parameter.

TRG_COPY_NAME

Specifies the name of the member to create in the RTEDEF data set to contain the SMP/E target copy settings. If the specified member already exists, it is not overwritten.

TRG_COPY_HILEV

(Optional) Specifies the high-level qualifier for the target copy data sets. If not specified, the **RTE_PLIB_HILEV** parameter value is used.

RTE_PLIB_HILEV

Specifies the high-level qualifier of the RTEDEF library; if the RTEDEF data set does not exist, it is created. This parameter value is also used as the default value for the high-level qualifier for the target copy data sets if **TRG_COPY_HILEV** is not specified.

After the member has been created, you must then review and update the settings, as needed. For more information about the created member, see [“RTEDEF\(trg_copy_name\)” on page 128](#).

Procedure

1. Modify the KFJJMCM sample job in TKANSAM (see example below) to select a **CREATE** action.
2. Specify **OPTION TRGCOPY**.
3. Specify a value for required parameter **TRG_COPY_NAME** to use as the name of the target copy member.
4. Specify the high-level qualifier of the RTEDEF in parameter **RTE_PLIB_HILEV**.
5. (Optional) Specify a value for parameter **TRG_COPY_HILEV** to use as the high-level qualifier for the target copy data sets.

6. Run the KFJJMCM job to create the SMP/E target copy member in the RTEDEF data set. Job messages for the **CREATE** action are written to the KCIPRINT SYSOUT data set.
7. Review and update the RTEDEF (*trg_copy_name*) member, as needed.

The following JCL job creates the SMP/E target copy member MYCOPY in data set TSOUID.PROD.RTEDEF. If the specified RTEDEF data set does not exist, it is created. If the specified member already exists, it is not overwritten.

Figure 57: Example JCL to create the SMP/E target copy member

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUITE.TKANMOD
//KCIFFLOW DD DISP=SHR,DSN=MONSUITE.TKANCUS(KFJOMEGA)
//KCIIVARS DD *
*
ACTION CREATE
OPTION TRGCOPY
TRG_COPY_NAME MYCOPY
RTE_PLIB_HILEV TSOUID.PROD
/*
```

Figure 58: Example SMP/E target copy member

```
* High-level qualifier of SMP/E target libraries
GBL_TARGET_HILEV MONSUITE

* SMP/E target directory containing TKANJAR files (KGW, KJJ)
GBL_USS_TKANJAR_PATH "/usr/lpp/kan/bin/IBM"

TRG_COPY_NAME MYCOPY
* High-level qualifier of the copy of SMP/E target libraries
TRG_COPY_HILEV TSOUID.PROD
* Directory for a copy of SMP/E TKANJAR files (KGW, KJJ)
TRG_COPY_TKANJAR_PATH "/var/rtehome/MYCOPY/kan/bin/IBM"
* Directory for a copy of SMP/E TKAYHFS files (KAY)
TRG_COPY_TKAYHFS_PATH "/var/rtehome/MYCOPY/kay"

CONFIGURE_TEMS_KDS Y * TEMS
CONFIGURE_E3270UI_KOB Y * Enhanced 3270
CONFIGURE_CICS_KC5 Y * CICS TS
CONFIGURE_CICS_TG_KGW Y * CICS TG
CONFIGURE_DB2_AGENT_KD5 Y * Db2
CONFIGURE_IMS_KI5 Y * IMS
CONFIGURE_JVM_KJJ Y * JVM
CONFIGURE_ZOS_KM5 Y * z/OS
CONFIGURE_MESSAGING_KMQ Y * MQ
CONFIGURE_MESSAGING_KQI Y * Integration Bus
CONFIGURE_NETVIEW_KNA Y * Netview
CONFIGURE_MFN_KN3 Y * Network
CONFIGURE_STORAGE_KS3 Y * Storage
CONFIGURE_OMEGAVIEW_KWO Y * Integration Monitor
CONFIGURE_ITCAMAD_KYN Y * ITCAM for Applications
CONFIGURE_ACM_KRN Y * Advanced Catalog Mgmt
CONFIGURE_ARD_KRH Y * Advanced Rpt and Mgmt
CONFIGURE_AAD_KRG Y * Advanced Audit
CONFIGURE_AAM_KRJ Y * Advanced Alloc Mgmt
CONFIGURE_ATAM_KRK Y * Automated Tape Alloc
CONFIGURE_ABR_KRV Y * Advanced Backup and Rec
CONFIGURE_ODP_KAY Y * Data Provider
```

What to do next

Use the new target copy member to make a copy of your SMP/E target libraries. See [“Copy SMP/E target libraries”](#) on page 157.

Note: Make sure to carefully review your target copy settings before continuing with the next step.

Copy SMP/E target libraries

Make a copy of your SMP/E target libraries using your defined SMP/E target copy settings.

Before you begin

The Configuration Manager target copy feature is used to create and maintain a copy of your SMP/E libraries, from which you can create or update your runtime environments. Before you can copy the libraries, you must have created and updated your SMP/E target copy settings, as described in [“Define SMP/E target copy settings” on page 156](#). You can then use your defined settings to copy the libraries, as described in this task.

Important: Make sure you have carefully reviewed your target copy settings before continuing with this task.

About this task

You use the **GENERATE** action with option **TRGCOPY** to copy your SMP/E target libraries from the original source to a new set of data sets, as defined in your SMP/E target copy settings in member RTEDEF (*trg_copy_name*).

Data sets for the copy of the SMP/E target libraries are allocated with the name *trg_copy_hilev.trg_copy_name*. Target copy libraries are always non-VSAM. Only the files for the products installed in your environment, as specified by the **CONFIGURE_*** flags in the RTEDEF (*trg_copy_name*) member, are copied.

When used to copy SMP/E target libraries, the **GENERATE** action supports the following settings

OPTION TRGCOPY

Creates a copy of your SMP/E target libraries from the original source to a new set of data sets, as defined in member RTEDEF (*trg_copy_name*).

TRG_COPY_NAME

Specifies the name of the member in the RTEDEF data set that contains the SMP/E target copy settings.

RTE_PLIB_HILEV

Specifies the high-level qualifier of the RTEDEF library.

Procedure

1. Modify the KFJJMCM sample job in TKANSAM (see example below) to select a **GENERATE** action.
2. Specify **OPTION TRGCOPY**.
3. Specify the name of the target copy member for required parameter **TRG_COPY_NAME**.
4. Specify the high-level qualifier of the RTEDEF in parameter **RTE_PLIB_HILEV**.
5. Run the KFJJMCM job to copy your SMP/E target libraries from the original source to a new set of data sets.
Job messages for the **GENERATE** action are written to the KCIPRINT SYSOUT data set.

The following JCL job copies SMP/E target libraries from the original source to a new set of data sets, as defined in the SMP/E target copy member MYCOPY in data set TSOUID.PROD.RTEDEF. In this example, new data sets are allocated using the high-level qualifier TSOUID.PROD.MYCOPY. The number of data sets copied depends on the products selected for configuration in the target copy member.

Important: The high-level qualifier for the data set name specified in the KCIFLOW DD statement must match the **GBL_TARGET_HILEV** parameter value that is specified in the RTEDEF (*trg_copy_name*) member.

Figure 59: Example JCL to copy SMP/E target libraries

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUIE.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=MONSUIE.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION GENERATE
```

```
OPTION          TRGCOPY
TRG_COPY_NAME  MYCOPY
RTE_PLIB_HILEV TSQUID.PROD
/*
```

What to do next

You can use the copy of the SMP/E target libraries to create or update your runtime environments.

Create a target copy for an existing runtime environment

Use this procedure to configure an existing runtime environment to use a copy of SMP/E target libraries.

Before you begin

This procedure builds upon information provided in the following topics:

- [“Define SMP/E target copy settings” on page 156](#)
- [“Copy SMP/E target libraries” on page 157](#)

You can review these topics for additional details.

About this task

The following procedure provides the steps to update your existing runtime environment to use a copy of SMP/E target libraries instead of original SMP/E target libraries.

Procedure

1. Create the target copy member by running the **CREATE** action with **OPTION TRGCOPY** and the following additional settings:
 - Include the **RTE_PLIB_HILEV** parameter, which must point to an existing RTEDEF data set.
 - Include the **TRG_COPY_NAME** parameter, which must specify a new member name, one that does not exist in the RTEDEF data set.

For more details, see [“Define SMP/E target copy settings” on page 156](#).

2. After new member RTEDEF (*trg_copy_name*) has been created, review the settings in the member and update as needed.

Note: Make sure to review parameter **TRG_COPY_HILEV** in the new member, which specifies the high-level qualifier for the SMP/E target library copy.

3. Create the copy of the SMP/E target libraries by running the **GENERATE** action with **OPTION TRGCOPY** and the following additional settings:
 - Include the **RTE_PLIB_HILEV** parameter, which must point to an existing RTEDEF data set.
 - Include the **TRG_COPY_NAME** parameter, which must specify the name of the newly created member.

For more details, see [“Copy SMP/E target libraries” on page 157](#).

4. After the **GENERATE** action has completed and the SMP/E target libraries have been copied to the new location, make the following updates:
 - a. Update parameter **GBL_TARGET_HILEV** in member RTEDEF (GBL\$PARM) or RTEDEF (GBL\$lpar) to point to the newly created copy of the SMP/E target libraries.
 - b. If parameter **GBL_USS_TKANJAR_PATH** parameter is required for your runtime environment, update the **GBL_USS_TKANJAR_PATH** parameter to use the value from **TRG_COPY_TKANJAR_PATH** in member RTEDEF (*trg_copy_name*).
5. Run the **GENERATE** action for the modified runtime environment.

6. Perform standard post-configuration steps, such as copying generated started tasks into a site-specific location.
7. (Optional) You can repeat steps 1 through 3 to create multiple copies of your SMP/E target libraries, which you can use to apply a staged rollout of maintenance for an existing runtime environment.

Maintain SMP/E target library copies

Keep the copy of your SMP/E target libraries up to date.

Before you begin

The Configuration Manager target copy feature is used to create and maintain one or more copies of your SMP/E target libraries, from which you can create or update your runtime environments.

After you have created a copy of your SMP/E target libraries using the target copy feature, as described in [“Copy SMP/E target libraries” on page 157](#), you can refresh the copy as needed, as described in this task.

About this task

You use the **GENERATE** action with option **TRGCOPY** to refresh the copy of your SMP/E target libraries.

Procedure

- To keep the SMP/E target copy in sync with your original SMP/E target libraries, simply run the **GENERATE** action with option **TRGCOPY** again. It will refresh all the libraries and will copy all the required members from your SMP/E target libraries to a copy of the SMP/E target libraries.

How-tos using Configuration Manager

How-tos provide some best practices and instructions for performing specific tasks. The topics in this section tell you *how to* perform these tasks using Configuration Manager. These instructions are meant to help you complete commonly used processes.

How to: Migrate to Configuration Manager from PARMGEN

You can migrate a runtime environment that is configured with PARMGEN to one that is configured with Configuration Manager.

To perform the migration, use the Configuration Manager **MIGRATE** action. The **MIGRATE** action imports configuration settings from a runtime environment that is configured with PARMGEN to one that is configured with Configuration Manager.

For more information about moving from PARMGEN to Configuration Manager, see the following topics:

- [Best practices for configuring OMEGAMON products and components](#)
- [PARMGEN: Rectifying your PARMGEN configuration](#)
- [Configuration Manager: Comparison with PARMGEN](#)
- [Configuration Manager: Parameters with different default values than PARMGEN](#)
- [Configuration Manager: MIGRATE](#)

How to: Add a new agent or product to a runtime environment

This topic describes how to add a new agent or product to an existing runtime environment using Configuration Manager.

About this task

To add a new agent or product to a runtime environment, you create new members for product-specific parameters in the existing runtime environment definition (RTEDEF) library.

Procedure

1. Run the JCL with the **CREATE** action. The **RTE_NAME** and **RTE_PLIB_HILEV** parameters should point to an existing RTEDEF library (the one you want to modify). See [“CREATE” on page 37](#) for more information. If you want to use override embed members, you should also specify Y for that parameter, for example, **KFJ_USE_EMBEDS Y**. See [“Using override embed members in Configuration Manager” on page 140](#) for more information.

The first step is to create new members in the RTEDEF library (for example, **KDS\$PARM**), if necessary, where you can set product specific parameters. This step will also copy new override embed members and security exits (if there are any).

Note: If the **Kpp\$PARM** member was not created for a new product, that means there are no mandatory or best practice parameters to set. However, if you want to add custom parameters, create the **Kpp\$PARM** member in the RTEDEF library and add custom parameters.

2. When step 1 is complete, manually update the RTEDEF (*rte_name*) member and specify which agent or product to configure. For example, to configure the KD5 agent, set the **CONFIGURE_DB2_AGENT_KD5** parameter to Y. For the complete list of **CONFIGURE_*** parameters, see [“Parameters that determine what products are configured” on page 78](#).

Note: If you want to remove an existing agent, set the parameter to N. The primary benefit of the N setting is that less CPU and time is used as the respective product-specific workflows are bypassed.

3. Manually update RTEDEF (GBL\$PARM), if there are any initial mandatory parameters for the new product, such as load library names. See “Parameters in the initial runtime environment configuration profile” on page 79 for information on how to add those parameters to GBL\$PARM and set correct values.
4. (Optional) Run JCL with the **DISCOVER** action. See “DISCOVER” on page 40 for more information.
5. Run the **GENERATE** action to finalize RTE configuration. See “GENERATE” on page 47 for more information.

How to: Add OMEGAMON® Data Provider to a runtime environment

This topic describes how to add OMEGAMON® Data Provider to an existing runtime environment using Configuration Manager.

Before you begin

You can add IBM Z® OMEGAMON Data Provider to an existing runtime environment by manually adding required parameters to your configuration, if the parameters are not present in the runtime environment definition (RTEDEF) library.

Important: If you are creating a new runtime environment, Configuration Manager includes the OMEGAMON® Data Provider parameters automatically, and you do not need to perform this task.

Before you begin, you need to know the SMP/E target locations where OMEGAMON® Data Provider is installed. In particular, you must know the z/OS® UNIX® System Services directory path specified by the SMP/E DDDEF TKAYHFS. The default path is /usr/lpp/omdp. For more information, see [Installing OMEGAMON Data Provider](#).

About this task

To add OMEGAMON® Data Provider to an existing runtime environment, you perform the following actions:

- Add required parameters to members *rte_name* and GBL\$PARM (or GBL\$lpar) in the runtime environment definition (RTEDEF) library.
You must set the following required parameters:

CONFIGURE_ODP_KAY

Enables or disables OMEGAMON® Data Provider configuration for the runtime environment

GBL_USS_TKAYHFS_PATH

Sets the path to the z/OS® UNIX® System Services directory that the SMP/E installation jobs define using the ddname TKAYHFS

- For parameters that are specific to OMEGAMON® Data Provider, create new member KAY\$PARM (or KAY\$lpar) in the existing RTEDEF library.
You can optionally set the following parameters:

For OMEGAMON Data Broker:

KAY_BROKER_MEMBER

Sets the two-character value for the KAYSIPnn member name and the OMEGAMON Data Broker procedure MEM=nn parameter value

KAY_BROKER_NAME

Sets the OMEGAMON Data Broker name, which must match the name specified in the KAYOPEN member

KAY_BROKER_STC

Sets the OMEGAMON Data Broker procedure name

For OMEGAMON Data Connect:

KAY_CONNECT_JAVA_HOME

Sets the path to the Java 17 installation directory

Note: Other Java versions are not supported.

KAY_CONNECT_STC

Sets the OMEGAMON Data Connect procedure name

Use the following procedure to add OMEGAMON® Data Provider to your existing runtime environment.

Procedure

1. In the RTEDEF library, edit member *rte_name* and add parameter **CONFIGURE_ODP_KAY** with the value Y.
2. In the RTEDEF library, edit member GBL\$PARM (or GBL\$Lpar) and add parameter **GBL_USS_TKAYHFS_PATH** with the value set to the path of the z/OS® UNIX® System Services directory that the SMP/E installation jobs define using the ddname TKAYHFS.
3. (Optional) In the RTEDEF library, create member KAY\$PARM (or KAY\$Lpar), if necessary, where you can set the following product-specific parameters:

KAY_BROKER_MEMBER
KAY_BROKER_NAME
KAY_BROKER_STC
KAY_CONNECT_JAVA_HOME
KAY_CONNECT_STC

For example, after you create member KAY\$PARM, add the **KAY_CONNECT_JAVA_HOME** parameter with a value that points to the Java 17 directory, as follows:

```
KAY_CONNECT_JAVA_HOME  "/path/to/Java/J17.0_64"
```

4. Run the **GENERATE** action to finalize the runtime environment configuration.

Result

The **GENERATE** action performs the following actions:

- Creates the following OMEGAMON® Data Provider procedures in library GBL_DSN_SYS1_PROCLIB, which is (by default) *rte_plib_hilev.SYS1.PROCLIB*:
 - OMEGAMON Data Connect procedure OMEGODC
 - OMEGAMON Data Broker procedure OMEGODB
- Copies member KAYOPEN to runtime library *rte_hilev.rte_name.RKANPARU*

Note: Member KAYOPEN is copied only once and is never replaced.

- Copies member KAYSIP00 to global source library %GBL_DSN_GLOBAL_SOURCE_LIB%, which is (by default) *rte_hilev.rte_name.RKANPARU*

Note: Member KAYSIP00 is copied only once and is never replaced.

- Creates the following z/OS® UNIX® directories for the runtime environment:

OMEGAMON Data Connect user directory

- *rte_uss_rtedir/rte_name/kay/user1/config*, which contains *connect.yaml* samples
- *rte_uss_rtedir/rte_name/kay/user1/extensions*, an empty directory that is ready for optional OMEGAMON® Data Provider extensions

OMEGAMON Data Connect installation directory

- For full runtime environments (**RTE_TYPE** parameter is set to FULL), Configuration Manager extracts the KAY11PAX archive to directory `rte_uss_rtedir/rte_name/kay/kay-110/`

What to do next

After the Configuration Manager process completes, review [Getting started with OMEGAMON Data Provider](#) to perform other necessary steps.

To stream status information from the Tivoli Enterprise Monitoring Server to OMEGAMON® Data Provider, see [\(Optional\) Configure situation status streaming](#).

How to: Use a shared procedure for multiple monitoring servers

You can configure your environment to use the same started task procedure for multiple monitoring servers across different systems in the sysplex. This topic explains how to perform this task using either Configuration Manager or PARMGEN.

Before you begin

A Tivoli Enterprise Monitoring Server (*monitoring server*) can be configured as one of several types: hub, high-availability hub, remote. By default, the started task procedure for each monitoring server is created to be run as a separate task. The generated procedure is different for each monitoring server by type; for example, the name of the runtime environment, the data set names, and the required DD statements can all vary by monitoring server type.

This default configuration is not suitable for some sites that might have any of the following requirements:

- The SYS1.PROCLIB (or equivalent procedure library) must be shared across all the systems in the sysplex.
- Only a single, specific agent procedure is allowed (that is, only one procedure per agent).

To satisfy such requirements, you must use a procedure that is the same for all monitoring servers across all systems; it must have the same name, the same contents, and reference the same data sets. To accomplish this setup, you can configure your environment to generate a *shared procedure*.

This topic explains how to generate a shared procedure for use with multiple monitoring servers using either Configuration Manager or PARMGEN.

About this task

This task requires the use of the following parameter:

KDS_TEMS_PROC_SHARED

This flag specifies if the monitoring server procedure can be shared among different systems in the sysplex. When the flag is set to Y, the same procedure is generated for a hub monitoring server or a remote monitoring server, and the procedure can be copied to a shared SYS1.PROCLIB. The default value for this parameter is N.

The use of system variables is also required.

Use the following steps to create a shared procedure for use with multiple monitoring servers.

Procedure

1. Set the flag parameter that enables the generation of a monitoring server procedure that can be shared:
 - Using Configuration Manager: Add parameter **KDS_TEMS_PROC_SHARED** set to Y to RTEDEF (KDS\$PARM) or RTEDEF (KDS\$lpar), and then run the **GENERATE** action.
 - Using PARMGEN: In WCONFIG (*rte_name*), set parameter **KDS_TEMS_PROC_SHARED** to Y, and then run the **\$PARSE** job.

Note: For more information about making these changes using PARMGEN, see [Scenario SMPE03: Applying SMP/E maintenance with new configuration changes to an existing RTE and overriding the IBM-supplied configuration defaults](#).

2. Review the generated procedure, and proceed as appropriate:

- If you use system variables in your runtime environment configuration, including a system variable for **RTE_NAME**, the generated procedure is ready to be shared.
- If you do not use system variables in your runtime environment configuration, you can use the generated procedure as a shared procedure if there is a variable in your PARMLIB system symbol list that corresponds to your **RTE_NAME** values across all the LPARs in the sysplex. To enable the procedure to be shared, update the following lines:

```
//TEMS PROC RGN=0M,TIM=1440,MEMLIM=NOLIMIT,
//          SYS=&SYSNAME. ,
```

In this example, **&SYSNAME.** resolves to an **RTE_NAME** on each system where it is started.

3. Copy the shared started task procedure to your system library.

How to: Create a high-availability hub monitoring server

You can ensure continuous availability in your monitoring environment by using a high-availability hub monitoring server. This task describes how to create a high-availability hub monitoring server using Configuration Manager.

Before you begin

A configuration that includes a [high-availability hub](#) is resilient and efficient. It is resilient because the high-availability hub can be relocated to any LPAR in the sysplex with minimal disruption to the other components. The configuration is efficient because the remote monitoring server on the same LPAR as the hub handles all communications with the monitoring agents and thus reduces the load on the hub. If your environment supports the requirements for a high-availability hub, you should configure one.

A high-availability hub Tivoli Enterprise Monitoring Server has the following requirements:

- Must reside on z/OS.
- Must be configured in a sysplex. A high-availability hub is not defined to run on any specific system.
- Must use runtime libraries stored on shared DASD. This setup allows the high-availability hub to start on any LPAR in the sysplex without requiring replication of the libraries.
- Must be configured in its own runtime environment, without any monitoring agents configured in the runtime environment. The high-availability hub can be configured on the same LPAR with a remote monitoring server.
- Must have a dynamic virtual IP address (DVIPA). Using a DVIPA allows the high availability hub to respond to the same IP address on any LPAR in the sysplex. The DVIPA must be defined on each LPAR that is a candidate for the high-availability hub. It is recommended that you use the VIPADEFINE statement to designate the DVIPA. For more information, see *z/OS Communications Server: IP Configuration Reference: [VIPADYNAMIC - VIPADEFINE statement](#)*.

Additionally, it is recommended that system variables are not used in the runtime environment for the high-availability hub. The high-availability hub retains the same parameter values on any system in the sysplex.

About this task

To create a high-availability hub monitoring server using Configuration Manager, you must isolate the configuration settings for the monitoring server in a separate runtime environment definition (RTEDEF) library.

Use the following parameters to configure a high-availability hub monitoring server:

- **[RTE_TCP_HOST](#)**
- **[KDS_TEMS_TYPE](#)**
- **[KDS_TEMS_HA_TYPE](#)**
- **[KDS_TEMS_TCP_KDEB_INTERFACELIST](#)**
- **[KDS_PH01_TEMS_TCP_HOST](#)**

Complete the following steps to create a high-availability hub monitoring server.

Procedure

1. Run the **CREATE** action to create an initial set of parameters in a new RTEDEF library for the high-availability hub runtime environment.

Note: Any generated members other than the global (**GBL_***) and monitoring server (**KDS\$***) members can be removed.

2. Update the RTEDEF (*rte_name*) member, as follows:
 - Ensure that the runtime environment contains only the high-availability hub monitoring server by setting the following parameter to Y:

```
CONFIGURE_TEMS_KDS          Y
```

All other **CONFIGURE_*** parameters are set to N.

- Specify the address where the runtime environment is being defined:

```
RTE_TCP_HOST                dvipa
```

where *dvipa* is the DVIPA for the high-availability hub.

3. Update the RTEDEF (KDS\$PARM) member to include the following parameters and values:

```
KDS_TEMS_TYPE              HUB
KDS_TEMS_HA_TYPE           HA
KDS_TEMS_TCP_KDEB_INTERFACELIST  !dvipa

KDS_PH                     BEGIN
KDS_PH01_ROW               01
KDS_PH01_TEMS_TCP_HOST    dvipa
KDS_PH                     END
```

where *dvipa* is the DVIPA for the high-availability hub.

4. Use the **GENERATE** action to create runtime members for the high-availability hub runtime environment.

How to: Configure passphrase and MFA support in the OMEGAMON 3270 Classic interface

In addition to using a regular password, you can also log on securely to the OMEGAMON 3270 Classic interface using a password phrase (passphrase) and multi-factor authentication (MFA). Some configuration steps are necessary to enable passphrase and MFA support for the OMEGAMON 3270 Classic interface. This topic explains how to perform this task using either Configuration Manager or PARMGEN.

Before you begin

A traditional mainframe password is eight bytes or less, while a passphrase is from nine to 100 bytes. MFA is an authentication method that typically requires a six-digit volatile numeric token that is paired with a password or passphrase value. A user ID must be set up in the security system to use a passphrase.

Note: Your security administrator must set up the user ID to use a passphrase. For RACF, use the PHRASE operand with the **ADDUSER** or **ALTUSER** command. For a security product other than RACF, refer to the documentation for that product for guidance on the equivalent actions.

On a 3270 screen, depending on the screen width, entering a long passphrase value into a field might require multiple lines. For example, if the screen width is 80 bytes, an input field would require multiple lines to support a value longer than 80 bytes. For a wider screen size, you can support a longer value on a single line, up to the available screen width.

On the OMEGAMON 3270 Classic interface logon screen, by default, the password fields support passwords that are eight bytes or less. Optionally, you can configure your product to support passphrase and MFA values for the OMEGAMON 3270 Classic interface; multiple settings are available. When passphrase support is

enabled, configuration parameters are used to specify the SAF security class and SAF application ID to use for the OMEGAMON 3270 Classic interface.

Important: When passphrase support is enabled, OMEGAMON implements the SAF interface for external security without the use of security exits. For more information, see [OMEGAMON® 3270 Classic interface security](#).

About this task

To use passphrase values and MFA for the OMEGAMON 3270 Classic interface, you must configure your product to enable passphrase support. Multiple passphrase configuration options are available that affect the length of the passphrase that is supported on a single line and the layout of the logon screen.

Note: It is recommended that you review the available configuration options, especially if you use programs to automate the logon process to the OMEGAMON 3270 Classic interface that rely on static placement of keywords and input fields.

Passphrase support for the OMEGAMON 3270 Classic interface is provided for the following products, listed with the respective product code:

- OMEGAMON for CICS (C2)
- OMEGAMON® AI for Db2 (D2)
- OMEGAMON for IMS (I2)
- OMEGAMON for z/OS (M2)
- IBM Z OMEGAMON Monitor for z/OS (M2)

Passphrase enablement and configuration is controlled by parameter **Kpp_CLASSIC_PASSPHRASE**, where *pp* is C2, D2, I2, or M2, depending on the supported product. When passphrase support is enabled, the SAF security class is defined by **Kpp_CLASSIC_SECCLASS** and the SAF application ID is defined by **Kpp_CLASSIC_SAFAPPL**, where *pp* is C2, D2, I2, or M2.

The following configuration options are available for passphrase support:

Kpp_CLASSIC_PASSPHRASE

This parameter specifies the passphrase support setting for the OMEGAMON 3270 Classic interface.

Note: In the following figures, a ruler is shown on the screen. The ruler is included in the documentation for illustrative purposes only and is not displayed in the product.

PARTIAL

Passphrase support is enabled with the **PASSWORD** and **NEW PASSWORD** fields each consisting of a single line. The minimum length of each of these fields is 34 bytes, and the maximum length (which can be up to 100 bytes) depends on the screen width. With this setting, the fields are aligned in the center of the screen, as shown in the following figure:

```

>
>          Copyright 1980-2022
>      IBM Corporation. All rights reserved.
>      Use permissible by license only.
>

          ENTER USERID ==>
            PASSWORD ==>
              GROUP ==>
            NEW PASSWORD ==>

          Press F3 to exit logon

|...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8
1234567890123456789012345678901234567890123456789012345678901234567890

```

Note: Passphrase support for the OMEGAMON 3270 Classic interface is introduced with APAR OA57133 (PTF UA98944). With the PARTIAL setting, the input field labels and placement are compatible with the screen layout before passphrase support was introduced.

MAX62

Passphrase support is enabled with the **PASSWORD** and **NEW PASSWORD** fields each consisting of a single line. The minimum length of each of these fields is 62 bytes, and the maximum length (which can be up to 100 bytes) depends on the screen width. With this setting, the fields are aligned at the left of the screen, as shown in the following figure:

```

>
>          Copyright 1980-2022
>      IBM Corporation. All rights reserved.
>      Use permissible by license only.
>

ENTER USERID ==>
  PASSWORD ==>
    GROUP ==>
NEW PASSWORD ==>

          Press F3 to exit logon

|...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8
1234567890123456789012345678901234567890123456789012345678901234567890

```

FULL

Passphrase support is enabled with the **PASSWORD** and **NEW PASSWORD** fields each consisting of two lines. The value in the second line is concatenated onto the end of the value in the first line. The length of the first line is 34 bytes and the length of the second line is 66 bytes, allowing the maximum passphrase value of 100 bytes to be entered. With this setting, the fields are aligned in the center of the screen, as shown in the following figure:

```

>
>          Copyright 1980-2022
> IBM Corporation. All rights reserved.
> Use permissible by license only.
>

          ENTER USERID ==>
          PASSWORD ==>

          GROUP ==>
          NEW PASSWORD ==>

          Press F3 to exit logon

|...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8
1234567890123456789012345678901234567890123456789012345678901234567890

```

NO or NONE

Passphrase support is not enabled. The lengths of the **PASSWORD** and **NEW PASSWORD** fields are eight bytes each. With this setting, if you have external security defined using a security exit, the fields are aligned in the center of the screen, as shown in the following figure:

```

>
>          Copyright 1980-2020
> IBM Corporation. All rights reserved.
> Use permissible by license only.
>

          ENTER USERID ==>
          PASSWORD ==>
          GROUP ==>
          NEW PASSWORD ==>

          Press F3 to exit logon

|...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8
1234567890123456789012345678901234567890123456789012345678901234567890

```

Note: If you do not have external security defined, none of the fields for credentials appear on the logon screen.

Use the following procedure to enable passphrase and MFA support for your OMEGAMON 3270 Classic interface. If you do not want to use passphrase or MFA when logging on to the OMEGAMON 3270 Classic interface, no configuration changes are needed.

Procedure

To enable passphrase support for your OMEGAMON 3270 Classic interface, perform the following steps for each of your supported OMEGAMON products. Use either of the following methods:

- Using Configuration Manager:
 - a. In RTEDEF(Kpp\$PARM) or RTEDEF(Kpp\$lpar), add the following parameters:
 - Kpp_CLASSIC_PASSPHRASE** set to value PARTIAL, MAX62, or FULL
 - Kpp_CLASSIC_SECCLASS** set to the OMEGAMON SAF security class
 - Kpp_CLASSIC_SAFAPPL** set to the OMEGAMON SAF application ID
 - b. Run the **GENERATE** action.

Recycle the OMEGAMON Classic started task for the configuration changes to take effect. See the product-specific documentation for more information.

Note: For more information about changing parameter values after you have completed configuration of the runtime environment using Configuration Manager, see [“Creating or updating a runtime environment” on page 26](#).

- Using PARMGEN:
 - a. In WCONFIG(*#rtename*), add the following parameters:
 - Kpp_CLASSIC_PASSPHRASE** set to value PARTIAL, MAX62, or FULL
 - Kpp_CLASSIC_SECCLASS** set to the OMEGAMON SAF security class
 - Kpp_CLASSIC_SAFAPPL** set to the OMEGAMON SAF application ID
 - b. Submit the \$PARSE job to refresh the profile.

Recycle the OMEGAMON Classic started task for the configuration changes to take effect. See the product-specific documentation for more information.

Note: For more information about changing parameter values after you have completed configuration of the runtime environment using PARMGEN, see [Scenario RTE03: Changing parameters in an RTE](#).

Troubleshooting

Use these topics to troubleshoot issues with Monitoring Configuration Manager.

Navigating Configuration Manager action output

Use the method described in this procedure to navigate Configuration Manager output when troubleshooting a problem.

Before you begin

Configuration Manager writes job output for the **KCIOMEGA** actions to a number of output data sets. Output from invoked utilities is also generated, with each utility writing to its own output data set.

Note: **KCIALPHA** is an APF-authorized version of the **KCIOMEGA** program. The information in this topic also applies when using **KCIALPHA**.

For a list of the standard sysout data sets, see [“Action job output” on page 74](#).

About this task

When troubleshooting an issue with Configuration Manager jobs, use the steps in the following procedure to navigate the Configuration Manager output.

To view Configuration Manager output, it is recommended that you use SDSF and the ? action character to list the output data sets. Consider sorting by DSID in SDSF to get the proper order.

Since most problems cause the Configuration Manager action to immediately stop, it is likely that the last utility invoked is the cause of the problem.

Note: For most problems, you do not need to examine the KCITRACE sysout data set as it is typically only required when there is a logic error in the workflow. KCITRACE is mostly used by IBM Software Support.

Procedure

1. View the KCIPRINT sysout data set, and look for the error message that explains why the problem occurred.
2. If the problem was caused by a utility, then scroll down to the end of the data set list and select the last file.
This is typically the output from the failed utility and should provide more information about the problem.
3. In the event of a system problem (for example, an abend), then view JESMSGLG (the JES message log) and look for abnormal system messages.

Collecting diagnostic data using PDCOLLECT

Use the Problem Determination Data Collection (PDCOLLECT) action to collect diagnostic information. This section explains how to perform this task using Configuration Manager.

Important: The Configuration Manager PDCOLLECT utility action was enhanced with [PTF UJ98129 for APAR OA68463 \(3Q25\)](#). The procedure that you use to run PDCOLLECT and the features that are available depend on whether this maintenance has been applied to your environment.

Use one of the following procedures, as appropriate for your environment.

Using PDCOLLECT (with OA68463)

Use the Problem Determination Data Collection (PDCOLLECT) action to collect diagnostic information. This topic explains how to perform this task using Configuration Manager with APAR OA68463 applied.

Before you begin

Note: The content in this topic is relevant only for environments with PTF UJ98129 for APAR OA68463 (3Q25) applied. If you do not have this maintenance applied, see [“Using PDCOLLECT \(without OA68463\)” on page 177](#).

To use the **PDCOLLECT** action, you must install SDSF and be licensed to use it.

Note: If you do not have SDSF, you can copy the complete address space logs, including the JES output, RKLVLG, RKPLOG, RKPLOG, SYSPRINT, and so on, into a data set. The DCB information for the output data set can be as follows:

```
Organization: PS
Record format: VB
Record length: 240
Block size: 27998
```

About this task

The **PDCOLLECT** utility action collects the following types of diagnostic data:

- System configuration information
- Runtime environment (RTE) libraries, including configuration files
- Configuration Manager-related information (RTEDEF and EMBEDS data sets)
- Network information, such as output from the **netstat** utility
- Output from the started task jobs, including address space logs
- SDSF Display Active Users (DA) panel statistics
- Self-describing agent (SDA) information:
 - Results of **ls -al** for your *rte/kds/support* and *rte/kds/support/TEMS*
 - Contents for *spawn_cmd.sh* and *spawn_cmd.trs*
 - Results of **java - version**

The following list provides details about specifying collection criteria for the **PDCOLLECT** action:

- The **PDCOLLECT** action does not require the **RTE_NAME** and **RTE_PLIB_HILEV** parameters and values. Omitting the runtime environment information provided by these parameters limits the types of diagnostic data that the **PDCOLLECT** action can collect. To collect the most information available about your environment, include the **RTE_NAME** and **RTE_PLIB_HILEV** parameters and values.
- You must specify information about your OMEGAMON address space logs. The parameters you use depend on the where the logs are located, as follows:
 - If your OMEGAMON address space logs are in the SDSF output queue, use the **KFJ_PDCOL_JOB_NAME** and **KFJ_PDCOL_JOB_ID** parameters and values. To collect data for a single job, specify the job name and job ID in parameters **KFJ_PDCOL_JOB_NAME** and **KFJ_PDCOL_JOB_ID**, respectively.

You can also capture multiple started task logs and SDSF Display Active Users (DA) panel snapshots by using filtering and wildcard characters. To specify multiple started task logs, enable filtering by specifying parameter **KFJ_PDCOL_JOB_FILTER** set to Y. You can then use the asterisk (*) and percent sign (%) as wildcard characters in the **KFJ_PDCOL_JOB_NAME** and **KFJ_PDCOL_JOB_ID**

parameter values to define the filters, as shown in the following example. Only started task logs matching the filters will be collected.

KFJ_PDCOL_JOB_NAME	OMEG*
KFJ_PDCOL_JOB_ID	*
KFJ_PDCOL_JOB_FILTER	Y

Tip: When job filtering is enabled (**KFJ_PDCOL_JOB_FILTER** set to Y), parameter **KFJ_PDCOL_JOB_ID** defaults to the asterisk (*) wildcard. If you accept this default value, you can omit parameter **KFJ_PDCOL_JOB_ID** from your **KCIVARS DD** statement.

- If your OMEGAMON address space logs have been copied to a sequential data set, use the **KFJ_PDCOL_JOB_OUTPUT** parameter and value to specify the data set name.
- You can select the diagnostic data to collect at a detailed level by using the **KFJ_PDCOL_COLLECT_data** collection flags. By collecting only the types of data that you need, you can reduce both the time and storage used by the **PDCOLLECT** action job. For more information, see [“PDCOLLECT collection flags” on page 175](#).
- The **PDCOLLECT** action creates a configuration data set named *kfj_pdcoll_hlq*.PDCLEF that contains the diagnostic data collection settings. Within this data set is member \$USER, which you can use to customize and save your collection settings. For more information, see [“PDCLEF\(\\$USER\)” on page 176](#).

Note: The **PDCOLLECT** action also creates members DEFAULTS and KCIVARS. These members are for internal use by the utility and for reference only. Do not update these members.

- You can fetch several SDSF DA snapshots for specified started tasks. The default is 10 samples every 10 seconds, and you can customize these settings using parameters **KFJ_PDCOL_DA_SAMPLES** and **KFJ_PDCOL_DA_INTERVAL**.
- When specifying how to store the collected diagnostic data, you can configure your job to produce output in one (or more, if needed) of the following formats:

TERSE

You can store the collected diagnostic data in a terse (.TRS) file by setting parameter **KFJ_PDCOLLECT_COMPATIBILITY** to Y. With this option, the **PDCOLLECT** action uses the IEBCOPY UNLOAD function to package the diagnostic data.

The file *kfj_pdcoll_hlq*.PDCOLPDS.TRS is generated.

Note: The TERSE format option is not available in environments where alternate utilities replace IEBCOPY. If IEBCOPY is not available, set parameter **KFJ_PDCOLLECT_COMPATIBILITY** to N.

PAX

You can store the collected diagnostic data in a PAX (.pax) file on a z/OS® UNIX® System Services file system by setting parameter **KFJ_PDCOLLECT_PAX** to Y and specifying the path in parameter **KFJ_PDCOLLECT_PAX_DIR**.

The file *kcipdcol*.pax is generated.

Note: When producing output in the PAX (.pax) file format, the **PDCOLLECT** action writes the Tivoli Enterprise Monitoring Server started task QA1 files (if collected) to the z/OS® UNIX® file system in hexadecimal print format. Additionally, the hex print data is converted to ASCII format and placed in the /converted subdirectory.

ADRSSU

You can store the collected diagnostic data in a z/OS DFSMSDss dump data set by setting parameter **KFJ_PDCOLLECT_ADDRSSU** to Y. This option requires the use of the KCIALPHA program. KCIALPHA is an APF-authorized version of KCIOMEGA.

The file *kfj_pdc0l_hlq.ADR.ARCHIVE* is generated.

Note: With this option, the **PDCOLLECT** action uses the z/OS DFSMSDss **DUMP** command. You might need authority to run **ADDRSSU**, which is the program that is invoked when using DFSMSDss.

- If you need to redirect your results to a different high-level qualifier, specify the **KFJ_PDCOL_HLQ** parameter in the KCIVARS DD statement.

The **PDCOLLECT** action writes status messages for the job to the KCIPRINT SYSOUT data set.

To collect diagnostic information using the **PDCOLLECT** action, use the following procedure.

Procedure

1. Modify the KFJMAINT sample job in TKANSAM (see example below) to use the **PDCOLLECT** action.
2. (Optional) Specify the **RTE_NAME** and **RTE_PLIB_HILEV** parameters that point to the RTEDEF data set.
3. Specify information about your OMEGAMON address space logs, as follows:
 - If the OMEGAMON address space logs are in the SDSF output queue, specify the **KFJ_PDCOL_JOB_NAME** and **KFJ_PDCOL_JOB_ID** parameters. To collect data for a single job, specify the job name and job ID in parameters **KFJ_PDCOL_JOB_NAME** and **KFJ_PDCOL_JOB_ID**, respectively. To specify multiple started task logs, specify parameter **KFJ_PDCOL_JOB_FILTER** set to Y, and define the job filters by using the asterisk (*) and percent sign (%) as wildcard characters in the **KFJ_PDCOL_JOB_NAME** and **KFJ_PDCOL_JOB_ID** parameter values.
 - If your OMEGAMON address space logs have been copied to a sequential data set, specify the **KFJ_PDCOL_JOB_OUTPUT** parameter, and specify the data set name.
4. (Optional) To override the default values of any of the diagnostic data collection settings (such as, collection flags, output package format, or SDSF DA settings), specify each override parameter and value in the KCIVARS DD statement. For more information about the available collection settings, see [“PDCOLLECT collection flags” on page 175](#) and [“PDCLDEF\(\\$USER\)” on page 176](#).
5. Specify the high-level qualifier for the **PDCOLLECT** action data sets in parameter **KFJ_PDCOL_HLQ**.
6. Submit the updated JCL.

Result

The archive of collected diagnostic data is generated. You can FTP the output data set to IBM Software Support.

Example

The following example collects diagnostic data for the OMEGAMON started task OMEGDS that has the job ID S654321. This job uses the default collection settings that are defined in the PDCLDEF (\$USER) member with the exception of the override setting to omit collection of network information. Because the runtime environment information is provided, the **PDCOLLECT** action can collect all available information.

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=<lpar>
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=<tlib_hlq>.TKANMOD
//KCIFFLOW DD DISP=SHR,DSN=<tlib_hlq>.TKANCUS(KFJMAINT)
//KCIVARS DD *
```

ACTION	PDCOLLECT
RTE_NAME	RTEZOS1
RTE_PLIB_HILEV	TSOUID.MONSUITE

KFJ_PDCOL_JOB_NAME	OMEGDS
KFJ_PDCOL_JOB_ID	S654321
KFJ_PDCOL_COLLECT_NETSTAT	N

PDCOLLECT collection flags

When using the **PDCOLLECT** action to gather diagnostic data, you can choose what content to collect at a detailed level.

Note: The content in this topic is relevant only for environments with PTF UJ98129 for APAR OA68463 (3Q25) applied. If you do not have this maintenance applied, see [“Using PDCOLLECT \(without OA68463\)” on page 177](#).

The collection flags are defined by parameters with the naming pattern **KFJ_PDCOL_COLLECT_***. Each of these parameters accepts Y and N as values.

The following table lists the collection flags that are available and describes the content that each setting collects.

<i>Table 28: PDCOLLECT collection flags</i>	
Parameter	Description
KFJ_PDCOL_COLLECT_STC	Collects the started task log
KFJ_PDCOL_COLLECT_NETSTAT	Collects network information using the netstat utility
KFJ_PDCOL_COLLECT_DASTATS	Collects SDSF Display Active Users (DA) statistics
KFJ_PDCOL_COLLECT_RK	Collects libraries RKANSAMU, RKANPARU, RKANCMDU, RKANDATV, RKCPDEFW, RKD2PAR, RKD2SAM
KFJ_PDCOL_COLLECT_TRG	Collects libraries <i>x</i> KANCMD, <i>x</i> KANSAM, <i>x</i> KANPAR, where <i>x</i> is T or R (shared or full RTE)
KFJ_PDCOL_COLLECT_LISTCAT	Collects LISTCAT command output, which lists all the data sets under the runtime environment qualifiers
KFJ_PDCOL_COLLECT_RTEDEF	Collects the runtime environment definition (RTEDEF) library
KFJ_PDCOL_COLLECT_EMBEDS	Collects the EMBEDS data set
KFJ_PDCOL_COLLECT_USERJCL	Collects the data set specified in the GBL_USER_JCL parameter
KFJ_PDCOL_COLLECT_WCONFIG	Collects the WCONFIG data set
KFJ_PDCOL_COLLECT_SNMP	Collects member RKANSAMU (KN3SNMP) or the data set specified in parameter KN3_SNMP_CONFIG_FILE
KFJ_PDCOL_COLLECT_STDENV	Collects the data sets mapped to the SDTENV DD statement
KFJ_PDCOL_COLLECT_SDADIR	Collects various information about the self-describing agent (SDA) directory, permission bits, contents, spawn_cmd.sh contents
KFJ_PDCOL_COLLECT_QA1	Collects QA1 files from the Tivoli Enterprise Monitoring Server started task
	<p>Note: When producing output in the PAX (.pax) file format, the PDCOLLECT action writes the QA1 files to the z/OS® UNIX® System Services file system in hexadecimal print format. Additionally, the hex print data is converted to ASCII format and placed in the /converted subdirectory.</p>
KFJ_PDCOL_COLLECT_OPSLOG	Collects data sets mapped to the RKMSLG DD statement
KFJ_PDCOL_COLLECT_MANIFEST	Collects component level information

Example

The following example show the collection flags with default settings:

```
* Data collection flags
KFJ_PDCOL_COLLECT_STC      Y
KFJ_PDCOL_COLLECT_NETSTAT Y
KFJ_PDCOL_COLLECT_DASTATS Y
KFJ_PDCOL_COLLECT_RK      Y
KFJ_PDCOL_COLLECT_TRG     Y
KFJ_PDCOL_COLLECT_LISTCAT Y
KFJ_PDCOL_COLLECT_RTEDEF  Y
KFJ_PDCOL_COLLECT_EMBEDS  Y
KFJ_PDCOL_COLLECT_USERJCL Y
KFJ_PDCOL_COLLECT_WCONFIG  Y
KFJ_PDCOL_COLLECT_SNMP    Y
KFJ_PDCOL_COLLECT_STDENV  Y
KFJ_PDCOL_COLLECT_SDADIR  Y
KFJ_PDCOL_COLLECT_QA1     Y
KFJ_PDCOL_COLLECT_OPSLOG  N
KFJ_PDCOL_COLLECT_MANIFEST Y
```

PDCLDEF(\$USER)

You can customize and save your diagnostic data collection settings for use with the **PDCOLLECT** action.

Note: The content in this topic is relevant only for environments with [PTF UJ98129 for APAR OA68463 \(3Q25\)](#) applied. If you do not have this maintenance applied, see [“Using PDCOLLECT \(without OA68463\)” on page 177](#).

The **PDCOLLECT** action creates a configuration data set named *kfj_pdcoll_hlq*.PDCLDEF that contains diagnostic data collection settings.

Within this data set is member \$USER, which you can use to customize and save your settings.

The first time you run the **PDCOLLECT** action, Configuration Manager creates the data set named *kfj_pdcoll_hlq*.PDCLDEF, including member \$USER.

Initially, the \$USER member contains the **PDCOLLECT** action parameters set to program defaults. If you have customized any of the same parameters in the KCIVARS DD statement in your JCL the first time you run the **PDCOLLECT** action, these settings override the default values and are stored in the \$USER member.

After the \$USER member has been created and populated during the first run of the utility, the **PDCOLLECT** action no longer updates the \$USER member. To modify the settings in the \$USER member, you must edit the member directly.

Parameters specified in the KCIVARS DD statement in the JCL override the settings in member \$USER. Any settings that are in the \$USER member do not need to be specified in the KCIVARS DD statement.

Example

The following example shows the KCIPDCOL .PDCLDEF (\$USER) member with default settings:

```
***** Top of Data *****
* Data collection flags

KFJ_PDCOL_COLLECT_STC      Y
KFJ_PDCOL_COLLECT_NETSTAT Y
KFJ_PDCOL_COLLECT_DASTATS Y
KFJ_PDCOL_COLLECT_RK      Y
KFJ_PDCOL_COLLECT_TRG     Y
KFJ_PDCOL_COLLECT_LISTCAT Y
KFJ_PDCOL_COLLECT_RTEDEF  Y
KFJ_PDCOL_COLLECT_EMBEDS  Y
KFJ_PDCOL_COLLECT_USERJCL Y
KFJ_PDCOL_COLLECT_WCONFIG  Y
KFJ_PDCOL_COLLECT_SNMP    Y
```

```

KFJ_PDCOL_COLLECT_STDENV      Y
KFJ_PDCOL_COLLECT_SDADIR     Y
KFJ_PDCOL_COLLECT_QA1        Y
KFJ_PDCOL_COLLECT_OPSLOG     N
KFJ_PDCOL_COLLECT_MANIFEST   Y

KFJ_PDCOLLECT_ADRDSSU        N
KFJ_PDCOLLECT_PAX            N
KFJ_PDCOLLECT_COMPATIBILITY  Y

KFJ_PDCOLLECT_PAX_DIR        $HOME

KFJ_PDCOL_DA_SAMPLES          10
KFJ_PDCOL_DA_INTERVAL        10

***** Bottom of Data *****

```

Note: The following parameters can only be specified in the KCIVARS DD and are not supported in the \$USER member:

```

RTE_PLIB_HILEV
RTE_NAME
KFJ_PDCOL_HLQ
KFJ_PDCOL_JOB_*
KFJ_PDCOL_ADR_ARCHIVE1
KFJ_PDCOL_RESTORE_HLQ1
KFJ_PDCOLLECT_RESTORE1

```

¹These parameters are for use by IBM Software Support only.

Using PDCOLLECT (without OA68463)

Use the Problem Determination Data Collection (PDCOLLECT) action to collect diagnostic information. This topic explains how to perform this task using Configuration Manager without APAR OA68463 applied.

Before you begin

Note: The content in this topic is relevant only for environments without PTF UJ98129 for APAR [OA68463 \(3Q25\)](#) applied. If you have this maintenance applied, see [“Using PDCOLLECT \(with OA68463\)”](#) on page 172.

To use the PDCOLLECT utility, you must install SDSF and be licensed to use it. If you do not have SDSF, you can copy the complete address space logs, including the JES output, RKLVLG, RKPDOU, RKPDLG, SYSPRINT, and so on, into a data set. The DCB information for the output data set can be:

```

Organization: PS
Record format: VB
Record length: 240
Block size: 27998

```

About this task

The Configuration Manager PDCOLLECT utility action collects data that includes the following:

- System configuration
- Network information
- Self-describing agent (SDA) information
- Configuration Manager-related information (RTEDEF and EMBEDS data sets)
- Run Time Environment (RTE) data sets
- Output from the specified job

Note: This feature is available with APAR OA62230 (PTF UJ06864).

Procedure

1. Locate the sample JCL job in the SMP/E target library *thilev*.TKANSAM(KFJMAINT).
2. Specify the following parameters in the KCIVARS DD statement:
 - a. Use **ACTION PDCOLLECT**.
 - b. Specify the **RTE_PLIB_HILEV** and **RTE_NAME** parameters that will point to the required RTEDEF data set.
 - c. If the OMEGAMON address space logs are in the SDSF output queue, specify the **KFJ_PDCOL_JOB_NAME** and **KFJ_PDCOL_JOB_ID** parameters.
 - d. If your OMEGAMON address space logs were copied to a sequential data set, replace the **KFJ_PDCOL_JOB_NAME** and **KFJ_PDCOL_JOB_ID** parameters with the **KFJ_PDCOL_JOB_OUTPUT** parameter, and specify the data set name.
3. Submit the updated JCL.

Result

The file &SYSUID.KCIPDCOL.PDCOLPDS.TRS is generated. You can FTP the output data set to the IBM Support Center.

If you have to redirect your results to a different high-level qualifier, specify the **KFJ_PDCOL_HLQ** parameter in KCIVARS DD statement.

Example

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=<lpar>
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=<tlib_hlq>.TKANMOD
//KCIFFLOW DD DISP=SHR,DSN=<tlib_hlq>.TKANCUS(KFJMAINT)
//KCIVARS DD *
ACTION                PDCOLLECT
RTE_NAME              RTEZOS1
RTE_PLIB_HILEV       TSQUID.MONSUITE

* For PDCOLLECT: following parameters are required
KFJ_PDCOL_JOB_NAME   OMEGDS
KFJ_PDCOL_JOB_ID     S654321
```

Messages

Use the information in these messages to help you diagnose and solve problems running Monitoring Configuration Manager jobs.

Message format

Monitoring Configuration Manager message identifiers have the following format:

KFxnns

where:

KF:

Origin of the message:

KFJ

A Monitoring Configuration Manager workflow.

KFU

The underlying KCIOMEGA program or its APF-authorized version, KCIALPHA. KCIOMEGA is the job template engine that runs Monitoring Configuration Manager workflows.

nnnn

5-digit message identification number.

s

Severity of the message:

I

Informational.

W

Warning to alert you to a possible error condition.

E

Error. Workflow processing typically stops.

The documentation for each message includes the following information:

Explanation

Describes what the message text means, why the message occurred, and what its variables represent.

System action

Describes what the system will do in response to the event that triggered this message.

User response

Describes whether a response is necessary, what the appropriate response is, and how the response will affect the system or program.

KFJ messages

Messages with the prefix KFJ are from Monitoring Configuration Manager workflows.

Many messages from Monitoring Configuration Manager workflows are self-explanatory and do not begin with an identifier. Only the messages that require further explanation have an identifier.

KFJ00001E: *Kpp* version *version* is not supported by this configuration tool

Explanation

IBM Z® Monitoring Configuration Manager checks the installed versions of products to be configured in the runtime environment.

System action

No action is performed. The job ends.

User response

In the JCL that runs Monitoring Configuration Manager, specify a KCIFLOW DD statement that refers to an installation containing product versions supported by Monitoring Configuration Manager. If you have only earlier product versions that are not supported by Monitoring Configuration Manager, then consider upgrading.

KFJ00002I: *Kpp* version *version* is not supported by this configuration tool

Explanation

IBM Z® Monitoring Configuration Manager checks the installed versions of products in an SMP/E installation target library. The *Kpp* product version cannot be configured using Monitoring Configuration Manager.

System action

The system proceeds with job execution.

User response

In the JCL that runs Monitoring Configuration Manager, specify a KCIFLOW DD statement that refers to an installation containing product versions supported by Monitoring Configuration Manager.

If you have only earlier product versions that are not supported by Monitoring Configuration Manager, then consider upgrading.

KFJ00003E: CONFIGURE_ *agent* is set to Y but K_ *pp*_ is not installed.

Explanation

The **GENERATE** action found the CONFIGURE_ *agent* parameter set to "Y" in RTEDEF, but this version of the K_ *pp* agent is not installed.

System action

The **GENERATE** action stops before generating runtime members.

User response

In the JCL that runs Monitoring Configuration Manager, verify that the correct installation data set is specified. Set CONFIGURE_ *agent* to "N" in RTEDEF.

KFJ00004E: CONFIGURE_ *agent* is set to Y but K_ *pp*_ installed version is unknown.

Explanation

The **GENERATE** action found the CONFIGURE_ *agent* parameter set to "Y" in RTEDEF, but could not determine the K_ *pp* version of the agent.

System action

The **GENERATE** action stops before generating runtime members.

User response

In the JCL that runs Monitoring Configuration Manager, verify that the correct installation data set is specified. Set `CONFIGURE_agent` to "N" in RTEDEF.

KFJ00005E: RTE NAMES do not match.**Explanation**

The MIGRATE action has not found the specified RTE in the source WCONFIG data set.

System action

The MIGRATE action stops.

User response

Verify that the `RTE_NAME` and `KFJ_MIGRATE_WCONFIG` values are correct.

KFJ00006E: *rte_name* has not been found in *kfj_migrate_wconfig***Explanation**

The specified RTE member, *rte_name*, has not been found in the migrate source WCONFIG data set listed in the message.

System action

The MIGRATE action stops.

User response

Verify that the `RTE_NAME` and `KFJ_MIGRATE_WCONFIG` values are correct.

KFJ00007E: RTEDEF dataset *rtedef* already exists.**Explanation**

Specified target RTEDEF data set, listed as *rtedef* in the message, already exists so it cannot be created again.

System action

The MIGRATE action stops.

User response

Use a different `RTE_PLIB_HILEV` or specify `CONFIRM=Y` in the job to overwrite the existing RTEDEF data set.

KFJ00008W: Agent *Kpp* is not installed in the target SMP/E**Explanation**

The MIGRATE action has detected that the agent *Kpp* is set to `CONFIGURE_agent_Kpp=Y` in the source WCONFIG, but it is not installed in the target SMP/E environment.

System action

The MIGRATE action sets `CONFIGURE_agent_Kpp=N` and continues.

User response

Verify that the correct SMP/E environment has been set in the job running the MIGRATE action.

KFJ00200E: Parameter required but not specified: *parameter***Explanation**

Before generating runtime members, the **GENERATE** action checks whether this required parameter has been specified.

System action

The **GENERATE** action stops before generating runtime members.

User response

Specify the required parameter in the appropriate RTEDEF member, and then resubmit the job.

KFJ00201E: GBL_USS_TKANJAR_PATH directory *path* does not exist**Explanation**

The **GBL_USS_TKANJAR_PATH** parameter specifies the path of the target z/OS® UNIX® System Services directory that is defined in the SMP/E installation jobs by ddname TKANJAR. Before generating runtime members, the **GENERATE** action checks whether this directory exists.

System action

The **GENERATE** action stops before generating runtime members.

User response

Specify the correct path in the appropriate RTEDEF member, and then resubmit the job. If you do not know the path, contact the person who installed the products.

The default path for the TKANJAR ddname is /usr/lpp/kan/bin/IBM.

KFJ00202E: RTE_USS_RTEDIR must not be a subdirectory of GBL_USS_TKANJAR_PATH**Explanation**

The **RTE_USS_RTEDIR** and **GBL_USS_TKANJAR_PATH** parameters each specify the path of a z/OS® UNIX® System Services directory. **GBL_USS_TKANJAR_PATH** is an SMP/E target directory, or a copy. **RTE_USS_RTEDIR** specifies where to generate runtime members; some runtime environment started tasks also write to files under this directory.

RTE_USS_RTEDIR must not be a subdirectory of **GBL_USS_TKANJAR_PATH** because SMP/E target directories and their descendants should be read-only for most users. However, some OMEGAMON® products write to files under the **RTE_USS_RTEDIR** directory.

Before generating runtime members, the **GENERATE** action checks whether **RTE_USS_RTEDIR** is a subdirectory of **GBL_USS_TKANJAR_PATH**.

System action

The **GENERATE** action stops before generating runtime members.

User response

Change the value of **RTE_USS_RTEDIR**, and then resubmit the job.

KFJ00203E: User requires write access to the *zos_unix_path_parameter* directory *path***Explanation**

The user who runs the job that performs the **GENERATE** action must be able to write to the z/OS® UNIX® System Services directory specified in the parameter, where *zos_unix_path_parameter* is **RTE_USS_RTEDIR** or **TRG_COPY_TKANJAR_PATH**, and *path* is the directory path.

The **RTE_USS_RTEDIR** parameter specifies the path of the z/OS® UNIX® directory where the **GENERATE** action writes runtime members.

The **TRG_COPY_TKANJAR_PATH** parameter specifies the path of the z/OS® UNIX® directory where the **GENERATE** action with option **TRGCOPY** copies SMP/E installation files.

System action

The **GENERATE** action stops before generating runtime members.

User response

Follow your local site practices to grant the user write access to the directory. For example, set the directory permissions to 775. The following z/OS® UNIX® shell command sequence (requires superuser for **chmod**) recursively sets the permissions of `/var/rtehome` and its descendants:

```
echo chmod -R 775 /var/rtehome | su
```

KFJ00204E: An error occurred creating RTE_USS_RTEDIR directory *path*

Explanation

The **RTE_USS_RTEDIR** parameter specifies the path of the z/OS® UNIX® System Services directory where the **GENERATE** action writes runtime members. If this directory does not exist, then the user who runs the job that performs the **GENERATE** action must be able to create this z/OS® UNIX® directory.

System action

The **GENERATE** action stops before generating runtime members.

User response

Follow your local site practices to grant the user write access to create the directory.

KFJ00205E: [GBL_USS_TKANJAR_PATH | KFJ_LOCAL_USS_TKANJAR_PATH] directory *path* is empty

Explanation

The z/OS® UNIX® System Services directory specified in the **GBL_USS_TKANJAR_PATH** or **KFJ_LOCAL_USS_TKANJAR_PATH** parameter must contain installation files.

System action

The **GENERATE** action stops before generating runtime members.

User response

Verify if the **GBL_USS_TKANJAR_PATH** or **KFJ_LOCAL_USS_TKANJAR_PATH** parameter is specified correctly.

KFJ00205W: The length of SYSNAME exceeds 4.

Explanation

Before generating members in the RTEDEF library, the **DISCOVERY**, **GENERATE**, and **MIGRATE** actions verify if the **SYSNAME** parameter exceeds 4 characters in length.

System action

The system action depends on whether **KFJ_SYSNAME** is specified.

- If **KFJ_SYSNAME** is not specified in the **KCIVARS DD** statement in the **JCL**, the system will use the **SYSSMFID** parameter instead of **SYSNAME**.
- If **KFJ_SYSNAME** is specified and it does not exceed 4 characters in length, the system will use the **KFJ_SYSNAME** value instead of **SYSNAME**.

User response

If you want to change the **SYSNAME** system parameter, specify the **KFJ_SYSNAME** parameter in the **KCIVARS DD** statement in the **JCL** **before** submitting **DISCOVERY** and **GENERATE** actions. See the example below.

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIALPHA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUITE.TKANMOD
//KCIFFLOW DD DISP=SHR,DSN=MONSUITE.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION DISCOVER
```

```
RTE_NAME      RTE1
RTE_PLIB_HILEV TSOUID.MONSUITE
KFJ_SYSNAME   MVS1
/*
```

KFJ00206E: RTE_X_SECURITY_EXIT_LIB data set *name* is not allocated

Explanation

The data set specified in parameter **RTE_X_SECURITY_EXIT_LIB** is not allocated. Typically, this data set is allocated during the **CREATE** or **MIGRATE** action.

System action

The **GENERATE** action stops before generating runtime members.

User response

Verify that the **RTE_X_SECURITY_EXIT_LIB** parameter is correctly specified in the RTEDEF (*rte_name*) member. Ensure that the **RTE_X_SECURITY_EXIT_LIB** data set is allocated and contains the required security exits.

KFJ00207E: Full discovery not attempted - APF authorization required

Explanation

The KCIALPHA load library has been used for the DISCOVER action, but it is not APF authorized.

System action

KCIALPHA performed a partial discovery.

User response

Partial discovery was still performed and relevant members might have been created in RTEDEF. These members will not be overwritten by a full discovery process. If full discovery is required, APF authorize the KCIALPHA load library and rerun the job.

KFJ00208E: User requires read access to the *name* directory

Explanation

The user who runs the job that performs the **PACKAGE** action must be able to read this z/OS® UNIX® System Services directory.

System action

The **PACKAGE** action stops before generating DUMP data sets.

User response

Follow your local site practices to grant the user **READ** access to the directory. **WRITE** access is also recommended as it will be used during the **GENERATE** action too.

KFJ00209E: Directory does not exist: *name*

Explanation

z/OS® UNIX® System Services directory does not exist. Specified directory should be created during **GENERATE** action.

System action

The **PACKAGE** action stops before generating DUMP data sets.

User response

Review **GENERATE** JCL Job output log for any z/OS® UNIX® errors.

KFJ00210E: Unable to create PAX archive *name*

Explanation

Unexpected error while trying to create PAX archive on configuration system.

System action

The **PACKAGE** action stops before generating DUMP data sets.

User response

Verify job output for more details. Consider multiple issues: permissions, space shortage.

KFJ00211I: Configuration Manager is allocating data sets using local settings for high-level qualifiers and z/OS UNIX System Services paths

Explanation

During runtime environment generation, the KFJ_LOCAL_* local parameters will be used to allocate runtime environment data sets instead of RTE_* or GBL_* parameters.

System action

The **GENERATE** action performs runtime environment generation using local parameters.

User response

Review the mapping table and make sure that all of the KFJ_LOCAL_* parameters are specified correctly.

KFJ00212E: No DUMP data sets found to deploy

Explanation

The **DEPLOY** action did not find any DUMP data sets to restore.

System action

The **DEPLOY** action stops with a return code of 8.

User response

Check that expected DUMP data sets are available on the system. Verify KFJ_PACK_* parameters, if they are used.

KFJ00213E: Customizing parameter *parameter_name* is not supported by the GENERATE action when using option parameter KFJ_LOCAL_PLIB_HILEV. Use the product provided default.

Explanation

The **KFJ_LOCAL_PLIB_HILEV** parameter was specified in the JCL (KCIVARS DD statement). The **GENERATE** action does not allow certain parameter customization for runtime environment generation for remote systems. In the following example, one of the parameters, **GBL_USER_JCL**, was found in the RTEDEF members, which caused the program to stop with a return code of 8:

```
KFU00002I INVOKE processing is about to commence; MEMBER=KFJOMEGA DDNAME=KCIFLOW
DSN=TEST1.SYSplex.SMPE.TKANCUS Maintenance level: 0A61601
01. Using parameters in TEST1.CM.R683L.RTEDEF(PGVK)
02. Using parameters in TEST1.CM.R683L.RTEDEF(KDS$PARM)
03. Using parameters in TEST1.CM.R683L.RTEDEF(GBL$PARM)
04. Using parameters in TEST1.CM.R683L.RTEDEF(PCK$PARM)
KFJ00213E Customizing parameter GBL_USER_JCL is not supported by the GENERATE action when
using option parameter KFJ_LOCAL_PLIB_HILEV. Use the product provided default.
KFU000086E REXX routine has failed; EXEC=VALVARS RC=8
```

```
KFU00004I KCIOMEGA is ending; RC=8 SYSPLEX=RSPLEX0K LPAR=1234 DATE=2021-08-25 08:23:21
Thursday
```

System action

The **GENERATE** action stops with return code 8.

User response

Review parameter *parameter_name* to ensure that it is not used by the **GENERATE** action. To achieve this, you can comment out this parameter from the RTEDEF members and restart the **GENERATE** action.

If parameter *parameter_name* is related to KD2 operational data set allocation, you can resolve this error by using parameter **KFJ_LOCAL_KD5_RUN_ALLOC** set to value **DEPLOY** or **NONE**.

KFJ00214E: Unable to extract PAX archive

Explanation

An unexpected error occurred while trying to extract the PAX archive on a remote (target) system.

System action

The **DEPLOY** action stops with a return code of 8.

User response

Review the job output for more details. Consider multiple issues: permissions, space shortage, corrupted data set.

KFJ00215W: RTE_X_HILEV_SHARING not supported. Parameter ignored.

Explanation

Monitoring Configuration Manager does not support the **RTE_X_HILEV_SHARING** parameter. The parameter value does not migrate from the existing PARMGEN configuration to the Configuration Manager RTEDEF data set.

System action

The **RTE_X_HILEV_SHARING** parameter value is ignored and not used. Processing continues.

User response

No action is required.

KFJ00216E: OPTION *option* not supported

Explanation

The **OPTION** parameter value specified in KCIVARS is not supported.

System action

The action stops with a return code of 8.

User response

Review the **OPTION** parameter value specified. Make sure that the option value is spelled correctly and that is available for the specific action.

KFJ00217E: OPTION *option_1* not compatible with option *option_2*

Explanation

Certain options are not compatible due to conflicting outcomes.

System action

The action stops with a return code of 8.

User response

Remove one or more of the conflicting options and rerun the job.

KFJ00218E: *\$PARM and/or *\$GLOB members already exist**Explanation**

The **MIGRATE** action checked the existing *rte_plib_hilev*.RTEDEF and found sysplex level members in the library.

System action

The **MIGRATE** action stops with a return code of 8.

User response

Delete or rename the members listed in KCIPRINT and rerun the **MIGRATE** action.

KFJ00219E: *\$lpar members already exist**Explanation**

The **MIGRATE** action checked the existing *rte_plib_hilev*.RTEDEF and found LPAR-specific members in the library.

System action

The **MIGRATE** action stops with a return code of 8.

User response

Delete or rename the members listed in KCIPRINT and rerun the **MIGRATE** action.

KFJ00220E: RTE_NAME member already exists**Explanation**

The **MIGRATE** action checked the existing *rte_plib_hilev*.RTEDEF and found the RTE_NAME member in the library.

System action

The **MIGRATE** action stops with a return code of 8.

User response

Delete or rename the RTE_NAME member listed and rerun the **MIGRATE** action.

KFJ00221I: DEBUG does not impact the deletion of DDs in JES3**Explanation**

JES3 does not support the deletion of DD outputs from the spool, so the function of **OPTION DEBUG** will not have impact on this function.

System action

No impact. Processing continues.

User response

No action is required.

KFJ00222I: Package *data_set_name* not found

Explanation

The **DEPLOY** action expects certain data sets to be available during the restore process. The availability of the data sets depends on the results of the **PACKAGE** action.

System action

No impact. Processing continues.

User response

Review the results from the **PACKAGE** action. Ensure that all data sets created by the **PACKAGE** action are processed during the **DEPLOY** action.

KFJ00223I: Global RTE PDS V1 setting is ON | OFF

Explanation

This message denotes the global persistent data store version 1 status for the runtime environment. This status is the final status that is set by the configuration process and is also written to the KPQHINIT member in RKANPARU. This status applies to all agents and does not reflect the initial status of, for example, **RTE_PDS2_ACTIVATION**, or individual **Kpp_PDS2_ACTIVATION** flags. This value is derived from the combined parameter flags that are used in the runtime environment.

This message is always followed by message [KFJ00224I](#). If PDS V1 is ON, details about why PDS V1 is ON appear after message [KFJ00224I](#). This information includes a list of agents for which PDS V2 is not activated and also indicates if PDS V2 is not activated due to an unsupported agent version. For more information about supported product versions, see [PDS V2 support](#).

For more information, see [How to: Activate PDS V2](#).

System action

No impact. Processing continues.

User response

No action is required.

KFJ00224I: Global RTE PDS V2 setting is ON | OFF

Explanation

This message denotes the global persistent data store version 2 status for the runtime environment. This status is the final status that is set by the configuration process, which is also written to the KPQHINIT member in RKANPARU. This status applies to all agents and does not reflect the initial status of, for example, **RTE_PDS2_ACTIVATION**, or individual **Kpp_PDS2_ACTIVATION** flags. This value is derived from the combined parameter flags that are used in the runtime environment.

For more information, see [How to: Activate PDS V2](#).

System action

No impact. Processing continues.

User response

No action is required.

KFJ00225E: EMBEDS data set invalid or not allocated - *rte_x_override_embeds_lib*

Explanation

If parameter **RTE_X_OVERRIDE_EMBEDS_LIB** is found in the RTE_NAME member of RTEDEF, it must be set to a valid value that corresponds to an existing embeds library. If the parameter value is empty or mistyped, this

error is issued. The message provides the specified parameter value in *rte_x_override_embeds_lib*; an empty string indicates that the parameter value was empty.

System action

Processing stops.

User response

Either correct the value specified in parameter **RTE_X_OVERRIDE_EMBEDS_LIB**, or, if no embed overrides are required, remove the parameter from the RTE_NAME member.

KFJ00226W: Non-secure TEMS communication protocol is used

Explanation

Runtime environment parameter **RTE_TEMS_TRANSPORT_MODE** is set to HTTP. The runtime environment is using the non-secure HTTP communication protocol for all agents.

System action

Processing continues.

User response

Consider changing the communication protocol to HTTPS. For more information, see [Update runtime environment to use HTTPS](#).

KFJ00227E: *hilev_parameter* parameter value must match a data set name used in JCL KCIFLOW DD statement

Explanation

The high-level qualifier value for the SMP/E target libraries that was specified in parameter *hilev_parameter* was not found in the KCIFLOW DD statement in the JCL, where *hilev_parameter* is **GBL_TARGET_HILEV** or **KFJ_LOCAL_TARGET_HILEV**. The KCIFLOW DD statement and the RTEDEF members must point to the same SMP/E target libraries.

For the **GBL_TARGET_HILEV** parameter, the parameter and value are specified in the GBL\$PARM or GBL\$lpar member.

For the **KFJ_LOCAL_TARGET_HILEV** parameter, the parameter and value are specified in the PCK\$PARM or PCK\$lpar member.

For example, in the GBL\$PARM or GBL\$lpar member, the high-level qualifier for the SMP/E target libraries is defined as follows:

```
GBL_TARGET_HILEV    "MONSUITE"
```

In the JCL, this same high-level qualifier value must be specified in the KCIFLOW DD statement, as shown:

```
//UID#ZMCM JOB ,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=ZOS1
//S1 EXEC PGM=KCIOMEGA,REGION=0M,DYNAMNBR=256
//STEPLIB DD DISP=SHR,DSN=MONSUITE.TKANMOD
//KCIFLOW DD DISP=SHR,DSN=MONSUITE.TKANCUS(KFJOMEGA)
//KCIVARS DD *
ACTION GENERATE
RTE_NAME RTE1
RTE_PLIB_HILEV TSOUID.MONSUITE
```

System action

Processing stops.

User response

Ensure that the KCIFLOW DD statement in the JCL points to the same SMP/E target libraries that are used inside the RTEDEF members.

- For a standard **GENERATE** action, verify the **GBL_TARGET_HILEV** parameter value.

- If **KFJ_LOCAL_PLIB_HILEV** is specified in the **KCIVARS DD** statement, verify the **KFJ_LOCAL_TARGET_HILEV** parameter value.

KFJ00228W: RTE_SHARE *xxx* not supported. RTE_SHARE changed to SMP.

Explanation

Configuration Manager supports **RTE_TYPE FULL**, or **RTE_TYPE SHARING** with **RTE_SHARE SMP**. However, the **MIGRATE** action has detected that the **PARMGEN WCONFIG** configuration uses an **RTE_SHARE** value that is not equal to **SMP**. If **RTE_TYPE** is set to **SHARING**, and **RTE_SHARE** is not equal to **SMP**, you will receive this warning message.

System action

RTE_SHARE is reset to the default value, which is **SMP**.

User response

None. If you were using a base library with **PARMGEN**, you might consider using the Configuration Manager target copy feature, which allows you to create and use a copy of your **SMP/E** target libraries. For more information, see [“Using SMP/E target library copies” on page 156](#).

KFJ00229E: Failed to copy files from *source_path* to TRG_COPY_TKANJAR_PATH directory *target_path*

Explanation

The **GENERATE** action failed to copy files from the directory specified in parameter **GBL_USS_TKANJAR_PATH** to the directory specified in **TRG_COPY_TKANJAR_PATH**.

System action

The **GENERATE** action stops before copying installation files.

User response

Ensure that the directory specified in parameter **GBL_USS_TKANJAR_PATH** contains all the required files. Ensure that the directory specified in parameter **TRG_COPY_TKANJAR_PATH** exists and the user who runs the job that performs the **GENERATE** action has the required permissions to create new files.

KFJ00230I: Checkpoint created

Explanation

The checkpoint for the impacted runtime environment has been created in the **WKANPARU** library. It indicates that the **GENERATE** action with **OPTION PREPARE** has completed successfully.

System action

Processing continues.

User response

Review the generated artifacts. For more information, see [GENERATE action with OPTION PREPARE](#).

KFJ00231I: Checkpoint detected

Explanation

The **GENERATE** action detected a checkpoint, which was created by a previous run of the **GENERATE** action with **OPTION PREPARE**.

Note: The **GENERATE** action does not attempt to detect a checkpoint if any of the following options are used: **USS**, **SECEXITS**, **RELINK**, **QUICKLOAD**.

System action

The **GENERATE** action processes the remaining workflow stages, bypassing the stages performed by the **PREPARE** option.

User response

Review the KCIPARSE DD statement output for more details.

KFJ00232I: Checkpoint deleted

Explanation

The **GENERATE** action detected a checkpoint. At the end of processing, the **GENERATE** action attempts to delete the checkpoint from the WKANPARU data set. This message indicates that the checkpoint was deleted successfully.

System action

Processing continues.

User response

No action is required.

KFJ00233E: Prepared RTE and SMP/E maintenance levels do not match

Explanation

The SMP/E maintenance level does not match the maintenance level in the checkpoint that was created by the **GENERATE** action with **OPTION PREPARE**. This happens when additional PTFs are applied to your SMP/E libraries prior to completing the full maintenance cycle for the impacted runtime environment.

System action

Processing stops.

User response

The checkpoint must be refreshed for the impacted runtime environment. With the additional maintenance applied to the SMP/E libraries, use the **GENERATE** action with **OPTION PREPARE** to regenerate the runtime environment work libraries and checkpoint.

KFJ00234W: Unable to delete checkpoint

Explanation

The **GENERATE** action detected a checkpoint. At the end of processing, the **GENERATE** action attempts to delete the checkpoint from the WKANPARU data set. This message indicates that the **GENERATE** action was unable to delete the checkpoint.

System action

Processing continues.

User response

Make sure the checkpoint member \$PREPARE in the WKANPARU data set is not locked by another process. If needed, remove the \$PREPARE member manually.

KFJ00235I: Alias *KFJ_X_ALIAS_DSN* resolved to *KFJ_X_ALIAS_DSN_RESOLVED*

Explanation

If an alias is used for SMP/E target data sets, Configuration Manager must resolve the alias before checking if the KCIFLOW DD statement and the RTEDEF members point to the same SMP/E target libraries. See the related error message [KFJ00227E](#).

System action

Processing continues.

User response

No action is required.

KFJ00236W: Netstat command failed

Explanation

During the **PDCOLLECT** action, a network diagnostics step running a **netstat** utility failed.

System action

Processing continues.

User response

If you want to retry network statistics collection, review the error message printed in the **NETSTAT** member in the **PDCOLLECT** archive. Make sure you have the appropriate authorizations to run the network statistics collection. After the problem is resolved, you can rerun the **PDCOLLECT** utility action.

KFJ00237E: RTEDEF data set %KFJ_I_PLIB_HILEV%.RTEDEF does not exist

Explanation

The Configuration Manager action that issued this message requires the use of a defined runtime environment definition (RTEDEF) library, but the library does not exist.

The **GENERATE**, **DELETE**, and **PACKAGE** actions require the use of the RTEDEF library, but these actions do not create the library. Only the **MIGRATE**, **CREATE**, **DEPLOY**, and **DISCOVER** actions create the RTEDEF library.

System action

Processing stops.

User response

In the **KCIVARS** input data set, review the specified value for parameter **RTE_PLIB_HILEV**, which must point to an existing RTEDEF library.

KFJ00240I: RTE_PLIB_HILEV parameter is not defined in KCIVARS

Explanation

Parameter **RTE_PLIB_HILEV** was not defined in the **KCIVARS DD** for the **PDCOLLECT** action. Some collection flags that normally require runtime environment information are disabled. To collect additional information, you must specify values for both the **RTE_PLIB_HILEV** and **RTE_NAME** parameters.

System action

Processing continues.

User response

To collect more information about your environment, specify the value of parameter **RTE_PLIB_HILEV** for the next run.

KFJ00241I: RTE_NAME parameter is not defined in KCIVARS

Explanation

Parameter **RTE_NAME** was not defined in the **KCIVARS DD** statement for the **PDCOLLECT** action. Some collection flags that normally require runtime environment information are disabled. To collect additional information, you must specify values for both the **RTE_PLIB_HILEV** and **RTE_NAME** parameters.

System action

Processing continues.

User response

To collect more information about your environment, specify the value of parameter **RTE_NAME** for the next run.

KFJ00242E: You must use PGM=KCIALPHA to create an ADRDSSU archive

Explanation

To be able to run the **ADRDSSU** program, you must use the APF-authorized program **KCIALPHA**.

System action

Processing stops.

User response

Specify **PGM=KCIALPHA** in the JCL and rerun the **PDCOLLECT** action.

KFJ00243E: Use PGM=KCIOMEGA to run PDCollect

Explanation

If you archive the collected diagnostic data only to a TERSE (.TRS) or PAX (.pax) file, then you must run the **PDCOLLECT** action with the **KCIOMEGA** program.

System action

Processing stops.

User response

Specify **PGM=KCIOMEGA** in the JCL and rerun the **PDCOLLECT** action.

KFJ00244E: KFJ_PDCOL_JOB_NAME or KFJ_PDCOL_JOB_OUTPUT must be specified

Explanation

The **PDCOLLECT** action must be given a reference to a running OMEGAMON started task, either by specifying a job name or a data set that contains the job output.

System action

Processing stops.

User response

Specify either the **KFJ_PDCOL_JOB_NAME** or the **KFJ_PDCOL_JOB_OUTPUT** parameter.

Note: Using the **KFJ_PDCOL_JOB_NAME** parameter also requires parameter **KFJ_PDCOL_JOB_ID** to be specified if **KFJ_PDCOL_JOB_FILTER "Y"** is not specified.

KFJ00245E: *data_set_name* data set not found

Explanation

A data set that is essential for the **PDCOLLECT** action was not found.

System action

Processing stops.

User response

Check why the data set *data_set_name* does not exist.

KFJ00246E: KFJ_PDCOL_JOB_ID must be specified**Explanation**

If the **PDCOLLECT** action runs without the enabled job filter option, then parameter **KFJ_PDCOL_JOB_ID** is required.

System action

Processing stops.

User response

Specify parameter **KFJ_PDCOL_JOB_ID** set to the job ID of the started task name specified in parameter **KFJ_PDCOL_JOB_NAME**.

KFJ00247I: Job output data set *kfj_pdcoll_job_output* will not be used**Explanation**

For the log analysis, the **PDCOLLECT** action can use either a running started task in SDSF or the job output. If both are specified, then the job output is ignored.

System action

Processing continues.

User response

If you have SDSF and use the combination of job ID and job name to identify the started task, you can remove parameter **KFJ_PDCOL_JOB_OUTPUT** from the parameter list in the **KCIVARS DD** statement.

KFJ00248W: No RTE definition given. Some collection flags will be forced off.**Explanation**

Some collection items require the runtime environment definition to be specified in the **KCIVARS DD** statement. If the runtime environment definition is not provided using parameters **RTE_PLIB_HILEV** and **RTE_NAME**, then Configuration Manager automatically disables the flags for the items that cannot be collected.

System action

Processing continues.

User response

No action is required; however, if you want to collect the diagnostic data that was not collected, specify the runtime environment information in parameters **RTE_PLIB_HILEV** and **RTE_NAME**.

KFJ00249I: Collection flags in effect: *collection_flags_list***Explanation**

The **PDCOLLECT** action lists the items that can be collected in this run.

System action

Processing continues.

User response

No action is required.

KFJ00250E: No collection flags selected. PDCollect cannot proceed.

Explanation

The **PDCOLLECT** action did not find any item to collect.

System action

Processing stops.

User response

Check that at least one flag is set in the PDCLDEF (\$USER) definition file or if there are any messages in the KCIPRINT SYSOUT output indicating that selected flags have been forced to N.

KFJ00251W: No data to terse.

Explanation

When you use the **KFJ_PDCOLLECT_COMPATIBILITY** mode, the **PDCOLLECT** action passes a final data container to the **AMATERSE** program. If no data was collected, the **AMATERSE** program issues a non-zero return code that indicates that it had zero bytes to process.

This result usually means that either the job ID and job name combination was incorrect, not pointing to a valid OMEGAMON started task; or too few collection flags were selected, and no data was found to collect.

System action

Processing stops.

User response

Check that the **KFJ_PDCOL_JOB_NAME** and **KFJ_PDCOL_JOB_ID** parameter values point to a valid OMEGAMON started task and that the collection flags are set correctly.

KFJ00252I: *data_set_name* data set not found

Explanation

A data set that is not essential for the **PDCOLLECT** action was not found, but processing can continue.

System action

Processing continues.

User response

No action is required.

KFJ00253E: Undefined symbols detected in RTEDEF: *&symbol*

Explanation

The validation process encountered symbol *&symbol*, which is not defined. The validation process checks the following locations for the symbol definition:

- System symbols defined in the PARMLIB library or member RTEDEF (SYS@*lpar*)
- User variables defined in members RTEDEF (VAR\$GLOB) or RTEDEF (VAR\$*lpar*)

System action

The system action depends on the **RTE_VALIDATION_LEVEL** parameter setting, as follows:

- If parameter **RTE_VALIDATION_LEVEL** is set to W (Warning) or E (Error), the **GENERATE** action stops with a return code of 8.
- If parameter **RTE_VALIDATION_LEVEL** is set to I (Informational), processing continues, allowing you to use the undefined symbol in your parameter setting. This behavior is acceptable in the following scenarios:
 - You have an undefined symbol that becomes defined when the agent starts, as the symbol can be defined in the PARMLIB library and added dynamically.

- Your runtime environment is being packaged for deployment to another system; during the **GENERATE** action, the validation process does not have access to the remote system symbols.

Tip: Optionally, you can discover the system symbols on the target system and send the results back the configuration system for validation.

User response

Verify that the system symbol definition is correct.

KFJ00254I: A collection configuration data set *kfj_pdc0l_hlq*.PDCLDEF has been created. You can edit the flags in the \$USER member to control what information is collected.

Explanation

The **PDCOLLECT** action created data set PDCLDEF where you can customize your diagnostic data collection preferences. This data set persists after the **PDCOLLECT** action job completes and is used by subsequent **PDCOLLECT** action jobs.

System action

Processing continues.

User response

No action is required.

KFJ00255I: Using collection flags from *kfj_pdc0l_hlq*.PDCLDEF

Explanation

The **PDCOLLECT** action found that data set *kfj_pdc0l_hlq*.PDCLDEF exists and attempts to use preferences found in member \$USER as overrides.

System action

Processing continues.

User response

No action is required.

KFJ00256E: Specified z/OS Unix PDCOLLECT directory cannot be empty

Explanation

You cannot specify an empty value in parameter **KFJ_PDCOLLECT_PAX_DIR** as it is used as the root to create directory *kcipdc0l/*.

System action

Processing stops.

User response

You must specify a non-empty value for the z/OS® UNIX® System Services directory parameter **KFJ_PDCOLLECT_PAX_DIR** in the **KCIVARS DD** statement.

KFJ00257I: When using a job filter, only STC logs and DA panel information can be collected.

Explanation

When job filtering is enabled (parameter **KFJ_PDCOL_JOB_FILTER** set to Y), the **PDCOLLECT** action can only fetch started task and SDSF DA panel information. All other flags are turned off.

System action

Processing continues.

User response

To collect more detailed information, you must specify the job name and job ID values for a specific started task.

KFJ00258E: KFJ_PDCOL_JOB_* parameters do not support filters. Use KFJ_PDCOL_JOB_FILTER 'Y' to enable filters.

Explanation

By default, the parameters for the started task job name (**KFJ_PDCOL_JOB_NAME**) and job ID (**KFJ_PDCOL_JOB_ID**) do not support wildcard characters * and %. To collect diagnostic data for more than one started task, you must enable job filtering. Only started task and SDSF Display Active Users (DA) data can be collected with a filter.

System action

Processing stops.

User response

Enable job filtering by specifying parameter **KFJ_PDCOL_JOB_FILTER** set to Y, and rerun the **PDCOLLECT** action job.

KFJ00259E: KFJ_PDCOL_RESTORE_HLQ is required

Explanation

For the **PDCOLLECT** action to extract the **ADRSSU** archive, a valid high-level qualifier must be specified, as the restore process must rename some collected data sets.

System action

Processing stops.

User response

Set parameter **KFJ_PDCOL_RESTORE_HLQ** to a valid high-level qualifier value. For example, use your TSO user name or any other high-level qualifier to which you have **UPDATE** access.

KFJ00260E: KFJ_PDCOL_RESTORE_HLQ must be a single qualifier

Explanation

A value consisting of multiple qualifiers is invalid. The **ADRSSU** program uses the specified value as the first qualifier for a rename option.

System action

Processing stops.

User response

Specify only one qualifier.

KFJ00262I: PDCOLLECT ADRDSSU archive restored from archive *kfj_pdcoll_adr_archive* to HLQ *kfj_pdcoll_restore_hlq*

Explanation

The **PDCOLLECT** action successfully restored the **ADRDSSU** archive under the given high-level qualifier.

System action

Processing continues.

User response

No action is required.

KFU messages

Messages with the prefix **KFU** are from the **KCIOMEGA** program or its APF-authorized version, **KCIALPHA**.

KCIOMEGA is the underlying job template engine that runs Monitoring Configuration Manager.

KFU00001I: KCIOMEGA is starting; SYSPLEX=*name* LPAR=*name* DATE=*date and time*

Explanation

The **KCIOMEGA** workflow utility is starting. The system, date and time are reported. The **KCIALPHA** utility, the APF-authorized version of **KCIOMEGA**, issues the same message.

System action

Processing continues.

User response

None required.

KFU00002I: *command* processing is about to commence; MEMBER=*name* DDNAME=*name* DSN=*name*

Explanation

A command to process a sub-workflow (**INVOKE**) or read parameters (**CONFIG**) is about to commence. The source member, ddname, and data set name are reported.

System action

For **INVOKE**, the sub-workflow is given control, returning the invoking workflow upon completion. For **CONFIG**, the parameters are read in and made available for the workflow to reference.

User response

None required.

KFU00003I: Workflow task recap: Programs=*program_count* MaxRC=*max_program_rc* REXX=*rexx_exec_count* MaxRC=*max_rexx_rc*

Explanation

Workflow processing has ended. The number of programs and REXX execs that were invoked is reported, along with the maximum return code for each.

System action

Processing ends.

User response

If either return code is greater than zero, then look in the KCIPRINT sysout data set for error or warning messages that indicate the cause of the problem and recommend corrective action. The return code issued in message KFU00004I will help determine if the return codes issued in KFU00003I are acceptable. If you suspect an error, contact IBM Software Support.

KFU00004I: KCIOMEGA is ending; RC=*return_code* SYSPLEX=*sysplex_name* LPAR=*LPAR_name* DATE=*date_and_time*

Explanation

The KCIOMEGA workflow utility is ending. The reported return code, *return_code*, represents the maximum return code for the Configuration Manager action that was performed. The message also provides the system information, date, and time.

Return code	Description
0	The workflow completed with no errors.
4	The workflow completed with one or more warnings. To determine the significance of the warnings, review the preceding messages in the KCIPRINT sysout data set and also, if necessary, KCITRACE. Tip: The DISCOVER action ends with return code 4 if you are performing <i>rediscovery</i> : if you have previously performed discovery for an LPAR, and RTEDEF (<i>Kpp@lpar</i>) members already exist. Instead of overwriting those members, the DISCOVER action writes RTEDEF (<i>Kpp#lpar</i>) members. For details, see “Members created by the DISCOVER action” on page 43.
8	The workflow stopped processing due to an unrecoverable error.

The workflow can also set its own return code via the **STOP** command.

System action

Processing ends.

User response

If the return code is greater than zero, then a previous error or warning message indicates the cause of the problem and recommends corrective action.

KFU00005I: Program is about to be invoked; PROGRAM=*name*

Explanation

The program is about to be invoked to perform a task.

System action

The program is given control, typically as a subtask.

User response

None required.

KFU00006I: Program has ended; PROGRAM=*name* OUTPUT=*output* WORKFLOW=*workflow* STEP=*step* RC=*return_code*

Explanation

The program has ended with the reported return code. The program is deemed to have succeeded because the return code is expected. Specifically, the return code is less than the MAXRC setting of the active workflow.

System action

Processing continues.

User response

No action is required.

KFU00007E: Program has abnormally terminated; PROGRAM=*name* ABEND=*system code*

Explanation

The program has abnormally terminated with the reported system code.

System action

Processing of the workflow stops.

User response

z/OS® MVS™ System Codes in IBM® Documentation describes the abend system code. Additional diagnostic messages may be recorded in the job log. The abend system code or job log messages may recommend or indicate corrective action. Otherwise, re-run the job with a SYSUDUMP DD to generate a dump and report the problem to IBM® Software Support.

KFU00008I: Workflow was told to stop

Explanation

The **STOP** command was issued by the workflow, typically as a result of an error condition, incorrect parameter input, or further action required.

System action

Processing of the workflow stops.

User response

See the KCIPRINT sysout data set for a message that describes why the workflow stopped, and then perform the recommended action.

KFU00009E: Workflow has stopped due to an unrecoverable error

Explanation

An unrecoverable error occurred in the workflow.

System action

Processing of the workflow stops.

User response

A previous error or warning message will indicate the cause of the problem and recommend corrective action.

KFU00010E: DD statement is missing, DDNAME=KCIPRINT or KCITRACE

Explanation

The reported file, either KCIPRINT or KCITRACE, is not allocated. If it is not explicitly specified in the job's JCL as a DD statement, then it is dynamically allocated to SYSOUT=*. Therefore, this error should not occur in normal circumstances and indicates an environmental problem that KCIOMEGA cannot recover from.

System action

Processing stops.

User response

Check in the job log for messages associated with the ddname and take corrective action. Otherwise contact IBM® Software Support.

KFU00011E: DD statement is missing, DDNAME=*name***Explanation**

The reported file, typically a system-generated ddname, is not allocated. The file was dynamically allocated but could not be found. This error should not occur in normal circumstances and indicates an environmental problem that KCIOMEGA cannot recover from.

System action

Processing stops.

User response

Check in the job log for messages associated with the ddname and take corrective action. Otherwise contact IBM® Software Support.

**KFU00012E: MVS™ service has failed; MACRO=BLDL PROGRAM=*name*
R15=*return code* R0=*reason code*****Explanation**

The reported program, about to be run by the workflow, could not be located.

System action

Processing stops.

User response

For return code 4, check that the program exists in the workflow STEPLIB. For all other return codes, refer to the **BLDL** completion codes in IBM® Documentation for corrective action. Otherwise, contact IBM® Software Support.

**KFU00013E: MVS™ service has failed; MACRO=LOAD PROGRAM=*name*
R1=*system code* R15=*reason code*****Explanation**

The reported program, about to be run by the workflow, could not be loaded.

System action

Processing stops.

User response

Check the job log for messages associated with the failed load request. A common abend condition is S806-04 indicating that the module could not be found. Check that the program exists in the workflow STEPLIB. Otherwise, contact IBM® Software Support.

**KFU00014E: MVS™ service has failed; MACRO=LINK PROGRAM=*name*
R1=*system code* R15=*reason code*****Explanation**

The reported program, about to be run by the workflow, could not be invoked.

System action

Processing stops.

User response

Check the job log for messages associated with the failed link request. A typical abend condition is S806-04 indicating that the module could not be found. Check that the program exists in the workflow STEPLIB. Otherwise, contact IBM® Software Support.

KFU00015E: MVS™ service has failed; MACRO=ATTACH PROGRAM=*name* R15=*return code*

Explanation

The reported program, about to be run by the workflow, could not be invoked.

System action

Processing stops.

User response

Check the job log for messages associated with the failed attach request. Refer to the **ATTACH** return codes in IBM® Documentation for corrective action. Otherwise, contact IBM® Software Support.

KFU00016E: Program has failed; PROGRAM=*name* OUTPUT=*output* WORKFLOW=*workflow* STEP=*step* RC=*return_code*

Explanation

The program invoked by the workflow has ended with the reported return code. The program is deemed to have failed because the return code is higher than expected. Specifically, the return code is not less than the MAXRC setting of the active workflow.

System action

Processing stops.

User response

The failing program will typically issue its own error messages to indicate the cause of the problem and recommend corrective action. These messages can be written to the following locations:

- In the KCIPRINT or KCITRACE sysout data set, immediately prior to this message.
- The output data set associated with the program or utility. OUTPUT=*output* in the error message identifies this data set.
- The job log containing the system messages for the job.

IBM Z® Monitoring Configuration Manager uses system and OMEGAMON® utilities to configure the runtime environment. These utilities control their own output messages; those messages will typically not appear in KCIPRINT or KCITRACE.

Utilities such as IEBGENER, IEBCOPY, and IDCAMS write messages to their SYSPRINT output data set. Note that the KCIOMEGA program might have renamed the SYSPRINT ddname to the name of the workflow step that invoked the program.

The OMEGAMON® utility KCIPARSE is used extensively to prepare the runtime members. Error messages might be written to either the associated SYSPRINT or to the job log.

If you cannot locate an associated error message or that message does not recommend corrective action, then contact IBM® Software Support.

KFU00019E: Variable value end quote is missing; VAR=*name source*

Explanation

The reported variable or parameter has an invalid value. The value is assumed to be enclosed in quotes because it starts with a quote, but the end quote is missing. Additional information is recorded in the message to identify the source of the variable, typically the KCIVARS data set or an RTEDEF library member.

System action

Processing stops.

User response

1. Ensure that the variable value, if it contains embedded blanks, is enclosed in quotes.
2. Ensure that the variable value, including quotes, does not extend beyond column 70.
3. Retry the request.

KFU00020E: CONFIG or INVOKE MEMBER= command has exceeded the maximum nesting level

Explanation

The **CONFIG** or **INVOKE** command in the workflow could not be run because it will exceed the maximum level of command nesting permitted.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00021E: Command is invalid: *workflow statement*

Explanation

The reported workflow statement has invalid syntax.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00022E: INVOKE member was not found; Member=*name* DSN=*name*

Explanation

The workflow **INVOKE** command could not find the member to be invoked in the reported data set.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00023W: CONFIG member was not found; Member=*name* DSN=*name*

Explanation

The workflow **CONFIG** command could not find the member to be processed in the reported data set.

System action

Processing continues, without any new parameters.

User response

If the workflow expects and requires the **CONFIG** member then create the member and re-run the job. If the **CONFIG** is intended to provide optional parameters only then no action is required.

KFU00024E: RC is not in the range 0 to 2147483647: *return code*

Explanation

The workflow tried to set the return code variable (RC) with a value outside the allowed range.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00025E: DD statement is missing, DDNAME=*name* RC=*return code*

Explanation

The reported file, typically a system-generated ddname, is not allocated. The file was dynamically allocated but could not be found. This error should not occur in normal circumstances and indicates an environmental problem that KCIOMEGA cannot recover from.

System action

Processing stops.

User response

Check the job log for messages associated with the ddname and take corrective action. Otherwise contact IBM® Software Support.

KFU00026E: DD OPEN error, DDNAME=*name* ABEND=*system code-reason code*

Explanation

The reported file, typically a system-generated ddname, could not be opened. A common abend condition is S913 indicating that access to the data set is not allowed by the security server, such as RACF®.

System action

Processing stops.

User response

Check the job log for messages associated with the ddname and take corrective action. Otherwise contact IBM® Software Support.

KFU00027E: Copy I/O error, DDNAME=*name* ABEND=*system code-reason code*

Explanation

The request to copy data from one file to another has failed.

System action

Processing stops.

User response

Check the job log for messages associated with the ddname and take corrective action. Otherwise contact IBM® Software Support.

**KFU00028E: TCP/IP settings are not available; RET=*return_code*
REAS=*reason_code* REQUEST=BPX1*xxx* GET*yyyyyyyyy***

Explanation

The request to fetch TCP/IP settings has failed. The request originated from either BPX1HST GETHOSTNAME or BPX1GAI GETADDRINFO.

System action

Processing stops.

User response

If this error was seen in security exits processing, check if z/OS Integrated Cryptographic Service Facility (ICSF) is active on the system where the configuration job is run. Otherwise contact IBM® Software Support.

**KFU00030E: Workflow command is not recognized; Member=*name*
Line=*number***

Explanation

The workflow encountered a command that was not recognized. The workflow member name and line number within the workflow identify the offending command.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

**KFU00031E: JCL statement has a syntax error; Member=*name*
Line=*number***

Explanation

The workflow encountered a JCL statement that was not recognized. The workflow member name and line number within the workflow identify the offending statement.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00032E: Workflow command is invalid, Member=*name* Line=*number*

Explanation

The workflow encountered a command with invalid syntax or used out of context. The workflow member name and line number within the workflow identify the offending command.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00037E: String in quotes is not terminated; Member=*name* Line=*number*

Explanation

The workflow encountered a string that started with a quote but was not terminated with a quote. The workflow member name and line number within the workflow identify the offending command.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00040E: JCL statement name is longer than 8; *statement*

Explanation

The reported JCL statement has a name longer than 8 characters. For an EXEC statement this is the step name. For a DD statement this is the ddname.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00041E: JCL statement operation is not EXEC or DD; *statement*

Explanation

The reported JCL statement does not specify a recognized operation. Only EXEC and DD statements are supported.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00042E: DD statement has an unsupported keyword parameter value; *parameter*

Explanation

The reported JCL DD statement has specified a keyword parameter with an unsupported value. Only some of the actual JCL DD statement parameter values are supported in workflows.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00043E: DD statement keyword parameter value is missing; *parameter*

Explanation

The reported JCL DD statement has specified a keyword parameter that has no value. This typically occurs when the value is parameterized, and the parameter is not defined or has no value. Some parameters support missing or null values, in which case the system default is used. Other keyword parameters, such as **DSNAME**, if specified in the DD statement must be resolved to an allowed value.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00044W: DD statement keyword parameter is not recognized; *statement*

Explanation

The reported JCL DD statement has specified a keyword parameter that is not supported in workflows. Only some of the actual JCL DD statement parameters are supported in workflows.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00045E: EXEC statement has an unsupported keyword parameter value; *statement*

Explanation

The reported JCL EXEC statement has specified a keyword parameter with an unsupported value. Only some of the actual JCL EXEC statement parameter values are supported in workflows.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00046E: EXEC statement keyword parameter value is missing; *statement*

Explanation

The reported JCL EXEC statement has specified a keyword parameter that has no value. This typically occurs when the value is parameterized, and the parameter is not defined or has no value. Some parameters support missing or null values, in which case the system default is used. Other keyword parameters, such as **PGM**, must be resolved to an allowed value.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00047W: EXEC statement keyword parameter is not recognized; *statement*

Explanation

The reported JCL EXEC statement has specified a keyword parameter that is not supported in workflows. Only some of the actual JCL EXEC statement parameters are supported in workflows.

System action

Processing stops.

User response

If the problem occurs in an IBM®-supplied workflow then contact IBM® Software Support.

KFU00050E: DYNALLOC request has failed; REQUEST=ALLOC EC=*error* *code* IC=*information code* DSN=*data set name*

Explanation

The DD statement in the workflow failed dynamic allocation with the reported error code. The message severity is reduced to a warning if the dynamic allocation is used only to check for the existence of the data set. Additional messages issued by dynamic allocation will be reported immediately after this message, explaining the problem.

System action

Processing stops when the message severity is an error. Processing continues when the message severity is a warning.

User response

Refer to the additional messages issued by dynamic allocation for corrective action. If the problem persists then contact IBM® Software Support.

KFU00051I: DYNALLOC request has failed; REQUEST=ALLOC EC=*error* *code* IC=*information code* DSN=*data set name*

Explanation

The DD statement in the workflow failed dynamic allocation with the reported error code. The dynamic allocation error is informational because the allocation is used only to check for the existence of the data set, or to allocate it. The WARNING=RETURN option was specified in the DD statement.

See message KFU00050E for more information about dynamic allocation errors.

System action

Processing continues.

User response

None required.

KFU00055E: MVS™ service has failed; MACRO=\$SWAREQ R15=*return* *code* SVA=*address* CB=*control block name*

Explanation

An internal service similar to SWAREQ was invoked to extract an SWA control and has failed.

System action

Processing stops.

User response

If the problem persists then contact IBM® Software Support.

KFU00060E: DD statement is missing, DDNAME=*name* RC=*return code***Explanation**

The reported file, typically a system-generated ddname, is not allocated. The file was dynamically allocated but could not be found. This error should not occur in normal circumstances and indicates an environmental problem that KCIOMEGA cannot recover from.

System action

Processing stops.

User response

Check the job log for messages associated with the ddname and take corrective action. Otherwise contact IBM® Software Support.

**KFU00061E: MVS service has failed; MACRO=DESERV FUNC=GET
R15=*return code* R0=*reason code* DDNAME=*name* DSN=*name*****Explanation**

The DESERV system service was invoked to provide information about a member in a data set, but failed with the reported return code.

System action

Processing stops.

User response

Check the job log for system messages associated with this service and take corrective action. Otherwise contact IBM® Software Support.

**KFU00062I: PDS library data set is empty; MACRO=DESERV
FUNC=GET_ALL R15=*return code* R0=*reason code* DDNAME=*name*
DSN=*name*****Explanation**

The DESERV system service was invoked to provide the list of members in a data set, but the data set is empty.

System action

Processing continues.

User response

None required.

**KFU00063E: MVS™ service has failed; MACRO=DESERV FUNC=GET_ALL
R15=*return code* R0=*reason code* DDNAME=*name* DSN=*name*****Explanation**

The DESERV system service was invoked to provide the list of members in a data set, but failed with the reported return code.

System action

Processing stops.

User response

Check the job log for system messages associated with this service and take corrective action. Otherwise contact IBM® Software Support.

KFU00064E: REXX service has failed; Routine=IRXEXCOM RETC=*return code* VAR=*name***Explanation**

The IRXEXCOM service was invoked to set the value of a REXX variable but failed with the reported return code.

System action

The REXX exec will throw an exception that either causes the exec to fail or give unpredictable results.

User response

Contact IBM® Software Support.

KFU00070E: DD statement is missing, DDNAME=*name* RC=*return code***Explanation**

The reported file, typically a system-generated ddname, is not allocated. The file was dynamically allocated but could not be found. This error should not occur in normal circumstances and indicates an environmental problem that KCIOMEGA cannot recover from.

System action

Processing stops.

User response

Check the job log for messages associated with the ddname and take corrective action. Otherwise contact IBM® Software Support.

KFU00071E: MVS™ service has failed; MACRO=DESERV FUNC=DELETE R15=*return code* R0=*reason code* DDNAME=*name* DSN=*name***Explanation**

The DESERV system service was invoked to delete a member in a data set, but failed with the reported return code.

System action

Processing stops.

User response

Check the job log for system messages associated with this service and take corrective action. Otherwise contact IBM® Software Support.

KFU00072E: MVS™ service has failed; MACRO=STOW FUNC=DELETE R15=12 R0=1234 DDNAME=12345678 DSN=**Explanation**

The STOW system service was invoked to delete a member in a data set, but failed with the reported return code.

System action

Processing stops.

User response

Check the job log for system messages associated with this service and take corrective action. Otherwise contact IBM® Software Support.

KFU00080E: REXX service has failed; Routine=IRXINIT RETC=*return code* REAS=*reason code***Explanation**

The IRXINIT service was invoked to initialize the REXX environment but failed with the reported return code.

System action

Processing stops.

User response

Contact IBM® Software Support.

KFU00081I: REXX routine is about to be invoked; EXEC=*name***Explanation**

The REXX exec is about to be invoked to perform a task.

System action

The exec is given control under the control of the REXX environment.

User response

None required.

KFU00082I: REXX routine has completed; EXEC=*name* OUTPUT=*output* WORKFLOW=*workflow* STEP=*step* RC=*return_code***Explanation**

The REXX exec has ended with the reported return code. The REXX exec is deemed to have succeeded because the return code is expected. Specifically, the return code is less than the MAXRC setting of the active workflow.

System action

Processing continues.

User response

No action is required.

KFU00083E: REXX service has failed; Routine=IRXTERM RETC=*return code***Explanation**

The IRXTERM service was invoked to terminate the REXX environment but failed with the reported return code.

System action

Processing stops.

User response

Contact IBM® Software Support.

KFU00084E: REXX service has failed; Routine=IRXEXCOM RETC=*return code* VAR=*name*

Explanation

The IRXEXCOM service was invoked to set the value of a REXX variable but failed with the reported return code. The service was invoked during REXX initialization to populate the REXX variable pool with the workflow variables.

System action

The REXX exec will be processed, but some of the workflow variables will not be available to the REXX exec.

User response

Contact IBM® Software Support.

KFU00085E: REXX service has failed; Routine=IRXSAY RETC=*return code*

Explanation

The IRXSAY service was invoked to write a message to the REXX output file but failed with the reported return code.

System action

The message is not issued and the REXX exec processing continues. The message was likely written to the KCITRACE sysout data set as a backup.

User response

Contact IBM® Software Support.

KFU00086E: REXX routine has failed; EXEC=*name* OUTPUT=*output* WORKFLOW=*workflow* STEP=*step* RC=*return_code*

Explanation

The REXX exec invoked by the workflow has ended with the reported return code. The REXX exec is deemed to have failed because the return code is higher than expected. Specifically, the return code is not less than the MAXRC setting of the active workflow.

System action

Processing stops.

User response

The failing REXX will typically issue its own error messages to indicate the cause of the problem and recommend corrective action. These messages can be written to the following locations:

- In the KCIPRINT or KCITRACE sysout data set, immediately prior to this message.
- The output data set associated with the REXX exec. OUTPUT=*output* in the error message identifies this data set.
- The job log containing the system messages for the job.

REXX typically writes messages to the SYSTSPRT output data set. The KCIOMEGA program might have renamed the SYSTSPRT ddname to the name of the workflow step that invoked the REXX.

If you cannot locate an associated error message or that message does not recommend corrective action, then contact IBM® Software Support.

KFU00090E: Command is not recognized; REXX=*name* COMMAND=*command*

Explanation

The KCIEXEC host command environment set up for REXX processing did not recognize the command in the exec.

System action

The REXX exec will throw a failure that will either stop exec processing or cause unpredictable results.

User response

If the REXX is supplied by IBM® then contact IBM® Software Support.

KFU00091E: Command is not recognized; REXX1=*name* REXX2=*name* COMMAND=*command*

Explanation

This message is a variation of message KFU00090E. In this case, the REXX exec that issued the command is not the original EXEC REXX=*name* specified in the workflow. REXX1 is the original workflow exec and REXX2 is the most recently called exec.

System action

The REXX exec will throw a failure that will either stop exec processing or cause unpredictable results.

User response

If the REXX is supplied by IBM® then contact IBM® Software Support.

KFU00092W: Command is not supported; REXX=*return code* COMMAND=*command*

Explanation

The KCIEXEC host command environment set up for REXX processing recognized the command but does not support it. For example, the **SUBMIT** command is recognized but is not performed.

System action

The REXX exec treats the command as a null operation and continues.

User response

If the task needs to be performed then you must do that manually after the workflow has completed. For example, you can submit the job after the workflow has completed.

KFU00093W: Command is not supported; REXX1=*name* REXX2=*name* COMMAND=*command*

Explanation

This message is a variation of message KFU00092W. In this case, the REXX exec that issued the command is not the original EXEC REXX=*name* specified in the workflow. REXX1 is the original workflow exec and REXX2 is the most recently called exec.

System action

The REXX exec treats the command as a null operation and continues.

User response

If the task needs to be performed then you must do that manually after the workflow has completed. For example, you can submit the job after the workflow has completed.

KFU00095E: REXX service has failed; Routine=IRXEXCOM RETC=*return code*

Explanation

The IRXEXCOM service was invoked to fetch or store one or more REXX variables but failed with the reported return code. The service was invoked during the processing of a KCIEXEC request.

System action

The REXX exec will continue processing but results in exec or workflow will be unpredictable.

User response

If the REXX is supplied by IBM® then contact IBM® Software Support.

KFU00100W: Full discovery requires APF authorization; PROGRAM=*name*

Explanation

The subsystem discovery service was invoked using a program that is not APF-authorized. KCIALPHA is the APF-authorized version of KCIOMEGA and is recommended for discovery. If KCIALPHA is the program that was used, then the job's STEPLIB data set is not an APF-authorized library.

System action

Discovery continues, all the subsystems will be discovered, but some of their attributes will be incomplete.

User response

For a complete discovery, APF-authorize the TKANMOD library using the following system command:

```
SETPROG APF,ADD,DSNAME=hlg.TKANMOD,SMS
```

Otherwise manually edit the discovery members in the RTEDEF library to complete the process.

KFU00101W: Discovery has abended; ABEND=12345678 SSID=*name* PHASE=*diagnostics*

Explanation

The subsystem discovery service has abended during the analysis of the reported subsystem. If the subsystem is not Db2®, MQ, CICS®, or IMS, then this is not a problem. This problem might occur when discovery is scanning control blocks to detect the type of subsystem, but encounters a control block whose storage is inaccessible.

System action

Discovery ignores this subsystem and moves onto the next subsystem.

User response

If the subsystem is Db2®, MQ, CICS® or IMS, then contact IBM® Software Support. Otherwise the message can be ignored.

KFU00102E: MVS™ service has failed; MACRO=ALESERV FUNCTION=ADD RC=*return code*

Explanation

Discovery is using the ALESERV system service to access, via cross-memory, the subsystem address space in order to extract its discoverable information but the service has failed with the reported return code.

System action

Discovery continues, the subsystem will be discovered, but some of its attributes will be incomplete.

User response

If the problem persists then contact IBM® Software Support.

**KFU00103E: MVS™ service has failed; MACRO=ALESERV
FUNCTION=DELETE RC=*return code*****Explanation**

Discovery is using the ALESERV system service to terminate its access to the subsystem address space but the service has failed with the reported return code.

System action

Discovery continues, the subsystem is discovered, and its attributes will be incomplete.

User response

If the problem persists then contact IBM® Software Support.

**KFU00104E: MVS service has failed; MACRO=LOAD
PROGRAM=EZBNMIFR R1=<*abend_code*> R15=<*reason_code*>****Explanation**

Network discovery could not load program EZBNMIFR, which is the [TCP/IP NMI service](#). A common problem is the ABENDS806-04 module is not found. Typically, EZBNMIFR is in SYS1.CSSLIB, the common services library, which is always included in the LINKLIST.

System action

The TCPIP stacks are discovered with incomplete information.

User response

Copy module EZBNMIFR to the LINKLIST and retry the request. To proceed with the configuration process, first review member KN3@<*lpar*> in your RTEDEF library and correct the parameters.

**KFU00105E: NMI (EZBNMIFR) failed to get TN3270 settings;
RET=<*return_code*> REAS=<*reason_code*> TCPIP=<*name*>****Explanation**

Network discovery issued a [TCP/IP NMI service](#) poll request to obtain the TN3270 servers with an affinity to the reported [TCPIP](#) stack. The request failed with the reported [return](#) and [reason](#) codes. When RET=0000006F, permission is denied (EACCES) because the discovery job is not APF authorized. Message KFU00100W is issued prior to this message and recommends corrective action.

System action

The TCPIP stacks are discovered with incomplete information.

User response

If the return code indicates a program error then contact IBM. To proceed with the configuration process, first review member KN3@<*lpar*> in your RTEDEF library and correct the parameters.

KFU00110I: Some SYSOUT files were not deleted; count=*number***Explanation**

Some of the output files that are not important could not be deleted. The number of files not deleted is reported. The files will remain part of the output for the completed job.

System action

This situation does not impact the function being performed by the workflow and has no effect on the final return code of the job. Processing continues.

User response

If the situation occurs in an IBM-supplied workflow, then contact IBM Software Support.

**KFU00111I: SYSOUT file delete was requested; DDNAME=*name*
DSN=*name*****Explanation**

The output file is not important, and its delete request was sent to JES2. In certain situations, the request may not be successful. This occurrence is normal and can occur when the output is busy being processed by JES2.

System action

Processing continues. If the output was not deleted on the first attempt, then the request will be retried later.

User response

No action is required.

KFU00112I: SYSOUT file is not used; DDNAME=*name* DSN=*name***Explanation**

The output file is eligible to be deleted but could not be found. It is likely that the file is already deleted because it was not used. You might consider removing it from the workflow.

System action

Processing continues.

User response

No action is required.

**KFU00113I: SYSOUT file could not be located; RDJFCB RC=*return_code*
DDNAME=*name*****Explanation**

The output file is eligible to be deleted because it is not important. However, it will not be deleted because its JES data set name could not be located. The file will remain part of the output for the completed job.

System action

This situation does not impact the function being performed by the workflow and has no effect on the final return code of the job. Processing continues.

User response

If the situation occurs in an IBM-supplied workflow, then contact IBM Software Support.

**KFU00114I: JES STATUS request failed: FUNCTION=SSST(80)
STATTYPE=*code* RC=*return_code* SSOBRETN=*return_code*
STATREAS=*reason_code* JOBNAME=*name* JOBID=*id*****Explanation**

The JES subsystem interface Extended status function call (SSI 80) was made to locate the output files created by the workflow steps. The request failed with the reported return and reason codes. SYSOUT files that are not important are not deleted and will remain part of the output for the completed job.

System action

This situation does not impact the function being performed by the workflow and has no effect on the final return code of the job. Processing continues.

User response

If the situation occurs in an IBM-supplied workflow, then contact IBM Software Support.

**KFU00115I: SYSOUT Application Program Interface (SAPI) failed;
FUNCTION=SSS2(79) SSS2TYPE=*code* SSS2UFLG=*flag*
RC=*return_code* SSOBRETN=*return_code* SSS2REAS=*reason_code*
DDNAME=*name* CTOKEN=*string***

Explanation

The JES subsystem interface SYSOUT application program interface (SAPI SSI=79) call was made to delete an output file that is not important. The request failed with the reported return and reason codes. The SYSOUT file is not deleted and will remain part of the output for the completed job.

System action

This situation does not impact the function being performed by the workflow and has no effect on the final return code of the job. Processing continues.

User response

If the situation occurs in an IBM-supplied workflow, then contact IBM Software Support.

**KFU00116I: SYSOUT file delete request failed; DDNAME=*name* DSN=*name*
STATUS=*code* CTOKEN=*string***

Explanation

The output file is not important but could not be deleted. Status information is recorded for diagnostic purposes.

System action

This situation does not impact the function being performed by the workflow and has no effect on the final return code of the job. Processing continues.

User response

If the situation occurs in an IBM-supplied workflow, then contact IBM Software Support.

KFU00140E: REXX service has failed; Routine=IRXEXCOM RETC=*return code* VAR=*name*

Explanation

The IRXEXCOM service was invoked to fetch or store REXX variable but failed with the reported return code. The service was invoked during KCIEXEC command processing of a **VGET** or **VPUT** request.

System action

The REXX exec will throw an exception that either causes the exec to fail or give unpredictable results.

User response

Contact IBM® Software Support.

**KFU00142E: Command syntax or parameter is not supported; REXX=*name*
COMMAND=LISTDS**

Explanation

The REXX exec issued the TSO/E **LISTDS** command. The request was intercepted and processed as a KCIEXEC command. A command parameter was not recognized.

System action

The command is not processed, and the exec throws an exception. The results of the exec are unpredictable unless the exec handles error conditions.

User response

If the REXX exec is supplied by IBM® then contact IBM® Software Support.

**KFU00143E: Command syntax or parameter is not supported;
REXX1=*name* REXX2=*name* VAR=*name* COMMAND=LISTDS****Explanation**

This message is a variation of message KFU00142W. In this case, the REXX exec that issued the command is not the original EXEC REXX=*name* specified in the workflow. REXX1 is the original workflow exec and REXX2 is the most recently called exec.

System action

The REXX exec will throw an exception that either causes the exec to fail or give unpredictable results.

User response

Contact IBM® Software Support.

KFU00144E: REXX service has failed; Routine=IRXEXCOM RETC=*return code* VAR=*name***Explanation**

The IRXEXCOM service was invoked to set the value of a REXX variable but failed with the reported return code. The service was called by the **LISTDS** command.

System action

The REXX exec will throw an exception that either causes the exec to fail or give unpredictable results.

User response

Contact IBM® Software Support.

KFU00145E: DD statement is missing, DDNAME=*name***Explanation**

The reported file, typically a system-generated ddname, is not allocated. The file was dynamically allocated but could not be found. This error should not occur in normal circumstances and indicates an environmental problem that KCIOMEGA cannot recover from. This error is associated with the processing of a TSO/E **CALL** command issued in a REXX exec.

System action

The REXX exec will throw an exception that either causes the exec to fail or give unpredictable results.

User response

Contact IBM® Software Support.

**KFU00147E: Command syntax or parameter is not supported; REXX=*name*
COMMAND=CALL****Explanation**

The REXX exec issued the TSO/E **CALL** command. The request was intercepted and processed as a KCIEXEC command. A command parameter was not recognized.

System action

The command is not processed, and the exec throws an exception. The results of the exec are unpredictable unless the exec handles error conditions.

User response

If the REXX exec is supplied by IBM® then contact IBM® Software Support.

**KFU00148E: Command syntax or parameter is not supported;
REXX1=*name* REXX2=*name* COMMAND=CALL****Explanation**

This message is a variation of message KFU00147E. In this case, the REXX exec that issued the command is not the original EXEC *REXX=*name** specified in the workflow. REXX1 is the original workflow exec and REXX2 is the most recently called exec.

System action

The command is not processed, and the exec throws an exception. The results of the exec are unpredictable unless the exec handles error conditions.

User response

If the REXX exec is supplied by IBM® then contact IBM® Software Support.

**KFU00149E: MVS™ service has failed; MACRO=LOAD PROGRAM=*name*
R1=*system code* R15=*reason code*****Explanation**

The REXX exec issued the TSO/E **CALL** command. The request was intercepted and processed as a KCIEXEC command. The call program could not be loaded.

System action

The command is not processed, and the exec throws an exception. The results of the exec are unpredictable unless the exec handles error conditions.

User response

Verify that the program being called is in the call library. Otherwise if the REXX exec is supplied by IBM® then contact IBM® Software Support.

**KFU00150E: Command syntax or parameter is not supported; REXX=*name*
PROBLEM=*reason* COMMAND=ALLOC****Explanation**

The REXX exec issued the TSO/E **ALLOC** command. The request was intercepted and processed as a KCIEXEC command. A command parameter was not recognized. Not all the actual TSO/E **ALLOCATE** options are supported. The reported problem describes the option that is not supported.

System action

The command is not processed, and the exec throws an exception. The results of the exec are unpredictable unless the exec handles error conditions.

User response

If the REXX exec is supplied by IBM® then contact IBM® Software Support.

KFU00151E: Command syntax or parameter is not supported; REXX1=*name* REXX2=*name* PROBLEM=*reason* COMMAND=ALLOC

Explanation

This message is a variation of message KFU00150E. In this case, the REXX exec that issued the command is not the original EXEC REXX= exec specified in the workflow. REXX1 is the original workflow exec and REXX2 is the most recently called exec.

System action

The command is not processed, and the exec throws an exception. The results of the exec are unpredictable unless the exec handles error conditions.

User response

If the REXX exec is supplied by IBM® then contact IBM® Software Support.

Index

\$

- \$REPORT sysout data set [74](#)
- \$VALRPT sysout data set [74](#)

A

- action options [36](#)
- Actions
 - CREATE [37](#)
 - DELETE [57](#), [59](#)
 - DEPLOY [67](#)
 - DISCOVER [40](#)
 - GENERATE [47](#), [49](#)
 - PACKAGE [64](#)
- actions to run [35](#)
- authority required [15](#)

B

- Basic concept of product [10](#)
- batch interface [34](#)
- batch processing
 - KCIOMEGA [73](#)

C

- commands overview [10](#)
- comparison with legacy PARMGEN [11](#)
- configuration flags [78](#)
- CREATE action [35](#), [37](#)
- creating runtime environment [19](#)

D

- defining OMEGAMON subsystem [17](#)
- definition library for RTE [120](#)
- definition library for RTE
 - initial members [123](#)
 - members [120](#)
 - order of members [122](#)
- DELETE action [35](#), [57](#), [59](#)
- DEPLOY action [67](#)
- DISCOVER action [35](#), [40](#)
- DSNPROUT sysout data set [74](#)

E

- embed override [140](#)
- EMBEDS data set [55](#)
- encryption, password [79](#)
- enhanced 3270 user interface
 - SAF security class name for [86](#)
- EXCLRPT [55](#)
- EXCLRPT sysout data set [74](#)
- exclude list [55](#)
- exits

- security [139](#)

F

- Fold password to uppercase field [86](#)

G

- GBL parameters [93](#)
- GBL_DSN_CSF_SCSFMODE0 parameter [79](#)
- GBL_DSN_IMS_RESLIB [81](#)
- GBL_USS_TKANJAR_PATH parameter [83](#)
- GBL_USS_TKAYHFS_PATH [84](#)
- GBL_UTIL_BINDER [93](#)
- GENERATE action [35](#), [47](#)
- GENERATE action
 - location of runtime members [130](#)
- GENERATE action options [49](#)
- global parameters [93](#)
- global parameters
 - GBL_DSN_CSF_SCSFMODE0 [79](#)
 - GBL_USS_TKANJAR_PATH [83](#)

I

- ICSF load library [79](#)
- interface
 - batch [34](#)

J

- JCL to run product [34](#)

K

- KCI\$IW2R [55](#)
- KCI\$XW2R [55](#)
- KCIOMEGA workflow [73](#)
- KCIPRINT sysout data set [74](#)
- KCITRACE sysout data set [74](#)
- KCIVARS [34](#)
- KCIVARSO sysout data set [74](#)
- KFJ messages [179](#)
- KFJ parameters [99](#)
- KFJ_ADRDSSU_ADMIN [99](#)
- KFJ_EMBEDS_LIB parameter [140](#)
- KFJ_LOCAL_HILEV [101](#)
- KFJ_LOCAL_KD5_RUN_ALLOC [101](#)
- KFJ_LOCAL_PDS_HILEV [102](#)
- KFJ_LOCAL_PLIB_HILEV [102](#)
- KFJ_LOCAL_SMS_MGMTCLAS [102](#)
- KFJ_LOCAL_SMS_STORCLAS [103](#)
- KFJ_LOCAL_SMS_UNIT [103](#)
- KFJ_LOCAL_SMS_VOLUME [103](#)
- KFJ_LOCAL_SMS_VSAM_MGMTCLAS [103](#)

KFJ_LOCAL_SMS_VSAM_STORCLAS [104](#)
 KFJ_LOCAL_SMS_VSAM_VOLUME [104](#)
 KFJ_LOCAL_TARGET_HILEV [104](#)
 KFJ_LOCAL_USS_RTEDIR [105](#)
 KFJ_LOCAL_USS_TKANJAR_PATH [105](#)
 KFJ_LOCAL_USS_TKAYHFS_PATH [105](#)
 KFJ_LOCAL_VSAM_HILEV [106](#)
 KFJ_PACK_DATACLAS [106](#)
 KFJ_PACK_HILEV [107](#)
 KFJ_PACK_MGMTCLAS [107](#)
 KFJ_PACK_STORCLAS [107](#)
 KFJ_PACK_TERSE [108](#)
 KFJ_PACK_UNIT [108](#)
 KFJ_PACK_VOLUME [108](#)
 KFJ_PDCOL_ADR_ARCHIVE [109](#)
 KFJ_PDCOL_COLLECT_data [109](#)
 KFJ_PDCOL_DA_INTERVAL [110](#)
 KFJ_PDCOL_DA_SAMPLES [110](#)
 KFJ_PDCOL_HLQ [111](#)
 KFJ_PDCOL_JOB_FILTER [111](#)
 KFJ_PDCOL_JOB_ID [111](#)
 KFJ_PDCOL_JOB_NAME [111](#)
 KFJ_PDCOL_JOB_OUTPUT [112](#)
 KFJ_PDCOL_RESTORE_HLQ [112](#)
 KFJ_PDCOLLECT_ADRSSU [112](#)
 KFJ_PDCOLLECT_COMPATIBILITY [113](#)
 KFJ_PDCOLLECT_PAX [113](#)
 KFJ_PDCOLLECT_PAX_DIR [113](#)
 KFJ_PDCOLLECT_RESTORE [114](#)
 KFJ_SYSNAME parameter [115](#)
 KFJ_USE_EMBEDS parameter [140](#)
 KFJ messages [198](#)

L

load libraries

ICSF [79](#)

M

member [140](#)

messages [179](#)

messages

KFJ messages [179](#)

KFU messages [198](#)

MIGRATE action [35](#)

MIGRPT sysout data set [74](#)

monitoring components [132](#)

O

OMEGAMON subsystem

defining [17](#)

OPTION parameter [36](#)

P

PACKAGE action [64](#)

Parameters [78](#)

Parameters

different default values [91](#)

for communication between servers [132](#)

initial RTE [79](#)

KFJ_EMBEDS_LIB [140](#)

KFJ_SYSNAME [115](#)

KFJ_USE_EMBEDS [140](#)

significant default values [91](#)

spare parm tables [118](#)

SYSNAME [115](#)

use of variables [137](#)

validation report [74](#)

parameters [93](#), [93](#), [93](#), [93](#), [93](#), [93](#), [94](#), [94](#), [94](#), [99](#), [99](#), [99](#),

[116](#), [116](#), [116](#)

PARMGEN parameter names

GBL_DSN_CSF_SCSFMODE0 [79](#)

RTE_SECURITY_CLASS [86](#)

PARMGEN parameters

KFJ_LOCAL_HILEV [101](#)

KFJ_LOCAL_PDS_HILEV [102](#)

KFJ_LOCAL_SMS_MGMTCLAS [102](#)

KFJ_LOCAL_SMS_STORCLAS [103](#)

KFJ_LOCAL_SMS_UNIT [103](#)

KFJ_LOCAL_SMS_VOLUME [103](#)

KFJ_LOCAL_SMS_VSAM_MGMTCLAS [103](#)

KFJ_LOCAL_SMS_VSAM_STORCLAS [104](#)

KFJ_LOCAL_SMS_VSAM_VOLUME [104](#)

KFJ_LOCAL_TARGET_HILEV [104](#)

KFJ_LOCAL_USS_RTEDIR [105](#)

KFJ_LOCAL_VSAM_HILEV [106](#)

KFJ_PACK_DATACLAS [106](#)

KFJ_PACK_MGMTCLAS [107](#)

KFJ_PACK_STORCLAS [107](#)

KFJ_PACK_TERSE [108](#)

KFJ_PACK_UNIT [108](#)

KFJ_PACK_VOLUME [108](#)

passphrase [86](#)

password encryption [79](#)

password phrase [86](#)

Port number field [88](#)

prerequisites for product [15](#)

privileges to access product [15](#)

R

remote deployment [59](#), [145](#), [154](#)

RTE parameters [94](#)

RTE_NAME [34](#)

RTE_PLIB_HILEV [85](#), [34](#)

RTE_SECURITY_CLASS parameter [86](#)

RTE_SECURITY_FOLD_PASSWORD_FLAG parameter [86](#)

RTE_TCP_HOST parameter [88](#)

RTE_TCP_PORT_NUM parameter [88](#)

RTE_TEMS_TRANSPORT_MODE [89](#)

RTEDEF

- members for variables [137](#)
- runtime environment**
 - basic extra parameters [91](#)
 - creating [19](#)
 - creating or updating [26](#)
 - definition library [120](#)
 - definition library members [120](#)
 - different default parameters [91](#)
 - in a sysplex [132](#)
 - initial library members [123](#)
 - order of definition library members [122](#)
 - parameters [79](#)
 - RTE member locations [130](#)
 - sparse parameters tables [118](#)
- runtime environment parameters [93](#), [94](#)
- runtime environment parameters**
 - RTE_SECURITY_CLASS [86](#)
 - RTE_SECURITY_FOLD_PASSWORD_FLAG [86](#)
 - RTE_TCP_HOST [88](#)
 - RTE_TCP_PORT_NUM [88](#)

S

- security exits [139](#)
- SMP/E target library copy [116](#)
- Specify SAF security class [86](#)

- supported products [8](#)
- SYSNAME parameter [115](#)
- SYSPRINT sysout data set [74](#)
- System Authorization Facility (SAF)**
 - security class name, specifying [86](#)

T

- target copy parameters [116](#)
- TRG parameters [116](#)
- TRG_COPY_HILEV [116](#)
- TRG_COPY_MGMTCLAS [116](#)
- TRG_COPY_NAME [117](#)
- TRG_COPY_STORCLAS [117](#)
- TRG_COPY_TKANJAR_PATH [117](#)
- TRG_COPY_TKAYHFS_PATH [117](#)
- TRG_COPY_UNIT [117](#)
- TRG_COPY_VOLUME [118](#)
- troubleshooting [171](#)

U

- updating runtime environment [26](#)

V

- variables**
 - in parameter values [137](#)

© Copyright International Business Machines Corporation 2020, 2025

US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp

